Un peu plus de SpamAssassin

Table des matières

Utilisation de SpamAssassin

Dans ce document, je choisis de ne pas décrire en détails comment installer et paramétrer SpamAssassin, j'ai déjà quelques tuto qui s'en occupent. On va plutôt regarder comment le rendre le plus efficace possible, les modules à activer, les customisations en fonction des spams spécifiques qu'on reçoit par l'écriture de règles propres à notre installation. La mise à jour régulière est aussi un élément très important.

1 – Modules à activer

En gros, il faut activer tous les modules pour lesquels vous pouvez avoir une certaine connaissance. (Si vous ne les connaissez pas, allez lire leurs documentations). Ça va ralentir un chouia le traitement des courriels, mais ça permet de donner des scores qui sont plus près de la « vérité » (éviter au maximum les faux-positifs et les faux-négatifs).
Dans une installation récente de Spamassassin, j'ai installé/activé entre les modules suivants :

1.1 – FuzzyOcr

C'est un paquet à installer sous ubuntu.
Fichier de configuration très simple.
Il vient avec un dictionnaire de mots à bannir.
Les images gif, jpeg, png, … sont scannés à la recherche des mots bannis.
C'est donc un plugin qui est utilisé pour tenter de détecter les spams cachés dans des images.

1.2 – DCC (Distributed Checksum Clearinghouse)

https://www.dcc-servers.net/dcc/
https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_DCC.html
« The DCC or Distributed Checksum Clearinghouse is a system of servers collecting and counting checksums of millions of mail messages. The counts can be used by SpamAssassin to detect and filter spam. »
On fait appel à un système qui calcule des checksums de courriels et en compte le nombre. Je n'ai pas chercher à savoir comment ça fonctionne en détails.
DCC travaille aussi avec Fuzzy vu plus haut.

Installation :

```
wget 'https://www.dcc-servers.net/src/dcc/dcc.tar.Z'
tar xvzf dcc.tar.Z
cd dcc-2.3.167/
./configure && make && make install
```

Config dans Spamassassin :

```
loadplugin Mail::SpamAssassin::Plugin::DCC
use_dcc 1
dcc_timeout 8
dcc_home /var/dcc/
dcc_path /usr/local/bin/dccproc
add_header all DCC _DCCB_: _DCCR_
```

Exemple d'extrait d'entête de courriel avec DCC :

```
X-Spam-DCC: debian: contrespam.yerbynet.com 1169; Body=1 Fuz1=29 Fuz2=27
```

1.3 – Shortcircuit

Lorsqu'activé, circuit-court (court-circuitage) permet d'économiser les ressources de la machine.
En effet, lorsqu'un courriel match une règle shortcircuit, le reste des règles n'est plus exécuté. Il faut donc l'utiliser de manière intelligente dans ses règles pour éviter l'effet « faux ».
J'ai fini par garder très peu de règles shortcircuit (uniquement celles avec les blacklist/whitelist, mais pas les bayes).
Activation se fait en décommentant la ligne :

```
loadplugin Mail::SpamAssassin::Plugin::Shortcircuit
```

1.4 – WhiteListSubject

C'est ce qui permet d'écrire des règles comme ceci et qu'elles soient prises en compte par SpamAssassin :

```
blacklist_subject Shoppers Drug Mart Customers
whitelist_subject [TSM *]
```

Donc, sans faire de règles très complexes, on arrive à filtrer sur les sujets des courriels grâce à ce module.
L'activation se fait avec cette ligne :

```
loadplugin Mail::SpamAssassin::Plugin::WhiteListSubject
```

1.5 – Pyzor, Razor

https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_Pyzor.html
https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_Razor2.html

http://pyzor.org/

Pyzor et Razor sont des réseaux de détection de spam sur « dénonciation » d'utilisateurs.
Ils sont activés par défaut dans Spamassassin. Si ce n'est pas le cas pour votre installation, installez les et activer les.
La config est très simple :

loadplugin Mail::SpamAssassin::Plugin::Pyzor
use_pyzor 1
pyzor_options --homedir /repertoire/.pyzor
pyzor_timeout 10
loadplugin Mail::SpamAssassin::Plugin::Razor2
use_razor2 1

