

Фильтруем спам в Exim с помощью SpamProbe

Павел Литвинов

Основой почтового сервера в нашей организации был Exim. Он отлично справлялся со своей задачей, при том что работал на достаточно слабом компьютере. Но через какое-то время количество нежелательной почты (спам) в организации возросло до такой степени, что это стало серьезной проблемой. И эту проблему надо было как-то решать.

Выбираем подходящий антиспам-фильтр

Стандартными средствами Exim отсеивалось большое количество спама, но не все. Посему было решено «прикрутить» к нему какой-нибудь антиспам-фильтр. Сказано – сделано. Небольшие поиски в Интернете определили два основных претендента:

- SpamAssassin;
- DSPAM.

Первый хоть и был «раскручен», обладал, на мой взгляд, одним, но весомым недостатком. Учитывая, что эта программа была написана на Perl, она сильно нагружала сервер при большом количестве входящей почты. А если взять в расчет то, что на почтовом сервере еще работал антивирус ClamAV, который и без того слабую машинку буквально вводил в ступор при анализе писем с вложенными файлами, было решено не использовать этот вариант.

Второй мне понравился больше, но я не смог в нем найти «белый список» с возможностью ручного редак-

тирования (не автообучаемый), да и не хотелось устанавливать на и так загруженный компьютер базу данных MySQL, которая была необходима для полноценной работы DSPAM. В общем, и этот вариант мне не подошел.

Начался второй этап поиска. Присматривая порты на момент ключевого слова «SPAM», мне на глаза попался SpamProbe. Что сразу понравилось, программка была написана на C++, была маленькой, шустрой в работе и не требовала установки базы данных, в отличие от DSPAM. К тому же алгоритм работы программы построен на основе математической теоремы Байеса. А ведь именно этот метод статистической фильтрации является наиболее удачным и используется практически всеми спам-фильтрами, в том числе и упомянутыми ранее.

Метод Байеса подразумевает использование статистической, оценочной базы, разделенной на две части, одна из которых содержит черный список слов, а другая – белый. При анализе письма подсчитывается количество совпадений каждого отдельного слова

(токена) со списками в базе, и на основании этого вычисляется оценка. Оценка эта колеблется в диапазоне от 0 до 1, где значение 0 означает отсутствие признаков спама, 1 – полную уверенность в том, что это спам.

По своей сути SpamProbe является маленькой программой, дающей оценку письму с определенной долей вероятности. Она даже не имеет конфигурационного файла, хотя, как мы увидим дальше, эта особенность не мешает нам создать на его основе спам-фильтр не только не уступающий, но даже превосходящий в некоторых моментах спам-фильтры, упомянутые мной выше.

Приступаем к установке и настройке

Установку программы производим, как всегда, из портов:

```
# cd /usr/ports/mail/spamprobe
# make -s install clean
```

После установки создаем базу, где SpamProbe будет хранить базу данных токенов (слов или словосочетаний, ко-

торые с определенной долей вероятности могут присутствовать в письме, содержащем или не содержащем спам):

```
# spamprobe -d /var/lib/spamprobe create-db
```

где /var/lib/spamprobe – путь к папке базы данных.

Изучение документации на предмет сопряжения SpamProbe и Exim в виде одного-единственного файла README.txt не дало ровным счетом ничего. Продолжительные поиски в Интернете вывели на блог, где был описан один из способов, который я, в свою очередь, с некоторыми доработками и комментариями предлагаю вашему вниманию.

В конфигурационный файл Exim добавляем следующие строки:

```
system_filter      = /usr/local/etc/spamprobe/exim.filter
# system_filter_user = mailnull
# system_filter_group = mail

#####
# ACL CONFIGURATION #
# Specifies access control lists for incoming SMTP mail #
#####
```

Проблема в том, что я так и не смог заставить полноценно работать системный фильтр не от привилегированного пользователя (root).

Комментарии приведены только для того, чтобы вы могли сориентироваться, в какое именно место конфигурационного файла Exim вставлять ссылку на системный фильтр.

