Automatic Filtering of Unsolicited Mail

A Bin for Trash Mail

Unsolicited spam mail is a pain – no doubt about it. Something needs to be done to prevent spam from sealing the fate of our electronic messaging systems – and surprisingly enough there is a solution. Spam filtering is a method of letting the computer do the work. Once setup this will save you valuable time and effort

BY MARC ANDRÉ SELIG

Advertising messages where ever you go – checking your mailbox first thing in the morning puts a damper on your good mood for the day. Spam accounts for more than half of the global mail traffic today, for some more than half their mail is spam. But how can you protect yourself? Can a computer intelligently tag unsolicited mail or even filter it completely?

In the course of the last two months, around 5500 unsolicited messages hit the author's private mail accounts, accounting for over three quarters of all incoming messages. Extremely nasty. But my computer recognized almost 97 percent of these messages as spam. In this period the computer tagged only two messages as spam, although they were in fact solicited messages – this extremely low false positive rate means I can really rely on my computer to delete spam.

These figures from a real-life scenario indicate the major problems that face a user attempting to deal with spam. Firstly, the amount of data is enormous. A provider side spam filter would be useful, after all, 5500 messages amount to many megabytes of traffic across the wire.

Many providers refuse to even consider spam filtering. In this case, you have to take up the fight against unsolicited mail armed only with the toolset your distribution provides.

No perfect answer

Secondly, there is no one hundred percent reliable way of recognizing spam. This is why **heuristics** are used, but some spam still gets through, despite advanced technologies.

Thirds: Heuristics can also make mistakes, overlooking spam, or falsely recognizing a genuine message as spam. The latter error is the more serious of the two and should be avoided if at all possible. This leads to spam filters being configured to let messages pass in case of doubt.

What is Spam?

Before you start developing a defense strategy, you should have a clear concept of what you are up against. A typical mailbox will be hit by at least three different types of unsolicited message, and not all of them are spam.

Specific persons may bombard you with the same programs or email with the same content. This could be the umpteenth version of a really funny (honestly folks) joke for the whole family. Your old computer up in the loft keeps reporting the success or failure of a backup job which is no longer of interest to you. It is quite easy to cope with nuisances of this kind. After all, the

sender is known, and more or less any mail client provides simple filters that will move messages with specific properties to a separate folder. Figure 1 shows an example using Mozilla.

The second category of unsolicited mail comprises of Windows worms that use email to propagate. The size of this category will depend to a great extent on the number of Microsoft users in your vicinity. Fortunately, Linux systems are not typically affected by this kind of attack, although it is obviously a nuisance. After recognizing the worm, you can again apply some simple filters to eradicate the problem, saving your time in the future.

The last category comprises of genuine advertising mail, or UBE ("*U*nsolicited *B*ulk *E*mail"). We will be dealing with this category in the rest of this article.

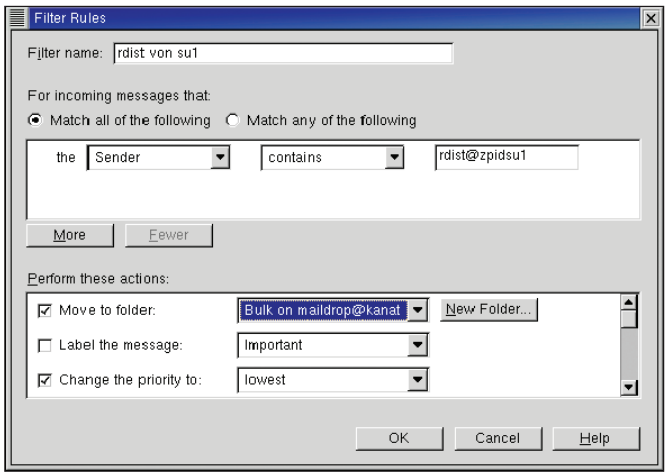

Figure 1: A Message Filter in Mozilla

Avoidance

The simplest kind of spam avoidance is to stay off the spammers lists. This involves keeping to a few simple rules: Never give anyone your address! Use disposable addresses for every type of communication! Never use your mail address in Usenet discussion forums or on the World Wide Web! Never compose any email messages! After all, your correspondents are bound to pass your address on some time – either to friends or business partners, or inadvertently due to a virus attack on their systems.

Of course, you will have noticed by now that these suggestions are completely illusory. You can apply some of these rules to mobile device addresses, using a non-intuitive email address for your cellular phone, for example, and not allowing public access to the address. You can then have any messages from that selected group of people forwarded to your cellphone. In a real-life situation, you simply have to accept that any genuine email address will end up on a spammer's list sooner or later. So let's try to re-formulate our task by asking: How can I get rid of spam?