1.6 – TextCat

https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_TextCat.html
Très intéressant !
Il tente de trouver la langue utilisée dans le message. Une directive ok_languages permettra de préciser les langues qui sont OK pour nous et celles qui sont KO (UNWANTED_LANGUAGE_BODY).
Config :

loadplugin Mail::SpamAssassin::Plugin::TextCat
ok_languages en fr
score UNWANTED_LANGUAGE_BODY <score_desiré>
*add_header all Languages _LANGUAGES_*

Ici, on est d'accord pour le français et l'anglais.
Voici un exemple d'extrait d'entête :

X-Spam-Languages: en

On pourrait demander à spamassassin de tenter de détecter plus de langues (opération qui gourmande en CPU). L'activation de cette fonction se fait en ajoutant la ligne de config :

inactive_languages

Si vous êtes très sûr de vos langues, vous pouvez très bien augmenter le score de *UNWANTED_LANGUAGE_BODY.*

Une directive semblable à ok_language est **ok_locales.**
ok_locale concerne le jeu de caractères utilisé dans le courriel.
On peut donc avoir une ligne de configuration comme ceci :

ok_locales en fr

1.7 – TxRep

https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_TxRep.html

On parle ici de la réputation du sender.
Il faut désactiver le plugin AWL (Auto-WhiteList) afin que l'activation de TxRep soit prise en compte.

```
loadplugin Mail::SpamAssassin::Plugin::TextCat
#loadplugin Mail::SpamAssassin::Plugin::AWL
use_txrep 1
```

1.8 – Phishing, HashBL

« This plugin finds uris used in phishing campaigns detected by OpenPhish or PhishTank feeds. »
On deal avec des listes d'URIs dans ces 2 modules.
https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_HashBL.txt
https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_Phishing.txt
Se basent donc sur les URIs et les domaines des emails (DNS BL) pour donner des scores aux courriels.

1.9 – OLEVBMacro

https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin_Plugin_OLEVBMacro.txt
Détecte les Object Linking and Embedding (OLE) Macro et donne des scores.

1.10 – Autres modules activés par défaut

URIDNSBL : Recherche les URL dans les DNS BL
AutoLearnThreshold : Limite supérieure pour apprendre automatiquement du ham et limite inférieur pour apprendre automatiquement du spam.
SPF : Faire du Sender Policy Framework (SPF)
DKIM : Faire du Domain Keys Identified Mail (DKIM)

1.11 – KAM (KEVIN A. MCGRAIL)

Ce n'est pas un module, mais c'est un ensemble de règles qui semble maintenu à jour. Je l'ai découvert en lisant de la documentation sur Zimbra : https://wiki.zimbra.com/wiki/Anti-spam_Strategies
On retrouve de la documentation ici https://mcgrail.com/template/projects#KAM1 et là https://cwiki.apache.org/confluence/display/SPAMASSASSIN/CustomRulesets pour l'installer et l'utiliser.
Un exemple de règle fournie par KAM :

```
#XEROX SCANS
header      __KAM_XEROX1   Subject =~ /Scan from a Xerox WorkCentre Pro \#\d+|Scanned from a Xerox Multifunction Device/i
meta        KAM_XEROX      (__KAM_XEROX1 + (KAM_IFRAME && T_HTML_ATTACH) + KAM_RAPTOR_ALTERED >= 2)
score       KAM_XEROX      5.0
```

describe KAM_XEROX Likely Fake Xerox Attachment

1.12 – SpamHaus

Si vous avez du budget, vous pouvez aussi investir dans un abonnement professionnel chez SpamHaus : https://docs.spamhaustech.com/datasets/docs/source/index.html
Vous pouvez aussi l'intégrer à votre serveur SMTP pour faire des REJECTs directement dans les sessions SMTP.