Теперь создаем непосредственно сам файл системного фильтра /usr/local/etc/spamprobe/exim.filter:

```
# Exim filter
# Ни в коем случае не удаляйте верхнюю строку

# Путь к лог-файлу
logfile /var/log/exim/spam-filter.log

# Указываем максимальный размер письма, которое будет
# подвержено анализу (Kb).
# Число взял из конфигурационного файла DSPAM
add 307200 to n0

# Проверяем письмо только один раз, даже если Exim
# с первого раза его не сможет доставить.
# Нечего плодить заголовки X-SpamProbe:
if first_delivery
then
if $message_size is above $n0
then
# Если не нужно вести лог-файл, закомментируйте строку
# ниже
logwrite "SPAM FILTER: Размер письма от $return_path
превышает $n0 Kb ; Письмо не будет проходить проверку"
else
headers add "X-SpamProbe: "
${run {/usr/local/etc/spamprobe/msgscore.sh}
${message_id} ${quote:${message_headers}}} "
${value} {expansion failed} }"
# Если не нужно вести лог-файл, закомментируйте строку
# ниже
logwrite "SPAM FILTER: Письмо от $return_path
успешно прошло проверку; $value"
endif
endif
```

И наконец скрипт msgscore.sh:

```
#!/bin/sh

# Путь к месту, где почта хранится до момента доставки
spooldir=/var/spool/exim/input
```

```
# Путь к базе, где SpamProbe хранит токены
dbdir=/var/lib/spamprobe

# Полный путь к программе SpamProbe
path_spamprobe=/usr/local/bin/spamprobe

# Находим полный путь к нашему письму.
# Если бы не опция "split_spool_directory = true"
# в конфигурационном файле Exim он бы соответствовал
# переменной '$spooldir/$1-D'
path_file="/usr/bin/find $spooldir -name $1-D -print"

echo "$2" > $spooldir/$1-M
sed '1 s/.*/' $path_file >> $spooldir/$1-M
$path_spamprobe -8 -d $dbdir score $spooldir/$1-M
rm $spooldir/$1-M
exit 0
```

После этого можно сказать, что основная часть работы выполнена, но не будем торопиться. Проверить работу фильтра просто, для этого необходимо пропустить пару писем через ваш сервер и посмотреть сначала лог-файл, а потом – появился ли в исходном тексте письма заголовок следующего вида:

```
X-SpamProbe: GOOD 0.9999999 a1c5e42b21f4e60f55596ca70ae48e7f
```

где GOOD – говорит о том, что ваше письмо с вероятностью 0,9999999 не является спамом. Набор из 32 букв и цифр – уникальный идентификатор письма.

Еще один момент. Пока вы не начнете обучать SpamProbe, все письма будут промаркированы как GOOD.

Если все прошло удачно, приступаем к следующему этапу. Настроим в Exim обработку промаркированных SpamProbe писем. Для этого в секцию ROUTERS CONFIGURATION добавим следующее:

```
#####
# ROUTERS CONFIGURATION #
# Specifies how addresses are handled #
begin routers

# SpamProbe start #####
SP_spam_router:
driver = accept
domains = +local_domains
local_part_prefix = spam
transport = SP_spam_transport

SP_no-spam_router:
driver = accept
domains = +local_domains
local_part_prefix = no-spam
transport = SP_no-spam_transport
# SpamProbe end #####

dnslookup:
driver = dnslookup
domains = ! +local_domains
transport = remote_smtp
ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8
no_more

# SpamProbe start #####
SP_check_router:
driver = accept
domains = +local_domains
local_parts = lsearch;/usr/local/etc/spamprobe/users
condition = ${if and {match{$h X-SpamProbe:}{SPAM}}
{!match address{$sender address}}
{lsearch;/usr/local/etc/spamprobe/whitelist}}}}
transport = SP_check_transport
no_more
# SpamProbe end #####
```

Здесь очень важно соблюсти последовательность при добавлении новых роутеров.

Пожалуй, стоит разъяснить некоторые строки кода.

Роутеры SP_spam_router и SP_no-spam_router необходимы для обучения нашего спам-фильтра. «Не ошибается тот, кто ничего не делает» – гласит пословица, так и в нашем случае. Если фильтр допустил ошибку и вместо того, чтобы пометить письмо как SPAM, пометил его GOOD, перешлите это письмо как вложение на адрес spam@(ваш домен), и вы увидите, что в следующий раз это письмо будет промаркировано правильно. Аналогично поступаем в обратной ситуации, но пересылать письмо теперь будем на адрес no-spam@(ваш домен).

