;login:

THE MAGAZINE OF USENIX & SAGE

August 2002 volume 27 • number 4

inside:

USENIX & SAGE

The Advanced Computing Systems Association &
The System Administrators Guild

ISPadmin

Stopping Spam: Part 1

by Robert Haskins

Robert Haskins is currently employed by WorldNET Internet Services, an ISP based in Norwood, MA. After many years of saying he wouldn't work for a telephone company, he is now affiliated with one.

rhaskins@usenix.org

Introduction

This edition of ISPadmin covers the methods used by ISPs to stop email spam on the server side. While the focus is on service providers, any enterprise can use the methods outlined below. The term "spam" will be used interchangeably with "unsolicited bulk email" (UBE) and "unsolicited commercial email" (UCE). If you ever wondered, the use of the word "spam" comes from episode 25 of *Monty Python's Flying Circus*, recorded on June 25, 1970, and broadcast sometime later in 1970.

Different people define spam in different ways. To be sure, the individuals and corporations who generate UCE do not consider their communications to be spam. On the other side of the spectrum, some recipients view any message that is commercial in nature which goes to more than two people to be UBE. Here are some attributes which may cause a message to be considered spam:

- Commercial content
- Large recipient list and/or use of BCC
- Concealing or forging message headers
- Numerous messages sent in a short period of time
- Use of recipient addresses without the owners' explicit approval
- Use of an open mail relay to send messages
- Invalid/Unresolvable To or From address header(s)

How Big Is the Problem?

It is difficult to come up with exact figures, but a reasonable estimate is 30% of all email messages could be considered spam. Some of the methods outlined below will catch 50% to 80% of UCE going through a provider's network. Depending on the exact situation, these numbers can vary significantly.

Where Can Spam Be Stopped?

There are two places to catch spam: inbound (messages on the recipient's, or recipient provider's, network) and outbound (messages on the sender's, or sender provider's, network). Both will be covered in detail: inbound in this installment and outbound next time. Usenet spam will be covered only briefly, as it is a much easier problem to solve than email spam.

INBOUND SPAM

Inbound UCE can be controlled utilizing the following methods:

- Source identification (including Realtime Blackhole List, Open Relay Database)
- Source analysis (Spamassassin)
- Source analysis services (Brightmail, Postini, etc.)
- Distributed key methods (Vipul's Razor and Rhyolite Software's Distributed Checksum Clearinghouse, or DCC)
- Electronic coin methods (camram, which is actually a combination of inbound and outbound methodologies)

Each method will be examined in detail. The newest and most promising of the bunch are the distributed key (Vipul's Razor and DCC) and electronic coin (camram) methods. Brightmail and Postini are very effective but are costly.

OUTBOUND SPAM

Outbound UBE can be controlled using the methods below:

- Log analysis
- MTA controls
- Authentication before sending (POP before SMTP)
- Mail message metering

Each one of these methods will be examined in detail. Disclaimer: this author developed a pending patent for the "mail message metering" method. (An article by this author titled "Mail Message Metering" in the December 2000 issue of *;login:* covered this solution in detail.) This article covers each method outlined above as it pertains to inbound spam. The next installment of ISPadmin will cover methods for dealing with outbound UBE.

Sendmail

Sendmail began implementing anti-spam relaying mechanisms as of about version 8.8. While most of the focus is on stopping outbound spam, version 8.8 included support for Realtime Blackhole List (RBL).

Here is a list of a few Sendmail features that can help control inbound spam:

- Access control database mechanism (/etc/mail/access)
- Connection rate throttle
- Limited number of recipients per message
- Disallowing connections from open relays, dial-up IP addresses, or black hole lists (e.g., RBL)

THE ACCESS DATABASE

All Sendmail databases are created from simple text files with key/value pairs. A program converts the text file into a quickly searched database format. Values and their meanings include:

OK	Allow message to pass; overrides other checks.
REJECT	Refuse connections from that host.
DISCARD	Silently delete message.
<message>	Reject message, informing sender of <message>.
RELAY	Allow domain/IP address to relay mail through this server (more on this in next installment's discussion of outbound spam).