1.13 – SURBLs

http://www.surbl.org/usage-policy
Une fois, j'ai pris un trial chez eux pendant près de 2 mois, je ne les ai pas trouvés particulièrement forts. Tout ce que mes autres techniques de détection n'avaient pas pu détecter n'a pu être détecté par SURBL. J'ai donc décidé que ça ne valait pas le coût pour mon installation.
Généralement, ils offrent des périodes d'essai de 1 mois. C'est donc bon à essayer.
Si vous n'êtes pas un gros consommateur, vous pouvez vous contenter de la version gratuite.

2 – Mise à jour

Il s'agit d'exécuter sur une base quotidienne le script **/etc/cron.daily/spamassassin**. Il faut faire en sorte qu'il roule correctement et mette à jour de spamassassin. Quand il roule normalement, il devrait apporter des modifications dans le dossier **/var/lib/spamassassin/3.xyz/updates_spamassassin_org/**.
C'est un tout petit chapitre, mais c'est très très important que cette mise à jour des règles (par défaut) de spamassassin fonctionne. En gros, c'est un **sa-update** qui est fait.

Les règles mises à jour avec le script décrit ci-dessus ne sont pas spécifiques à notre spamassassin.
Un outil tel que **sa-learn** permet de tenir compte des rétro-actions de nos propres clients (c'est donc spécifique à notre installation). Plusieurs techniques existent, l'essentiel est de passer le paramètre **--ham** pour du courriel taggué fautivement spam, et le paramètre **--spam** pour du courriel non classé spam alors qu'il le devrait.
sa-learn --ham <message_classé_spam_mais_qui_nen_nest_pas>
sa-learn --spam <message_non_classé_spam_mais_qui_en_est_un>

On peut demander aux usagers de transférer les courriels vers des comptes de courriels créés à cet effet. Par exemple, rediriger vers spam@yerbynet.com tout ce qui est spam mais n'a pas été détecté, et vers ham@yerbynet.com tout ce qui n'est pas spam mais qui a été marqué. Un cron avec sa-learn pourrait par la suite rouler de temps en temps pour faire sa moulinette et vider les boîtes en question.

Dans un logiciel tel que Zimbra, le fait de cliquer sur le bouton [Spam] ou le bouton [Pas spam] déclenche des actions qui redirigent les courriels vers des boîtes. Il faudra donc analyser ces boîtes à lettres avec sa-learn pour tenir compte des clics des usagers.

3 – Écriture de règles custom

C'est la partie la plus intéressante de ce document. C'est la raison réelle pour laquelle j'ai décidé de rédiger ces lignes et j'espère que ça vaudra la coût. Ça fait au moins 10 ans que j'écris des règles (assez basiques tout de même) de spamassassin, et je me suis rendu compte que je n'ai jamais rédigé quoi que ce soit sur le sujet.
Alors, c'est parti!

3.1 – Blacklist_from

Comme le nom l'indique, blacklister selon le From :

blacklist_from accounts@post.com
*blacklist_from *@mellingrush.eu*

On se fait spammer solide avec des emails venant de accounts@post.com, et des emails venant de plusieurs comptes sur le domaine mellingrush.eu. On ne va pas se mettre à faire plus d'analyse, blacklist_from, et l'affaire est réglée.

Une autre version du blacklist_from consiste à ajouter « in replyto ».
Il y a des spammeurs un peu plus malins qui utilisent un lot assez diversifié d'emails dans le From, par contre, ils gardent le champ replyto à une seule et unique adresse email. On va donc viser le champ reply-to de cette manière (vu que le from est inutilisable dans une telle situation) :

blacklist_from reply@Date.com in replyto

Donc, lorsque spamassassin retrouve reply@Date.com dans le reply-to d'un courriel, il le marque tout de suite comme du spam. C'est assez pratique.

3.2 – Blacklist_subject

On va blacklister des sujets, tout simplement :

blacklist_subject Password Expired !!!!!!

Tous les courriels avec comme sujet contenant « Password Expired !!!!!! » seront considérés spam.

Il existe aussi whitelist_subject dans le même style que blacklist_subject.

