Filtering malware and spam with Postfix

http://advosys.ca/papers/postfix−filtering.html
Jan 05 2003

Introduction

Postfix (http://www.postfix.org/) is the popular Sendmail replacement that is fast, secure, and easy to administer.

Besides being an easy replacement for Sendmail, Postfix makes an outstanding secure mail gateway or "mail firewall", shielding fragile Intranet mailers such as MS Exchange and Lotus Notes from direct contact with the Internet.

Postfix can filter mail through external programs, similar to what Sendmail can do with its "milter" extension. Using the appropriate external filters, Postfix can remove malicious content such as viruses and MS Outlook exploits, and identify spam.

This paper demonstrates how to use the filtering capabilities of Postfix to perform high−level content filtering including:

- Dropping hostile attachments such as .EXE and .VBS files
- Preventing MS Outlook from auto−executing unknown attachments
- Preventing multiple MIME and HTML exploits
- Disabling "web bugs" that spammers use to track recipients
- Identifying spam based on message content

The method shown here is multipurpose: it is suitable for local mail (when the final destination is an mbox or maildir on the Postfix server itself) and when Postfix is used as a "mail firewall" to relay mail to an internal mail system. This method also has the advantage of applying one spam and content filtering policy for all users. Users can also share one auto−whitelist database, if you use that feature in SpamAssassin.

However, if you require unique settings for each user, it may be better to filter mail via a mail delivery agent such as procmail instead of using the method shown here.

The software

We extend the capabilities of Postfix using two best−of−breed open source programs that perform content filtering and spam identification:

Anomy Sanitizer

Anomy Sanitizer (http://mailtools.anomy.net/) is not a virus scanner (though a virus scanner can be used with it). Instead, it removes the mail security problems that e−mail virus scanners do not: MIME exploits, dangerous HTML tags, and other problems that have made MS Outlook so infamous.

Anomy Sanitizer is written in Perl. It filters SMTP messages checking for most known exploits and hostile file attachments. It can remove attachments, rename unknown file types, "defang" exploitable HTML tags, fix MIME headers and perform other essential mail security tasks.

SpamAssassin

SpamAssassin (http://spamassassin.org/) is a mail filter that attempts to identify junk e−mail ("spam"). It examines the content of mail messages looking for key phrases and other identifiers common to most spam.

The approach is more sophisticated than the simple keyword matching provided by most SMTP anti−virus software. SpamAssassin uses a scoring system: messages are tagged as spam only when they have enough spam characteristics in total. This in combination with other features results in very few false positives. In our experience, a properly managed SpamAssassin installation correctly identifies 90% to 95% of spam with less than 1% false positives.

SpamAssassin doesn't block spam. Instead, it tags messages as *probable* spam by changing the Subject line and message headers. This is very wise: no automated system can recognize spam with 100% certainty... deciding "what is spam" is a judgment call. All automated spam filters will produce some false positives (wanted e−mail mistakenly tagged as spam) and false negatives (spam not identified as such).

SpamAssassin identifies probable spam e−mail, but leaves the choice of what to do with it up to you. Most organizations instruct their users how to add rules to their e−mail software to delete identified messages or, better yet, move them to a folder for later review. Others choose to use an additional filter such as procmail to sideline suspect messages in a holding area for further review by an administrator.

In our opinion, tagging messages then allowing users to decide is the best choice. It gets around certain legal issues organizations like ISPs and government departments face when filtering mail: allowing each user to decide what to do with spam, instead of deleting it at the gateway as corporate policy, is less open to legal challenge and complaints from users.

> If you insist on automatically deleting suspected spam, a good compromise is to use the scoring feature of SpamAssassin and a custom filter. Pass low scoring messages to the end user, but delete very high scoring ones in Postfix. A message with a SpamAssassin score over 15 is unlikely to be legitimate mail.

The method shown here tags suspected spam only. No automated deletion is performed, but the filter script can be changed without too much effort to sideline or delete suspected spam if that's what you want.

Preparing the server

Prerequisites

We assume you have a Unix or Unix−flavour server such as Linux or BSD. The method shown here has been used on Solaris 8, Debian Linux 3.0, RedHat Linux 7.x and many other unix−like operating systems.

You need Postfix version 1.1 or higher. Type "postconf | grep version" at the Unix command line to see the version number. We have not yet tested this method on Postfix 2.x.

> **NOTE:** The server *must* have Postfix already installed and accepting mail the way you want it: please do not try to follow these instructions and also get Postfix running at the same time: first get Postfix working to your liking, then come back and follow these instructions.