For example, the entry below would reject any mail coming from the domain "knownspammer.com" but allow mail from "friend@knownspammer.com", and silently discard mail from "spammer@spamhaus.com":

knownspammer.com	REJECT
friend@knownspammer.com	OK
spammer@spamhaus.com	DISCARD

All Sendmail databases are created from simple text files with key/value pairs.

Note that the configuration file must be built with the FEATURE(access_db) in your sendmail.mc and converted to sendmail.cf utilizing the m4 utility. Also, the Sendmail daemon must be restarted to re-read the new configuration.

SENDMAIL CONFIGURATION TWEAKS

A connection rate throttle will limit the number of connections per second from a given server. The line below (in sendmail.mc or its equivalent) enables this feature:

```
define('confCONNECTION_RATE_THROTTLE',3)dnl
```

Limiting the number of recipients per message can be a good way to limit some spammers. (Enabling this feature may cause problems with legitimate bulk emailers.) The restriction can be accomplished by adding the following line to the sendmail.mc:

```
define('confMAX_RCPTS_PER_MESSAGE',25)dnl
```

For the purposes of this article, "simple filter service" means a service that requires nothing more than an existing message transport agent, or MTA (e.g., Sendmail). Other filter services that require additional machines (e.g., certain Brightmail offerings) will be covered below.

The Mail Abuse Prevention System, or MAPS, maintains several IP databases for "black holing" spammers. When subscribing to a service like RBL, the service essentially routes traffic from sites in the list to the bit bucket. As a result, the end subscriber will never be able to receive packets (i.e., email) from a machine on the list. MAPS is a commercial service, except for "Individual/Hobby" sites. The databases it maintains are as follows (note that MAPS controls what IP addresses are placed into these lists):

RBL+	combines RBL, RSS and DUL
RBL	Spammers, the granddaddy of them all
RSS	Relay Spam Stopper, list of open mail relays
DUL	Dial-Up User List, contains addresses that shouldn't be originating mail (for example, dial-up IP pools, DSL/cable modem customer IP pools, etc.)
NML	Non-Confirming Mailing List, mailing lists that do not verify email addresses of subscribers

More information is available on the MAPS Web page. Such lists can be activated within Sendmail by adding line(s) similar to the following to the sendmail.mc file:

```
FEATURE(dnsbl, 'blackholes.mail-abuse.org',
    'Rejected - see http://www.mail-abuse.org/rbl/')dnl
```

Note that depending upon what type of services desired, changes to the DNS configuration need to be made in order to properly activate the MAPS services.

One other black hole service bears mentioning: the "OR" family of open relay black hole services. This service, in its various iterations, has been the subject of much debate within the anti-spam community. This is due to the ultra-strict checking that some of these open relay checkers perform. Several have been shut down due to litigation and/or threats of litigation. The Open Relay Database (ORDB) is one of the more well known ones. The OsiruSoft site contains many others, as well as the ability to check names/IPs to see if such names/IP addresses appear in some well-known open relay databases. This can be a very useful tool. Care must be taken, however, as some of these services are considered by some to be overly restrictive.

What to Do With Email Tagged as UCE?

One problem common to many solutions that attempt to filter spam (including the distributed key method as well as services like Brightmail) is what to do with the message once it is identified as UCE. The possible dispositions of such messages are:

- Deletion
- Tagging
- Sidelining

For a service provider, deletion of a potential spam message is most likely not an option. Paying customers do not appreciate having their mail deleted, given the lack of a common definition of spam as well as possible false positives (tagging messages as spam that are, in fact, not spam).

