IntegratedSpamdInPostfix

Integrating SpamAssassin into Postfix using spamd

To integrate SpamAssassin into Postfix, you'll need to pipe incoming e-mail through a script or program that passes the e-mail to SpamAssassin for rewriting, and then either chooses to send it on or discard it (you may wish to discard it, for example, if SpamAssassin reports a very high spam score).

Below are some examples of how to do this using your own script to feed e-mail to SpamAssassin. Aside from the methods below, you can look in the examples folder included with SpamAssassin (located, for example, at /usr/share/doc/spamassassin/examples on Debian installs by default) for other options, like procmail. Detailed information on setting up mail filtering (upon which the examples below are based) can be found in the Postfix FILTER_README documentation.

Note that it is generally good practice to configure Postfix NOT to bounce undeliverable messages, as this can cause backscatter. Either have a 'catch-all' address that all e-mail to invalid recipient addresses goes to, or reject e-mail to invalid recipient addresses. You could also silently discard e-mail to invalid recipient addresses, although this is a bit hostile to legitimate users trying to send you e-mail. Never bounce e-mail detected as spam; either rewrite it and deliver it, reject it during the SMTP connection, or silently discard it.

bash script to rewrite spam method

This is a simple method to integrate SpamAssassin. All spam e-mail will still come through to your address(es), but it will be clearly marked as spam.

1) Create a bash script to receive e-mail from Postfix and pipe it to SpamAssassin for rewriting. Then forward the rewritten version to Postfix's sendmail implementation:

{{{#!highlight bash
#!/bin/bash
#
spamfilter.sh
#
Simple filter to plug SpamAssassin into the Postfix MTA
#
Modified by Jeremy Morton
#
This script should probably live at /usr/bin/spamfilter.sh
... and have 'chown root:root' and 'chmod 755' applied to it.
#
For use with:
Postfix 20010228 or later
SpamAssassin 2.42 or later

Note: Modify the file locations to suit your particular
server and installation of SpamAssassin.
File locations:
(CHANGE AS REQUIRED TO SUIT YOUR SERVER) SENDMAIL=/usr/sbin/sendmail SPAMASSASSIN=/usr/bin/spamc

logger <<<"Spam filter piping to SpamAssassin, then to: $SENDMAIL $@"
${SPAMASSASSIN} | ${SENDMAIL} "$@"

exit $?
}}}
This, of course, assumes you're using the spamd/spamc implementation of SpamAssassin to improve performance - spamc is the command you'll want to invoke to check an e-mail (by setting SPAMASSASSIN to its location).

2) Ensure the newly-created /usr/bin/spamfilter.sh has correct permissions (0755), and is owned by root:root.

3) Modify the /etc/postfix/master.cf file; first, change the first 'smtp' line of the file to:

```
smtp      inet  n       -       -       -       -       smtpd -o content_filter=spamfilter
```

Then, add the following (a call to our newly-created spamfilter script) at the end:

```
spamfilter
          unix  -       n       n       -       -       pipe
   flags=Rq user=spamd argv=/usr/bin/spamfilter.sh -oi -f ${sender} ${recipient}
```

This setup assumes you have a 'spamd' user for the script to be run as. If you wish to run it as a different user, modify the 'user' argument above.

4) Restart the Postfix service.

Now, Postfix should be sending every incoming e-mail through the spamfilter.sh script (which means it's going through SpamAssassin), with the e-mail headers (and maybe the rest of the e-mail, if it's spam) being rewritten before being delivered to the target mailbox. Optionally, you may wish to tweak SpamAssassin's configuration to cause it to rewrite the e-mails in a more useful way (only rewrite spam mail subject but not whole e-mail, add a 'ham' header to give detailed info on even non-spam, etc.)

5) (optional) Tweak SpamAssasin's configuration. Ensure that the below is in /etc/mail/spamassassin/local.cf and then restart SpamAssassin:

```
# Rewrite the subject of suspected spam e-mails
#rewrite_header Subject *****SPAM*****
rewrite_header Subject *****SPAM***** (_SCORE_)
# Just add an X-Spam-Report header to suspected spam, rather than rewriting the content of the e-mail
report_safe 0
# Also we want to add a detailed ham report header to even e-mail that ISN'T suspected to be spam
add_header ham HAM-Report _REPORT_
# Set the threshold at which a message is considered spam (3 is usually sufficient)
required_score 3.0
```

advanced bash script to discard 'likely spam' method

NOTE: _At the time of writing, the following script will only work with a patched version of spamc from SVN trunk (which can be found at http://svn.apache.org/repos/asf/spamassassin/trunk). It needs to be patched with the 'Second patch' from the following Bugzilla bug:

https://issues.apache.org/SpamAssassin/show_bug.cgi?id=6890

It introduces a -T (override threshold) parameter which is necessary for this shell script's operation. I'm hoping it will be merged into SpamAssassin trunk one day._

This is a more advanced method to integrate SpamAssassin. The script will set a threshold, and only messages with a score above that threshold will be discarded. Messages between the normal configured SpamAssassin threshold and the threshold set in the script will be considered "grey area" messages, and whilst they will be rewritten as spam by Spamassassin, they will still be delivered to the mailbox. Messages below the normal configured SpamAssassin threshold will be delivered as normal.