You need Anomy Sanitizer revision 1.49 or later. The revision number is part of the download file name (eg. "anomy−sanitizer−1.49.tar.gz") and is also in the CHANGES file included in the Anomy tar file.

Also, you need SpamAssassin version 2.42 or later. Type "spamassassin −V" at the command line to see the version number. Please check that you have this or a later version of SpamAssassin before proceeding.

> **Packaged versions:** Some Linux distributions include packaged versions of SpamAssassin and Anomy Sanitizer as RPM or DEB files. Often, the packaged versions are out of date. Also they often move files the default locations and make other changes. To avoid problems, we recommend installing SpamAssassin and Anomy Sanitizer manually as described below instead of using packaged versions. Most packaged versions of Postfix, however, work well.

Creating a filter account

To use external filtering with Postfix, create a Unix group on the server named "filter".

Next, create a user account named "filter" on the server and make it a member of group "filter". This will be a least−privileged account used by the scripts.

No other user should belong to group "filter". Logins for the "filter" account should be disabled (eg. 'passwd −l filter' on Linux and Solaris) and the shell in /etc/passwd should be set to an invalid shell such as /bin/false.

Installing required Perl modules

Perl 5.005_03 or higher must be available on the server. Perl 5.6 or higher is recommended. Also, download and install the following Perl modules from CPAN (http://search.cpan.org/):

- MIME::Base64
- MIME::QuotedPrint
- Mail::Audit (and all prerequisite modules needed by Mail::Audit)

If you've set up the CPAN module in your Perl installation, the easiest way to add these modules is to use CPAN to download, build and install automatically:

```
perl −MCPAN −e shell
o conf prerequisites_policy ask
install MIME::Base64
install MIME::QuotedPrint
install Mail::Audit
quit
```

Installing Anomy Sanitizer

Anomy Sanitizer is installed simply by unpacking the tar file into a suitable directory on your mail server. We recommend /usr/local/anomy but any other directory will work. Read the instructions in file sanitizer.html included with Anomy.

Anomy configuration

In directory /usr/local/anomy, create a file named anomy.conf and add your anomy configuration rules. What specific settings you choose depends on your particular e−mail policies and the type of internal e−mail software you are protecting. The "real world configuration" shown in the Anomy documentation is a good starting point.

Below is a configuration we've used on mail gateways that protect MS Exchange and MS Outlook users. It defangs HTML and MIME exploits, plus drops all executable attachments (we strongly recommend dropping

executable attachments as a general policy: see our paper "E−mail policies that prevent viruses" at http://advosys.ca/papers/mail−policies.html).

```
# Example configuration file for Anomy Sanitizer
#
# From http://advosys.ca/papers/postfix−filtering.html
# Advosys Consulting Inc., Ottawa
#
# Works with Anomy Sanitizer revision 1.53

# Do not log to STDERR:
feat_log_stderr = 0

# Don't insert log in the message itself:
feat_log_inline = 0

# Advertisement to insert in each mail header:
header_info = X−Sanitizer: Advosys mail filter
header_url = 0
header_rev = 0

# Enable filename based policy decisions:
feat_files = 1

# Protect against buffer overflows and null values:
feat_lengths = 1

# Replace MIME boundaries with our own:
feat_boundaries = 1

# Fix invalid and ambiguous MIME boundaries, if possible:
feat_fixmime = 1

# Trust signed and/or encrypted messages:
feat_trust_pgp = 1
msg_pgp_warning = WARNING: Unsanitized content follows.\n

# Defang shell scripts:
feat_scripts = 0

# Defang active HTML:
feat_html = 1

# Defang UUEncoded files:
feat_uuencoded = 0

# Sanitize forwarded content too:
feat_forwards = 1

# Testing? Set to 1 for testing, 0 for production:
feat_testing = 0

# # Warn user about unscanned parts, etc.
feat_verbose = 1

# Force all parts (except text/html parts) to
# have file names.
feat_force_name = 1

# Disable web bugs:
feat_webbugs = 1

# Disable "score" based mail discarding:
```

```
score_panic = 0
score_bad = 0

msg_file_drop  = \n*****\n
msg_file_drop += NOTE: An attachment named %FILENAME was deleted from
msg_file_drop += this message because it contained a windows executable
msg_file_drop += or other potentially dangerous file type.
msg_file_drop += Contact the system administrator for more information.

##
## File attachment name mangling rules:
##

# Specify the Anomy temp file and quarantine directory
file_name_tpl       = /var/spool/filter/att-$F-$T.$$

# Number of rulesets we are defining:
file_list_rules = 2
file_default_policy = defang

# Delete probably nasty attachments:
file_list_1 = (?i)(winmail.dat)|
file_list_1 += (\.(exe|com|vb[se]|dll|ocx|cmd|bat|pif|lnk|hlp|ms[ip]|reg|sct|inf
file_list_1 += |asd|cab|sh[sb]|scr|cpl|chm|ws[fhc]|hta|vcd|vcf|eml|nws))$
file_list_1_policy = drop
file_list_1_scanner = 0

# Allow known "safe" file types and those that will be
# scanned by the user's desktop virus scanner:
file_list_2 = (?i)\.
#  Word processor and document formats:
file_list_2 += (doc|dot|txt|rtf|pdf|ps|htm|[sp]?html?
#  Spreadsheets:
file_list_2 += |xls|xlw|xlt|csv|wk[1-4]
#  Presentation applications:
file_list_2 += |ppt|pps|pot
#  Bitmap graphic files:
file_list_2 += |jpe?g|gif|png|tiff?|bmp|psd|pcx
#  Vector graphics and diagramming:
file_list_2 += |vsd|drw|cdr|swf
#  Multimedia:
file_list_2 += |mp3|avi|mpe?g|mov|ram?|mid|ogg
#  Archives:
file_list_2 += |zip|g?z|rar|tgz|bz2|tar
#  Source code:
file_list_2 += |[ch](pp|\+\+)?|s|inc|asm|patch|java|php\d?|jsp|bas)
file_list_2_policy = accept
file_list_2_scanner = 0

# Any file type not listed above gets renamed to prevent
# ms outlook from auto-executing it.
```