Роутер SP_check_router собственно выполняет проверку и дает оценку письму путем добавления в тело заголовка X-SpamProbe:

```
local_parts = lsearch;/usr/local/etc/spamprobe/users
```

Определяет список пользователей, для которых будет работать спам-фильтр. В моем случае не требовалось фильтровать письма абсолютно всех пользователей, это обуславливалось личными предпочтениями каждого. Поэтому было принято решение создать специальный файл, содержащий имена тех пользователей, кому это действительно было необходимо.

Синтаксис файла users такой:

```
# Список пользователей, письма которых анализируются
# антиспамом
admin
maxim
hostmaster
oleg
gl_sten
```

Примечание: никаких регулярных выражений, только точное соответствие.

Теперь по поводу строки:

```
condition = ${if and {{match{$h X-SpamProbe:}{SPAM}} }
  {!match address{$sender address} }
  {lsearch;/usr/local/etc/spamprobe/whitelist}}}
```

с первым условием, я думаю, все понятно, а второе условие – это своеобразная попытка сделать белый список адресов, почта с которых априори не является спамом. Синтаксис файла whitelist аналогичен синтаксису файла users:

```
# Список адресов, которым можно доверять,
# и не анализировать почту, приходящую от них
murzilka@ukr.net
meren@mail.ru
office@predpriyatie.com
info@rambler.ku
```

А в секции TRANSPORTS CONFIGURATION непосредственно сам транспорт:

```
#####
# TRANSPORTS CONFIGURATION #
# SpamProbe start #####
SP_check transport:
  driver = pipe
  command = "/usr/local/libexec/dovecot/deliver -d $local_part@$domain -m INBOX.Spam"
  user = mailnull
  group = mail
```

```
log_output = true
return_path_add
# headers_remove = X-SpamProbe

SP_spam_transport:
  driver = pipe
  command = "/usr/local/bin/spamprobe -d /var/lib/spamprobe spam"
  return_path_add = false
  return_fail_output = true
  log_output = true
  user = mailnull
  group = mail

SP_no-spam_transport:
  driver = pipe
  command = "/usr/local/bin/spamprobe -d /var/lib/spamprobe good"
  return_path_add = false
  return_fail_output = true
  log_output = true
  user = mailnull
  group = mail
# SpamProbe end #####
```

В этом разделе последовательность при добавлении не имеет такого значения, как в разделе ROUTERS CONFIGURATION.

Как мы можем наблюдать, письма, отмеченные как SPAM, отправляются в папку .INBOX.Spam. В моем случае с помощью программы deliver из пакета dovecot. Причем у каждого пользователя существует своя папка .INBOX.Spam, поэтому в случае ошибки спам-фильтра пользователь всегда может заглянуть в нее и поискать пропавшее письмо.

Хочу отметить, что мною приведен пример, который является базовым, но ни в коем случае не окончательным. Если у вас есть желание, то не поленитесь, прочитайте документацию по Exim (<http://www.exim.org>) (перевод можно найти по адресу <http://www.lissyara.su/?id=1200>). Вам вполне по силам будет добавить, а может, и улучшить вышеописанный способ фильтрации спама.

Ну и напоследок несколько полезных скриптов:

Первый – для обучения spamprobe в ручном режиме:

```
#!/bin/bash


spamprobe -d /var/lib/spamprobe spam ./spam/*
spamprobe -d /var/lib/spamprobe good ./nospam/*
```

Этот скрипт вам, возможно, понадобится на начальном этапе работы спам-фильтра.

И второй для cron:

```
0 3 * * * root find /var/mail/*/*/.INBOX.Spam/new/* -l
  -ctime +30 -exec rm -f {} \;
```

Этот скрипт проверяет все папки .INBOX.Spam у всех пользователей на наличие в них писем старше 30 дней и удаляет их.

Вот, пожалуй, и все. 

P.S.: Хочу выразить благодарность:

- Михаилу Позднякову за статью «Фильтрация СПА-Ма с помощью SpamProbe» – <http://www.ru-clc.org/node/102>.
- Виталию Захарову за статью «DSPAM extension to exim» – <http://www.lissyara.su/?id=1301>.
- Алексею Кеда за проект www.lissyara.su.