1) Create a bash script to receive e-mail from Postfix and pipe it to SpamAssassin for rewriting. Check whether SpamAssassin tells us whether the message is over the threshold or not; if so, discard the message. Otherwise, deliver it through Postfix's sendmail implementation:

{{{#!highlight bash
#!/bin/bash
#
spamfilter.sh
#
Advanced filter to plug SpamAssassin into the Postfix MTA
Only discards messages that are above the threshold that is
passed in to spamc; other spam messages that are below this
threshold will be rewritten, but not discarded.
#
Modified by Jeremy Morton
#
This script should probably live at /usr/bin/spamfilter.sh
... and have 'chown root:root' and 'chmod 755' applied to it.
#
For use with:
Postfix 20010228 or later
??? SpamAssassin 2.42 or later ??? - should change to the latest version when/if spamc patch gets applied to trunk

Note: Modify the file locations to suit your particular
server and installation of SpamAssassin.
File locations and settings:
(CHANGE AS REQUIRED TO SUIT YOUR SERVER) THRESHOLD_OFFSET=4.0
^ Remember, we're using the threshold offset build, not the absolute value one. Passing in a -T value of 4.0
will give us an absolute threshold of maybe (base) 3.0 + 4.0 = 7.0. MAX_MESSAGE_SIZE=10485760
^ Allow e-mails up to size 10485760 (10MB) - they're not too intensive to process, and any e-mails above that
size don't get processed, and are assumed to be non-spam. Clearly it's important that we process as many
messages as possible, and there should be very few messages indeed that exceed 10MB size. SENDMAIL=/usr/sbin/sendmail SPAMASSASSIN=/usr/bin/spamc

# We use -4 to force IPv4 because alas at the time of writing, spamd doesn't listen out for connections on IPv6... logger -s -p mail.notice -t spamfilter <<<" Spam filter piping to SpamAssassin: $SPAMASSASSIN -4 -x -E -s $MAX_MESSAGE_SIZE -T $THRESHOLD_OFFSET" CAUGHT_OUTPUT=${SPAMASSASSIN} -4 -x -E -s $MAX_MESSAGE_SIZE -T $THRESHOLD_OFFSET
SPAMASSASSIN_EXITCODE=$?

if ["$SPAMASSASSIN_EXITCODE" -gt 1]; then
logger -s -p mail.warning -t spamfilter <<<"Error code $SPAMASSASSIN_EXITCODE processing spam!"
exit $SPAMASSASSIN_EXITCODE elif ["$SPAMASSASSIN_EXITCODE" -eq 1]; then
logger -s -p mail.notice -t spamfilter <<<"SpamAssassin says message is likely spam; discarding."
exit 0 fi

Message is OK, or grey-area and has been rewritten. Deliver to mailbox. logger -s -p mail.notice -t spamfilter <<<"OK. Piping to sendmail: $SENDMAIL $@"
${SENDMAIL} "$@" <<<"$CAUGHT_OUTPUT" exit $?
}}}
This, of course, assumes you're using the spamd/spamc implementation of SpamAssassin to improve performance - spamc is the command you'll want to invoke to check an e-mail (by setting SPAMASSASSIN to its location).

2) Ensure the newly-created /usr/bin/spamfilter.sh has correct permissions (0755), and is owned by root:root.

3) Modify the /etc/postfix/master.cf file; first, change the first 'smtp' line of the file to:

```
smtp      inet  n       -       -       -       -       smtpd -o content_filter=spamfilter
```

Then, add the following (a call to our newly-created spamfilter script) at the end:

```
spamfilter
          unix  -       n       n       -       -       pipe
   flags=Rq user=spamd argv=/usr/bin/spamfilter.sh -oi -f ${sender} ${recipient}
```

This setup assumes you have a 'spamd' user for the script to be run as. If you wish to run it as a different user, modify the 'user' argument above.

4) Restart the Postfix service.

Now, Postfix should be sending every incoming e-mail through the spamfilter.sh script (which means it's going through SpamAssassin), with the e-mail headers (and maybe the rest of the e-mail, if it's spam) being rewritten before being delivered to the target mailbox. Optionally, you may wish to tweak SpamAssassin's configuration to cause it to rewrite the e-mails in a more useful way (only rewrite spam mail subject but not whole e-mail, add a 'ham' header to give detailed info on even non-spam, etc.)