Once the config file is in place, change ownership of the entire anomy directory tree to owner "root", group "filter". For example:

```
chown -R root:filter /usr/local/anomy
```

Change permissions on the anomy directory tree so it is not world readable. For example:

```
chmod 0750 /usr/local/anomy
```

Installing SpamAssassin

The SpamAssassin documentation describes how to install and configure that software. As described above, if you have the CPAN module set up on your server, it can download and install SpamAssassin automatically. For example:

```
perl −MCPAN −e shell
o conf prerequisites_policy ask
install Mail::SpamAssassin
quit
```

SpamAssassin configuration

SpamAssassin is installed "out of the box" with a good set of default identification rules. It also lets you specify your own settings in /etc/mail/spamassassin/local.cf.

It's a good idea to leave local.cf alone at first, until you get things working and have a feel for how the default rules identify the spam you receive. Later on you can fine tune SpamAssassin settings in local.cf as required.

However, we recommend making one change to local.cf right away:

1. Whitelist well−known senders so their mail will never be identified as spam.

Manual whitelists

You should whitelist the e−mail addresses of well−known legitimate senders to avoid the chance of them being mis−identified by the SpamAssassin default rules. Add "whitelist_from" settings to /etc/mail/spamassassin/local.cf listing important clients, mailing lists and other known spam free sources. For example:

```
whitelist_from   director_8345@hotmail.com   # whitelist one specific sender
whitelist_from   @advosys.ca                 # whitelist an entire domain
whitelist_from   @securityfocus.com
```

Configuring filtering in Postfix

You must have Postfix installed and accepting mail the way you want. To preserve your sanity, do not attempt to add filtering *and* get a first−time installation of Postfix working at the same time. Install and configure Postfix first, and verify it is working the way you want (accepting mail for your domain(s), relaying to and from internal servers, and so on). Getting Postfix running the first time is complex enough without trying to add filtering at the same time.

If you installed Postfix from source, the file `/README_FILES/FILTER_README` in the Postfix source tar file describes ways to add external content filtering to Postfix. Here we use the "Simple content filtering" method described in that file: plug a shell script into Postfix by modifying the `master.cf` file.

Creating the filter script

The following is a variation of the example filter script provided with Postfix. It simply pipes each message sent to it through Anomy Sanitizer, then through SpamAssassin. Postfix then picks up the result and re−injects it for final delivery.