Tagging (adding a header – e.g., X-Spam-Score: – to a possible spam message) is a good idea, but, unfortunately, the only common mail client that supports arbitrary header filtering is Eudora 5.1. Neither Netscape Mail nor Outlook Express currently support such filters. (Mozilla 1.0RC2 as well as Netscape 7 Preview Release 1 both support this type of filtering.)

Sidelining involves sending a potential spam message to an alternate email box (usually a Webmail machine). Once the message is sidelined, the recipient needs to check the sidelined mailbox and dispose of the mail. Hopefully, it is all indeed spam. As a practical matter, until a perfect anti-spam solution is available, some non-spam mail will probably get tagged as spam. Sidelining requires additional hardware, as well as added complexity (and associated costs) for the organization running the email domain.

As a practical matter, until a perfect anti-spam solution is available, some non-spam mail will probably get tagged as spam.

Mail Filtering and UCE

Several programs exist which attempt to filter mail on their own, without external intervention. Such mail filters (and services) are a good method to stop spam.

STAND-ALONE MAIL FILTER PROGRAMS

Spamassassin can be used to identify spam by various attributes in the mail header and body. It has hooks for DCC and Razor (see below) and supports black holing via MAPS and other methods. Another such program (which is also a virus checker) is MailScanner. Figure 1 illustrates how these mail filter programs might be implemented in an ISP's mail infrastructure.

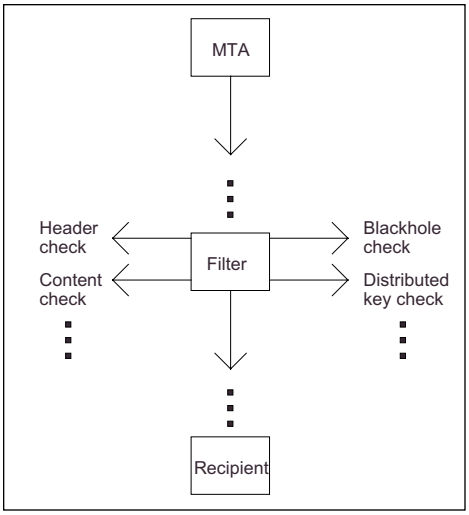

Figure 1: Message flow through static filter

The MTA box in Figure 1 represents Sendmail, qmail, etc. It has hooks (via procmail, milter or other mechanisms) to call an external program to process a message. In this case, the program is indicated by the box labeled "Filter." The filtering program then makes a number of checks against the header and body of the message, utilizing internal static filters, or external filter sources (such as black hole lists or distributed key hosts). If the message is identified as spam, a score is assigned to the message. The program then adds a header indicating that the message has been processed, along with the score it received. The message then is sent along to the rest of the mail infrastructure until it reaches the recipient. The recipient's mail client is responsible for determining what to do with messages tagged as spam.

The limitations of such static filters are such that they will never tag 100% of spam, and they also have false positives. Another downside is the requirement that such fil-

ters be constantly updated, as spammers are always creating new ways for their messages to escape filters.

DISTRIBUTED KEY METHODS

One of the most promising anti-spam ideas of late is the concept of exchanging keys summarizing mail messages with a mail server's peers. Essentially, these are static rules which count the number of times a server has processed a version of a message. The message checksums are then exchanged with other servers so that everyone knows to block a certain message based on its checksum.

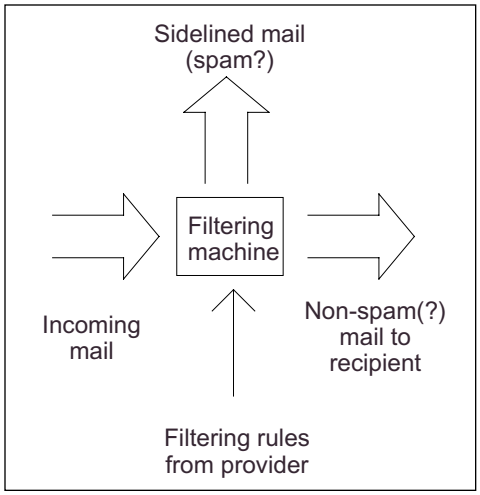

Figure 2: "Filter Service" message flow

The downside to such systems is the requirement the mail server keep "white hat" lists of legitimate bulk email, as such mail will have the same high hit counts that UBE will. If such mailing lists are not placed into the white hat list, then the legitimate bulk email will be tagged as spam. Also, the fuzzy algorithms which the programs use to identify spam need to be tweaked periodically as spammers adjust their methods to avoid detection systems.