5) (optional) Tweak SpamAssasin's configuration. Ensure that the below is in /etc/mail/spamassassin/local.cf and then restart SpamAssassin:

```
# Rewrite the subject of suspected spam e-mails
#rewrite_header Subject *****SPAM*****
rewrite_header Subject *****SPAM***** (_SCORE_)
# Just add an X-Spam-Report header to suspected spam, rather than rewriting the content of the e-mail
report_safe 0
# Also we want to add a detailed ham report header to even e-mail that ISN'T suspected to be spam
add_header ham HAM-Report _REPORT_
# Set the threshold at which a message is considered spam (3 is usually sufficient)
required_score 3.0
```

Old alternative method

A major problem with the old method filter (below) is that Postfix attempts to bounce messages flagged as spam. This is a waste of resources as most spams have fake return addresses. Furthermore, if they fake a return address that doesn't belong to them, you may end up bouncing the message to an innocent 3rd party. Enough of this can result in YOU being on an RBL.

_ **Not true.** (1) Postfix only bounces messages if you tell it to. I currently use the above filter, and I use a sieve rule to filter `X-Spam-Flag: YES` into a folder in the same account which I occasionally review. Many people sort mailing lists into separate folders, and this can typically be done with the same mechanism. It also makes it more practical to actually check it every now and then... all spam systems have false positives, so silently blackholing likely spam (potential ham) is far more rude than bouncing it. (2) If you use a proper bounce message (as opposed to those stupid virus filter notifications), there are ways for the innocent party to filter out bounces from messages they didn't send. When I get some time, I'll write one up and link to it. -Slamb_

Instead, we'd like to create a blackhole for spam.

To accomplish this while still using an after-queue filter, the spam may be redirected to another account on the system. Steps to do this are as follows:

1) Modify /etc/postfix/master.cf with:

```
smtp      inet  n       -       n       -       -       smtpd -o content_filter=spamassassin

... and then at the end, add:

spamassassin
          unix  -       n       n       -       -       pipe
   flags=Rq user=nobody argv=/path/to/filter.sh -oi -f ${sender} ${recipient}
```

2) Create filter.sh as follows:

{{{#!highlight sh
#!/bin/sh

filter.sh
#
This script redirects mail flagged as spam to a separate account
You must first create a user account named "spamvac" to hold the flagged mail

SENDMAIL="/usr/sbin/sendmail -i" SPAMASSASSIN=/usr/bin/spamc COMMAND="$SENDMAIL $@"
#If your SQL preferences set to "user" USER=`echo $COMMAND | awk '{ print $NF }' | sed 's/@.*$//'`
#If your SQL preferences set to "user@domain"
#USER=`echo $COMMAND | awk '{ print $NF }'`

NEW_COMMAND=`echo $COMMAND | awk '{ $6 = "spamvac"; NF = 6; print }'`

# Exit codes from <sysexits.h>
EX_TEMPFAIL=75 EX_UNAVAILABLE=69

umask 077

OUTPUT="`mktemp /tmp/mailfilter.XXXXXXXXXX`"

if ["$?" != 0]; then
/usr/bin/logger -s -p mail.warning -t filter "Unable to create temporary file."
exit $EX_TEMPFAIL fi

Clean up when done or when aborting. trap "rm -f $OUTPUT" EXIT TERM

$SPAMASSASSIN -x -E -u $USER > $OUTPUT return="$?" if ["$return" = 1]; then
$NEW_COMMAND < $OUTPUT
exit $? elif ["$return" != 0]; then
/usr/bin/logger -s -p mail.warning -t filter "Temporary SpamAssassin failure (spamc returned $return)"
exit $EX_TEMPFAIL fi

$SENDMAIL "$@" < $OUTPUT exit $?
}}}
This causes incoming smtp mail to be checked for spam. If the mail's spam score exceeds the threshold (set in local.cf, or in ~/.spamassassin/user_prefs) then `spamc -E` returns 1 and the mail is redirected to the spamvac account. Otherwise it is passed on to the intended recipient. Bounces no longer occur except on a true error condition.

Old method (original)

The easiest way to integrate postfix and spamassassin is to use spamd in an after-queue inspection. This configuration does not allow rejecting messages within the SMTP transaction, so it unfortunately contributes to backscatter email. On the other hand, it has important performance advantages over before-queue inspection.