GLOSSARY

Heuristics: *(from the Greek heurískein = find, discover) procedures for problem resolution based on experience, rules of thumb or algorithms.*

Filtered

The answer is by filtering it out. Spam filters can be installed on your provider's machines or on your own. The former approach has enormous benefits. For one thing, you will never be confronted with spam that is filtered by your provider. You will not need to download spam across slow ISDN links, or wait until your computer reaches a decision on the status of the message – it is as though the message was never sent. Unfortunately, you need a really co-operative provider for this and this description does not apply to many mass mailers. In addition, the task of configuring a spam filter on a remote system is non-trivial.

So let's stick to local filters. Again their are innumerable variants. To provide a better understanding of these, let us first take a quick look at how some email messages actually reach your system (see Figure 2).

The MTA or "*M*ail *T*ransport *A*gent" is only responsible for forwarding messages. This task is traditionally performed by a mail server, such as *sendmail*, *qmail*, or *postfix*, that talks to other mail servers via SMTP ("Simple *M*ail *T*ransfer *P*rotocol"). If you do not have a mail server permanently attached to the Internet, you will need to use a different method to retrieve your messages from a provider.

The traditional solution is to call *popclient* or *fetchmail* to fetch the messages from your ISP's POP server and forward them to another MTA, your local mail server. Your local mail server recognizes that these messages have reached their final destination and pass them on to the MDA or "*M*ail *D*elivery *A*gent".

Powerful Linux

Whereas */bin/mail* was formerly the typical MDA, most modern Linux systems today use the quicker and more powerful *procmail* MDA. The MDA writes the message to a file on the local computer, in the */var/mail* directory for example, or straight in to your home directory.

Powerful mail filters can be applied at this point. There are innumerable software packages, for *procmail* in particular, that provide efficient and reliable spam recognition capabilities that can be easily configured.

Finally, the user calls her MUA, or "*M*ail *U*ser *A*gent", the mail client proper. *pine* or *mutt* are some examples of common command line MUAs for

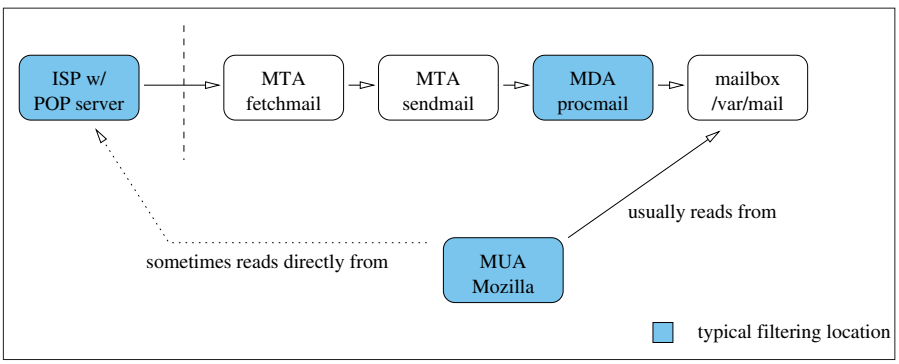

Figure 2: The path an email message takes to reach your computer

Linux, and their GUI counterparts are *kmail* or *exmh*, but you could just as easily use Netscape or Mozilla [1], of course.

Most modern MUAs do not require a functional local mail system, however, Mozilla can read your email from */var/mail*, if required, but is equally capable of retrieving messages from your provider.

It stands to reason that this approach is far more convenient for users, as they only need to configure and operate a single software package. However, this type of MUAs will bypass any mail filters, particularly those applied at Mail Delivery Agent level.

Filter Criteria

Spam filters are distinguished both by the point of application and by their approach. How is a computer to know whether a message is advertising or not? At present, three major, but fundamentally different, algorithms help reach this decision.

Text filters delete messages with a specific text element. They are easy to understand, simple and quick, but extremely susceptible to spoofing – and what's worse, they tend to discard absolutely harmless messages.

Filters that discover the true sender of a message are quite complex, slow, and difficult to implement, but extremely powerful. This makes them ideally suited for use on servers or with a classical Mail Delivery Agent.

Statistical filters are fairly exacting on the CPU and require interactive monitoring, which in turn requires frontend support. This is more than compensated by the fact that they are highly configurable. The current Mozilla versions [2] provide statistics based spam filters.