3.3 – Whitelist_from

Il y a des compagnies qui gèrent tellement mal leurs emails que ceux-ci tombent tout le temps dans les spams. Cependant, si c'est une compagnie avec laquelle vous faîtes affaire réguliement, vous avez peut-être intérêt à accepter leurs courriels. C'est ainsi qu'on est amené à faire des règles de whitelist_from.
Exemple :

*whitelist_from *@yerbynet.com*

Aucun des emails du domaine yerbynet.com ne sera traité comme spam.

3.4 – Blacklist_to et Whitelist_to

On imagine assez facilement que ce sont les correspondants des directives avec les from.

whitelist_to, je l'ai utilisé une seule fois parce qu'un utilisateur m'a dit qu'il voulait recevoir ses spams et les traiter lui-même. Spamassassin ne traite donc pas le message lorsqu'il voit son email comme étant destinataire.
Exemple :

whitelist_to jeveuxspam@yerbynet.com

blacklist_to, c'est assez intéressant. Des spammeurs qui cachent les adresses destinations en BCC mais qui utilisent des adresses spécifiques dans le TO.

blacklist_to e6x83l98llczabeewvbw24iyb4dc7q77@bm5150.com

3.5 – Traitements des extensions de fichiers

À partir d'ici, et pour toute la suite, il est important de noter que le noms donné à une règle est complètement dépendant de la volonté et du goût de celui qui baptise sa règle. L'essentiel est de ne pas avoir deux fois le même nom. Je rajoute donc le préfixe **YERB_** à mes règles, et je fais en sorte qu'il n y ait pas de collisions dans mes suffixes.

Exemple 1 :
On va créer une règle qui match les fichiers joints avec des extensions **exe**, et on va ajouter un score pour celle-ci.

*mimeheader **YERB_EXE_ATTACHED** Content-Type =~ /**.exe**/i*
*score **YERB_EXE_ATTACHED** 10*

Le nom de la règle est : YERB_EXE_ATTACHED
Le score attribué est 10.

Exemple 2 :

*mimeheader **YERB_ISO_ATTACHED** Content-Type =~ /**.iso**/i*
*score **YERB_ISO_ATTACHED** 10*

Ici, on regarde les fichiers **iso**, la règle s'appelle YERB_ISO_ATTACHED et le score est 10.

Exemple 3 : Un peu plus précis

mimeheader YERB_FAC_DOC_ATTACHED Content-Type =~ /Facture[0-9]+\.[azAZ]+\.doc/i
score YERB_FAC_DOC_ATTACHED 1.3

C'est un .doc un peu particulier, on ajoute donc un score pas très élevé.

3.6 – Traitements sur les entêtes

Exemple 1 :

*header YERB_SUBJECT_TD_NOTICE Subject =~ /.*TD Business Notice.*/i*
score YERB_SUBJECT_TD_NOTICE 20
describe YERB_SUBJECT_TD_NOTICE BANK_PHISHING

Ce qui nous importe dans cette règle est le sujet. Tout courriel dont le sujet contient « *TD Business Notice »* verra son score augmenté de 20 par cette règle.

Exemple 2 :

header YERB_FROM_DATE_TLD From =~ /.+\@.+\.date/
score YERB_FROM_DATE_TLD 3
describe YERB_FROM_DATE_TLD SPAM_FROM_DATE

On augmente de 3 les scores des emails venant de sous-domaines du TLD .date.

3.7 – Règles sur les BODY

Exemple 1 :

body YERB_GBAGBO /mariegbagbo\@presidency\.com/i
score YERB_GBAGBO 7

Si le corps du message contient mariegbagbo@presidency.com, on ajoute un score de 7.

Exemple 2 :

```
body YERB_BODY_BITCOIN /bitcoin\b/i
score YERB_BODY_BITCOIN 2
```

On va à l'attaque du mot **bitcoin** dans les corps des emails.
Caractère **b** : est mis pour matcher exactement le mot bitcoin
Caractère **i** : parce qu'on ne veut pas respecter la casse (insensitive).