First, edit /etc/postfix/master.cf, find the

```
# ==========================================================================
# service type  private unpriv  chroot  wakeup  maxproc command + args
#               (yes)   (yes)   (yes)   (never) (50)
# ==========================================================================
...
smtp      inet  n       -       n       -       -       smtpd
...
```

line and just add " -o content_filter=spamassassin" to the end of the line:

```
# ==========================================================================
# service type  private unpriv  chroot  wakeup  maxproc command + args
#               (yes)   (yes)   (yes)   (never) (50)
# ==========================================================================
...
smtp      inet  n       -       n       -       -       smtpd -o content_filter=spamassassin
...
```

Then, go to the end of the file, and add this:

```
# ==========================================================================
# service type  private unpriv  chroot  wakeup  maxproc command + args
#               (yes)   (yes)   (yes)   (never) (50)
# ==========================================================================
...
spamassassin
          unix  -       n       n       -       -       pipe
   user=nobody argv=/path/to/spamc -e /path/to/postfix/sendmail -oi -f ${sender} ${recipient}
# make sure it's all on one line or
# start the consecutive lines with a whitespace
# like I did here
```

Make sure that you have adjusted the path to the spamc and sendmail commands above! (Please note that the path required is Postfix's sendmail and not the standalone package, Sendmail the SMTP server. It will not work if you're not careful about which one is installed). Then, setup spamd to start with the system, and you are ready to go. If you wish to provide spamassassin preferences, change "user=nobody" to a valid system user (except for root, since Postfix will NOT call external programs as root), and add .spamassassin into that user's home directory.

If you want spamd to setuid to the user so it can save settings and leaning data you need to tell it the user

```
# ==========================================================================
# service type  private unpriv  chroot  wakeup  maxproc command + args
#               (yes)   (yes)   (yes)   (never) (50)
# ==========================================================================
...
spamassassin
          unix  -       n       n       -       -       pipe
   user=nobody argv=/path/to/spamc -u ${user} -e /path/to/postfix/sendmail -oi -f ${sender} ${recipient}
```

If you use user preferences stored in SQL, you should change "spamassassin" service in master.cf to following:

```
# ==========================================================================
# service type  private unpriv  chroot  wakeup  maxproc command + args
#               (yes)   (yes)   (yes)   (never) (50)
# ==========================================================================
...
spamassassin
          unix  -       n       n       -       -       pipe
   flags=Rq user=nobody argv=/path/to/spamc -u ${user} -e /path/to/postfix/sendmail -oi -f ${sender}
${recipient}
```

Notice "-u ${recipient}" added. Otherwise "username" field in database will always appear as user which postfix is invoking spamc(in this example it is 'nobody'). Do not forget also to set spamassassin_destination_recipient_limit = 1 in main.cf . spamc doesn't expect to parse multiple recipients at once.

Note that this exact method of invoking SpamAssassin is not very robust. Postfix's pipe error behavior is as follows:

- If the `exec()` fails (e.g., `/path/to/spamc` is incorrect), Postfix will bounce the message.
- If the command returns `EX_OSERR` or `EX_TEMPFAIL` (defined in `/usr/include/sysexits.h`), Postfix will defer the message.
- If the command returns `EX_OK`, Postfix will consider the message done. The command should deliver it; it will be discarded otherwise.
- If the command returns any other error code, Postfix will bounce the message.

With `spamc -e`, spamc will return `EX_OK` when spamd is unavailable but sendmail is working, so Postfix will deliver the message. You probably want it to be deferred instead. This is not possible with spamc directly, but you can do so with a simple script. Replace the lines above with:

```
spamassassin
          unix  -       n       n       -       -       pipe
   flags=Rq user=nobody argv=/path/to/filter.sh -oi -f ${sender} ${recipient}
```

and create a `filter.sh` as follows:

{{{#!highlight bash
#!/bin/bash SENDMAIL="/usr/sbin/sendmail -i" SPAMASSASSIN=/usr/bin/spamc

# Exit codes from <sysexits.h>
EX_TEMPFAIL=75 EX_UNAVAILABLE=69

umask 077

OUTPUT="`mktemp -p /tmp mailfilter.XXXXXXXXXX`" if ["$?" != 0]; then
/usr/bin/logger -s -p mail.warning -t filter \
"Unable to create temporary file."
exit $EX_TEMPFAIL fi

Clean up when done or when aborting. trap "rm -f $OUTPUT" EXIT TERM

$SPAMASSASSIN -x > $OUTPUT return="$?" if ["$return" = 1]; then
echo "Message content rejected"
exit $EX_UNAVAILABLE elif ["$return" != 0]; then
/usr/bin/logger -s -p mail.warning -t filter \
"Temporary SpamAssassin failure (spamc returned $return)"
exit $EX_TEMPFAIL fi

$SENDMAIL "$@" < $OUTPUT exit $?
}}}
You still must be careful that the script is in the right place and executable or email will bounce. However, this will defer messages on most failures.

A more complex solution is the one proposed in the FILTER_README file that comes with the Postfix distribution. See IntegratePostfixViaSpampd for one approach.

Only mail received by SMTP will be scanned with this method, i.e. mail injected with sendmail(1) will not be fed to SpamAssassin (if it were, it too would be fed to the filter script, which would send the e-mail via sendmail, etc. and there would be an infinite loop).