Text Filters

The first and simplest approach to spam detection is the simple text filter. If a message contains a specific text string, the filter simply discards it. The filter can check both the mail **header** and the body text.

Text filters are simple to implement and can be applied to any part of the mail system (see Figure 2). However, their approach does pose a number of problems. For example, inexperienced users tend to blacklist the address of a spam perpetrator. This approach should be avoided as spam message headers tend to be spoofed. The spammer would have to be really dumb to reveal his own identity as easily as that.

Filtering text strings in the mail body is also fraught with pitfalls. Deleting messages that contain a dollar sign might get rid of a lot of spam, but friends and business partners will no longer be able to send you messages, source code, memos containing dollar signs…

Some text filters are quite harmless and still discard a fair amount of spam. Not many native speakers of English use Asiatic character sets, for example. **Boxout 1** shows a few rules that support this approach. You can apply them as filter mechanisms for your favorite email program.

Incidentally, the new M2 mail program provided with Opera 7.11 [3] provides a selection of granular filter rules, which will automatically detect quite a lot of spam.

The Original Sender

Methods that attempt to determine the original sender of a message promise better hit rates and less false positives. Although the spammer will fake most, if not all, headers in an unsolicited mail, the message will pass through several known mail servers en route to your mailbox. Each of these adds at least one additional header, which is beyond the control of the spammer. However, you know which headers are normally added and can use this information to discover the original IP of a message.

The source IP of a spam message will either be the spammer's own computer or a compromised system, typically an insecure Windows system or an open relay.

All over the world many organizations concern themselves with documenting troublesome IP addresses of this kind. These "blacklists" are published in a simple format that lends itself to automatic querying. Some blacklists contain only open relays, others only IP addresses that belong to spammers. Some list IP addresses that are currently being used for spamming – non-verified, and not entirely reliable, but up-to-date. A combination of various blacklists should allow you to configure your spam defenses individually.

A detailed description of the installation steps is beyond the scope of this overview. You can refer to [4] for a sample configuration file, for example. Another approach that does not require as much manual customization would

GLOSSARY

Header: *A mail header comprises lines of transport and management information, followed by the mail body with the actual mail content. Many GUI MUAs hide a majority of mail headers (except the subject, the sender ("From header") and transmission date ("Date header") from the user.*

Box 1: Definite Signs of Unsolicited Mail

- The *Content-type:* header contains character set definitions uncommon to the Western World, such as *big5*, *euc-kr*, *gb2312* or *iso-2022*.
- The *From:* or *To:* lines do not contain an at sign @.
- The *Subject:* contains multiple space or tab characters, or several (not necessarily contiguous) tildes.
- *Subject:* contains codes typical of non-western alphabets (*=?big5*, *=?euc-kr*, *=?gb2312* or *=?iso-2022*).
- The *Subject:* line contains the *ADV* string, required by Californian anti-spam laws.
- *Subject:* contains multiple contiguous characters above ASCII 127.
- The message contains one of the following headers:

```
X-Library: Indy
X-Spam-Status: Yes
```

- The message contains one of the following *X Mailer:* headers:

```
X-Mailer: Bulk Email
X-Mailer: Easy Mass Mailer
X-Mailer: E-Master
X-Mailer: jpfree group mail
X-Mailer: mailer signature
X-Mailer: MailWorkZ
X-Mailer: SuperMail
X-Mailer:  V[0-9],[0-9],[0-9],[0-9]
X-Mailer: Vallen e-Mailer
X-Mailer: VUvacation
X-Mailer: X-Mailer
```

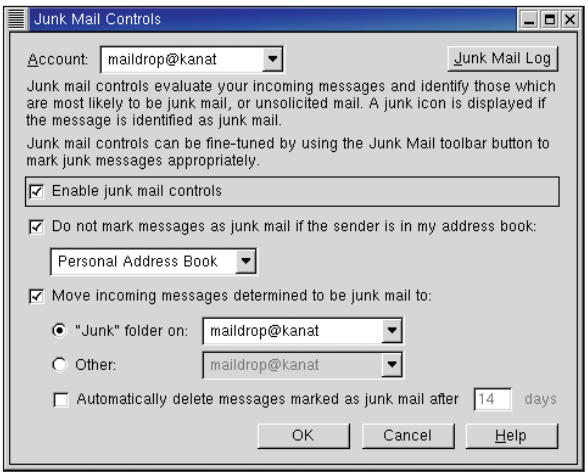

Figure 3: Statistical Filtering in Mozilla

be to install the popular anti-spam tool, SpamAssassin [5]. However, you should be aware of the fact that both approaches will only work if you use traditional mail retrieval methods (Figure 2). If your mail client retrieves messages directly from your provider, it will need to query the blacklists itself – and none of the popular MUAs are capable of doing that at present.