Note: Modify the file locations specified in this file to match your particular server and installation of SpamAssassin.

```
#!/bin/sh
#
# filter.sh
#
# Simple filter to plug Anomy Sanitizer and SpamAssassin
# into the Postfix MTA
#
# From http://advosys.ca/papers/postfix-filtering.html
# Advosys Consulting Inc., Ottawa
#
# For use with:
#    Postfix 20010228 or later
#    Anomy Sanitizer revision 1.49 or later
#    SpamAssassin 2.42 or later
#
# Note: Modify the file locations to match your particular
#       server and installation of SpamAssassin.

# File locations:
# (CHANGE AS REQUIRED TO MATCH YOUR SERVER)
INSPECT_DIR=/var/spool/filter
SENDMAIL=/usr/lib/sendmail
ANOMY=/usr/local/anomy
SANITIZER=/usr/local/anomy/bin/sanitizer.pl
ANOMY_CONF=/usr/local/anomy/anomy.conf
SPAMASSASSIN=/usr/local/bin/spamassassin

export ANOMY

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

cd $INSPECT_DIR || { echo $INSPECT_DIR does not exist; exit $EX_TEMPFAIL; }

# Clean up when done or when aborting.
trap "rm -f in.$$; rm -f out.$$" 0 1 2 3 15

cat | $SPAMASSASSIN -x | $SANITIZER \
   $ANOMY_CONF 2>>/tmp/anomy.log > out.$$ || \
   { echo Message content rejected; exit $EX_UNAVAILABLE; }


$SENDMAIL "$@" < out.$$

exit $?
```

Place the filter.sh script into the same directory you installed Anomy Sanitizer (eg. /usr/local/anomy). The script should have the following permissions:

```
-rwxr-x---    owner=root  group=filter
```

Create a temporary directory

The script needs a directory to store temporary files. Create a directory that is writable by group "filter". For example:

```
mkdir /var/spool/filter
chown root:filter /var/spool/filter
chmod 0770 /var/spool/filter
```

The directory should be located on a partition large enough to store a few incoming mail messages. On most systems, the /var partition is the best choice. Whichever directory you choose, change the INSPECT_DIR setting in filter.sh to match.

Send a test message

Before plugging all this into Postfix, it would be wise to first try sending a few test messages from the command line.

As root, change to the /usr/local/anomy directory. Use your favorite text editor to create a file named "test.txt" containing the following:

```
From: <tester>
To: <you>

Hi there. This is a test message.
```

(The blank line between the "To:" and "Hi there" is required).

Send the above test message through the filter script as follows:

```
cat test.txt | ./filter.sh -f tester -- root
```

If things are working, no error messages should be printed and the shell prompt should return. Then in the mailbox for account "root" you should receive the test message.

Should error messages be output, such as "cannot execute..." or "cannot find...", double check what you just typed and try again. Depending on your system, the error messages should lead you to where the error is... in the anomy program, the filter.sh shell script, or file permissions. You might also find clues in the file /tmp/anomy.log. Resolve all the errors before proceeding... if you can't send mail manually using this method, Postfix probably won't be able to send any either.

Changing master.cf

Once you are sure you can send messages manually, you're ready to configure Postfix to use the filter.sh script as well.

Add the following line to the *bottom* of /etc/postfix/master.cf:

```
filter    unix  -   n   n   -   -   pipe
    user=filter argv=/usr/local/anomy/filter.sh -f ${sender} -- ${recipient}
```

(The above must all be on one single line.)

Next, change the line in master.cf that controls the Postfix smtp daemon as follows:

```
smtp   inet  n   -   n   -   -   smtpd -o content_filter=filter:
```

(Again, the above must all be on one single line.)

Save the changes to master.cf and use 'postfix reload' to force the postfix daemons to re−read the configuration files.

Testing it out

Postfix should now be filtering every message it receives through both Anomy Santizer and Spamassassin.

Test it by sending a few messages with MIME attachments through Postfix. The Postfix log file should now show each the message being received *twice*: once by the SMTP daemon with "relay=filter", then again by the command−line "sendmail" compatibility program.

Separating inbound mail from outbound

The configuration described here sends both inbound and outbound mail through the filters. This is usually not desirable: most organizations only want to filter spam and malware from inbound mail and leave outbound mail untouched.

Unfortunately, Postfix has no inherent ability to distinguish inbound and outbound mail. There a few convoluted ways to trick it into handling inbound mail separately from outbound, but they are complex and may not work in newer versions of Postfix.

We've found that the simplest and most reliable way to separate traffic is run two instances of Postfix: one for inbound mail, another for outbound. Each instance runs on the same server, but uses the Postfix `inet_interfaces` setting to receive mail on separate IP addresses: one IP for inbound mail, another for outbound.