However, a distributed key filtering method can block an impressive amount of spam with minimal work and is certainly worth considering. The best way to implement distributed key solutions is via something like Spamassassin (mentioned above).

FILTERING SERVICES

Services (such as Brightmail and Postini) are similar in nature to the DCC and Vipul's Razor applications. These services will, for a fee, maintain filter "lists" which identify certain messages as potential spam. These filtering rules are applied to incoming mail before hitting a customer's mail infrastructure. Figure 2 is an example of how such a filter service might be implemented.

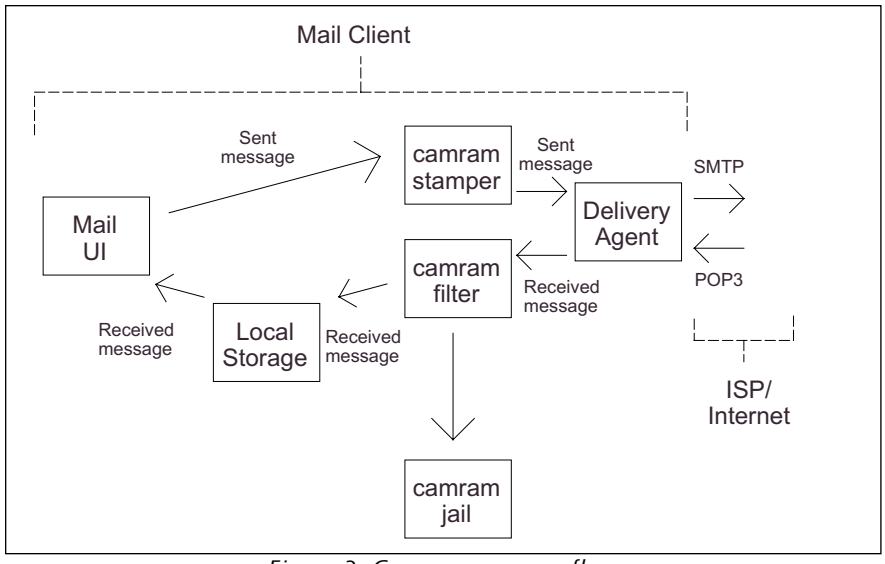

Figure 3: Camram message flow

Mail arrives at the filtering machine from the provider's mail relay. This filtering machine has human-built filters applied to it in a bulk manner from the anti-spam service provider, such as Brightmail. These rules are applied to incoming mail, and messages which the filters identify as UCE are sidelined to be viewed by the recipient at a later date. If the message is not tagged as spam, it continues through the provider's mail system.

Such anti-spam filtering services differ from the distributed key mechanisms in a two important ways:

- They use human-based rather than machine-based filters.
- They usually require additional hardware running in front of a customers existing mail relay(s).

These commercial services will catch a lot of spam. However, they can be costly (start-up costs range in the tens of thousands of dollars, plus a per-user per-month fee) and while good, are not perfect.

Electronic Coin Method

At least one solution, camram, is based on the idea of electronic "coins," also known as HashCash. Money is not really exchanged; rather, computational time is required on the sender's machine in order to generate the "coin." Figure 3 diagrams how camram would be set up in an ideal situation. "Ideal situation" refers to a mail client with embedded camram support.