3.8 – Règles sur les URIs

Exemple 1 :

```
uri YERB_URI_XYZ  /.+\..+\.xyz\/[a-z]+/
score YERB_URI_XYZ 1.6
describe YERB_URI_XYZ URI_TLD_XYZ
```

Quelques soucis avec le TDL .xyz m'ont poussé à mettre en place cette règle (pas très pénalisante – score juste à 1.6).

Exemple 2 :

```
uri YERB_URI_DE_PAY  /.+\.de\/.*(payment|Payment|address|Address).*/
score YERB_URI_DE_PAY 5.5
```

Un score de 5.5 (un peu plus sérieux) pour ces URLs d'Allemagne qui nous demandent de payer.

3.9 – Les métas

Ce sont des règles un peu plus complexes qui font appel à plusieurs sous-règles.

Exemple 1 :

```
header __YERB_SUBJ_ACC_SUSPEND      Subject =~ /.*account.+(suspended|activated).*/i
uri __YERB_URI_PHP /.+\/.+\.php/
meta YERB_EMAIL_ACC_SUSPEND ( __YERB_SUBJ_ACC_SUSPEND && __YERB_URI_PHP )
score YERB_EMAIL_ACC_SUSPEND 3
```

Un courriel avec un sujet semblable et contenant une URL sous cette forme qui se termine par .php se verra ajouter un score de 3. Il faut que les 2 sous-règles *__YERB_SUBJ_ACC_SUSPEND* et __YERB_URI_PHP matchent.

Exemple 2 :

```
body __YERB_MAIL_SIZE1 /mailbox/i
body __YERB_MAIL_SIZE2 /size/i
body __YERB_MAIL_SIZE3 /reached/i
body __YERB_MAIL_SIZE4 /quota/i
body __YERB_MAIL_SIZE5 /limit/i
```

```
   body __YERB_MAIL_ACC /account/i
   uri __YERB_URI_MAIL1 /.+\/.+\.php.*email.*/i
   uri __YERB_URI_MAIL2 /.+\/.+\/mail?\.php.*/i
   meta YERB_MB_QUOTA ((( __YERB_MAIL_SIZE1 + __YERB_MAIL_SIZE2 + __YERB_MAIL_SIZE3 +
__YERB_MAIL_SIZE4 + __YERB_MAIL_SIZE5 + __YERB_MAIL_ACC ) > 1 ) && (( __YERB_URI_MAIL1 +
__YERB_URI_MAIL2 ) > 0 ))
score YERB_MB_QUOTA 1.5
```

Au moins 2 termes dans le groupe (mailbox, account, …) et au moins une URL similaire => se faire chopper par la règle *YERB_MB_QUOTA* et avoir +1.5.

Exemple 3 :

```
   body YERB_BODY_NICKDARKNET /My nickname in darknet is .*/i
   body YERB_BODY_YOURPASSWIS /So, your password from .*/i
   header YERB_SUBJECT_ISHACKED Subject =~ /.*yerbynet\.com is hacked/i
   score YERB_BODY_NICKDARKNET  2
   score YERB_BODY_YOURPASSWIS 1
   score YERB_SUBJECT_ISHACKED 2
      meta YERB_META_HACKED ( YERB_BODY_NICKDARKNET + YERB_BODY_YOURPASSWIS +
YERB_SUBJECT_ISHACKED > 1 )
   score YERB_META_HACKED 50
```

Dans ce cas-ci, c'est une scam avec une certitude assez élevée, on ajoute donc un score de 50.

Sources :

- https://spamassassin.apache.org/
- https://spamassassin.apache.org/full/3.4.x/doc/Mail_SpamAssassin.html
- http://johnbokma.com/spam/spamassassin-cookbook.html
- https://cwiki.apache.org/confluence/display/SPAMASSASSIN/WritingRules
- https://cwiki.apache.org/confluence/display/SPAMASSASSIN/SpamAssassinRules
- https://cwiki.apache.org/confluence/display/SPAMASSASSIN/CustomRulesets
- https://www2.cs.duke.edu/csl/faqs/spam/spamassassin