Each Postfix instance has it's own configuration files and spool directories. The inbound mail instance is configured as described above to filter via Anomy Sanitizer and SpamAssassin. The outbound instance is configured as a more generic Postfix installation, with no filtering. We've used the two−instance method in large installations with great success. Fortunately, Postfix is reasonably light weight: running multiple instances on one server doesn't consume huge amounts of memory or disk resources.

Creating two instances of Postfix is straightforward: copy the `/etc/postfix` directory to some other directory (for example, copy `/etc/postfix` to `/etc/postfix−out`), create a separate spool directory for the new instance (for example, create `/var/spool/postfix−out` Each Postfix instance *must* have a separate queue directory). Edit the appropriate lines in `/etc/postfix−out/main.cf` to turn off filtering for outbound mail, bind to a different IP address and so on. Be sure to change the `queue_directory` setting to the location of the new queue directory.

The second Postfix instance shares the same binaries as the first instance so you don't have to recompile and install Postfix binaries a second time. To start the second instance, use the usual `postfix start` command except point to the directory containing the second instance configuration files. For example: `postfix −c /etc/postfix−out start`

A word about performance

The "simple content filtering" method is the easiest and most reliable way to filter messages with Postfix. However, performance suffers because each e−mail message has the overhead of invoking a shell, starting the Perl interpreter, and creating a temporary file.

The file creation overhead can be greatly reduced by mounting /var/spool/filter as a memory filesystem ("tmpfs" in Linux and Solaris). These filesystems are thousands of times faster than physical disk and are ideal for short−lived temp files.

Ideally the Anomy program should be able to exist as a daemon, allowing use of the Postfix "advanced" filtering example. SpamAssassin already provides a daemon option, most notably "spampd" from World Design Group (http://www.worlddesign.com/index.cfm/rd/mta/spampd.htm). Anomy Sanitizer doesn't provide a similar option, though we've found it is possible to plug it into spampd without too much trouble.

Even without tmpfs or spampd, the method described above has been able to filter upwards of 50,000 messages per day on one dual−processor Sun e250 server with 512MB memory, using two Postfix instances to separate inbound and outbound mail as described above. A Linux implementation on a single−cpu 750Mhz Pentium II platform with 512MB demonstrated similar performance levels.

Comments, suggestions, criticisms, additions to this document?
Please e−mail **papers (at) advosys.ca**

Copyright and terms of use

Use of this document

Permission to use this document from the Advosys Consulting web site is granted, provided that (1) This notice notice appears in all copies, (2) use is for informational and non−commercial or personal use only and will not be copied, reprinted, or posted on any network, computer or broadcast in any media, and (3) no modifications of the document are made.

Educational institutions (specifically K−12, universities and community colleges) may reproduce the Documents for distribution in the classroom, provided that (1) the below copyright notice appears on all copies, and (2) the original Uniform Resource Locator ("URL") of the document on the Advosys Consulting web site appears on all copies.

Use of this document for any other purpose requires written permission of Advosys Consulting Inc.

Limitation of liability

Advosys Consulting take no responsibility for the accuracy or validity of any claims or statements contained in the Documents and related graphics ("the content") on the Advosys web site. Further, Advosys Consulting Inc. makes no representations about the suitability of any of the information contained in the content for any purpose. All such documents, related graphics, products and services are provided "as is" and without warranties or conditions of any kind. In no event shall Advosys Consulting Inc. be liable for any damages whatsoever, including special, indirect or consequential damages, arising out of or in connection with the use or performance of information, products or services available on or through the Advosys Site.

Trademarks

Product, brand and company names and logos used on the Advosys web site site are the property of their respective owners.

About Advosys Consulting

Advosys Consulting is a privately held systems management corporation. Our global headquarters is in Ottawa, Ontario Canada. Formerly known as "Webber Technical Services", we have been providing systems management, computer engineering, security and consulting to private sector and government clients since 1991.

Areas of expertise

Advosys is a diversified consulting firm providing services in many areas of Information technology:

- Internet technologies
- Firewalls and information security
- Web applications
- Network architecture
- Unix and Linux systems management

Unbiased recommendations

Advosys Consulting is an *independent* consulting firm. We have broad experience with multiple vendors including Sun, Hewlett−Packard and Microsoft but are not a reseller of their hardware or software.

Unlike many consulting firms, Advosys receives no commissions, percentages, or other rewards from companies for promoting particular products or services.

This allows us the freedom to offer uncompromised objectivity. We have knowledge and experience with a broad range of products and technologies and can recommend solutions from any manufacturer, or open−source freeware if that is what best fits the requirements. Advosys works only for you, our client, not for a product manufacturer.

For more information, please visit us at http://advosys.ca