It is important to realize that under this scheme, camram support must be on the same machine that is running the user interface. As a result, supporting a solution such as camram will require client-side changes. These changes are required so that it is relatively easy for one machine to generate a small number of coins; however, it is very time-consuming for a machine to generate a *large* number of coins.

One way to minimize such changes is to run camram in a proxy mode whereby camram functionality is implemented separately from the mail UI. This will likely be a "transitional" mode, as once code is placed in all widely used mail clients, the need for such proxy programs disappears.

In Figure 3, inbound mail comes into the client machine and passes through the camram filter, which determines whether or not appropriate "postage" exists for the message. If there is no or not enough postage, the message goes to a "jail," which is similar to sidelining. If there is enough postage, the message continues on its journey, eventually making it to the recipient. It is important to note that exception lists can be made for legitimate bulk mail that doesn't have appropriate postage.

Outbound mail passes through a camram "stamper" which "affixes" enough postage to the message and sends the message on its merry way. The recipient's mail client would need a camram decoder, if they wanted to filter spam. Otherwise, if they didn't use a camram decoder, the special headers would simply be ignored by the non-camram-aware mail headers. Of course, the recipient could choose to treat messages that haven't been signed by camram as junk mail, and deal with it at a later time.

A side effect of coin generation is the ability to easily add some level of public key encryption. With camram, you already know a lot about the sender, and adding a digital signature would be an easy modification to the protocol.

The camram concept's big appeal is its ability to significantly slow persons generating large amounts of spam. However, it still requires a white list for legitimate bulk emailers, so in that respect it is very similar to the distributed checksum method. A spammer would likely not use a solution such as camram, as the sender would quickly run out of "coins." The issue with an idea such as camram is simply one of adoption: the wider its use, the more useful it is. Time will tell if a HashCash-based solution works effectively or not.

Conclusion

I wish to thank Eric Johannson for his input into this article (Eric is one of the people behind camram, who is looking for help on the project; check the camram site for more information). Next time, I'll continue my look at spam by examining how to stop it at the outbound side. Until then, please send me your comments.

REFERENCES

abuse.net's anti-spam pages for admins – *http://spam.abuse.net/adminhelp/mail.shtml*

Blackmail – *http://www.jsm-net.demon.co.uk/blackmail/blackmail.html*

Brett Glass, "Stopping Spam and Malware with Open Source" – *http://www.brettglass.com/spam/paper.html*

Brightmail – *http://www.brightmail.com/*

Camram – *http://www.camram.org*

Dan Garcia's Spam Homepage – *http://www.cs.berkeley.edu/~ddgarcia/spam.html*

Eudora – *http://www.eudora.com/*

HashCash –*http://www.cypherspace.org/~adam/hashcash/*

Kai's spam shield – *http://spamshield.conti.nu/*

Mail Message Metering – *http://www.ziplink.net/ziplink/solutions/mmm/*

MailScanner – http://www.sng.ecs.soton.ac.uk/mailscanner/

MAPS/RBL/DUL – *http://www.mail-abuse.org/*

Milter – *http://www.milter.org/*

Monty Python Spam – skit *http://www.ibras.dk/montypython/finalripoff.htm#Spam*

Obtuse System Corp's Juniper firewall –*http://www.obtuse.com/smtpd.html*

ORDB – *http://www.ordb.org/*

OsiruSoft Open Relay black list – *http://relays.osirusoft.com/*

Postini – *http://www.postini.com/*

Procmail – *http://www.procmail.org/*

Qmail – *http://www.qmail.org*

Rhyolite Software's DCC – *http://www.rhyolite.com/anti-spam/dcc/*

Sendmail – h*ttp://www.sendmail.org/*

Sendmail's anti-spam pages *http://www.sendmail.org/antispam.html*

Spamassassin – *http://spamassassin.org/*

The Complete Monty Python's Flying Circus: All The Words (Pantheon Books, 1989); ISBN: 0679726470.

Vipul's Razor – *http://razor.sourceforge.net/*