Statistical Spam Filters

So-called Bayes' filters are another interesting development on the anti-spam market. They boil down to simple statistics.

To start defining a filter of this type, you will need to explicitly tag a few mails as spam or non-spam. The filter program calculates the probability of a word belonging to a spam message for each individual word in the message.. A common word such as "have" will have very little influence on the "spam/non-spam" issue, but a word such as "buy" might be interesting.

When a new message arrives, the filter checks it word for word to decide whether it is spam or not. The filter program collates the results, and tells you what it suspects. If the program is right – well that's just fine, and if it gets things wrong, you can simply correct the error. The filter will then apply the words from the re-classified message directly to its word pool, and use them for future messages. The more use you make of a filter, the better it gets.

The mail client of the current Mozilla versions contains an integrated Bayes' filter. You can enable the filter in a "*Mail & Newsgroups*" window, via *Tools / Junk Mail Controls* (Figure 3). The next task is to tag each incoming spam message as such using the trashcan button in the toolbar (or by selecting *Tools / Mark Selected Messages as Junk*). A trashcan icon appears next to the message.

But the filter will soon start to work autonomously. This will involve tagging incoming messages with a trashcan icon, or filing them away in the specified folder (Figure 4). It is important to let Mozilla know that it has made a mistake by clicking on the trashcan icon or selecting *Mark Selected Messages as Junk-free* to remove the junk tag. This is the only way the system can learn.

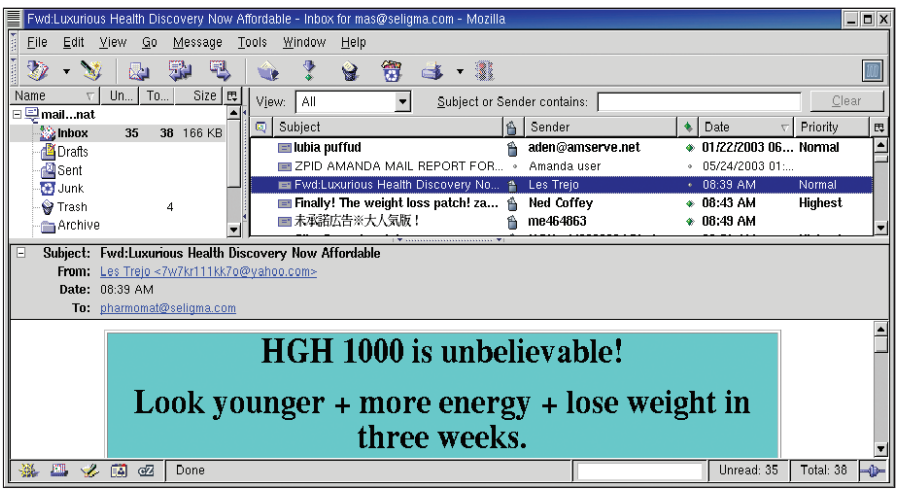

Figure 4: After a learning phase, Mozilla will automatically classify mail

Working well

Statistical filters work astonishingly well. But multilingual users are in for a surprise. Of course, you will distinguish between languages. You may receive a large amount of private messages in Spanish for example, but only a few in English.

However, spammers typically do not pay attention to a user's native language – most junk mail thus tends to be in English, and this may lead to a statistical spam filter categorizing English messages as spam, and Spanish messages as non-spam.

This is one good reason for not letting statistical filters delete mail automatically. The same approach applies here as to other filter types – check your mail manually before you dispatch alleged junk mail to the happy hunting grounds. You should wait a few weeks or even months, and ensure that you really are satisfied with the filter results, that is that you have not lost any legitimate messages, before you delete spam automatically.

When that day comes, you will be able to celebrate an important victory in your own personal battle against electronic junk. ■

INFO

[1] Andrea Müller, Patricia Jung: "Mail and more", Linux Magazine issue 29, 2003, p. 44

[2] Mozilla: *http://www.mozilla.org/*

[3] Opera: *http://www.opera.com/*

[4] Using blacklists with *procmail*: *http://www.ordb.org/faq/#usage_procmail*

[5] SpamAssassin: *http://www.spamassassin.org/*

[6] An early description of statistical spam filters: *http://www.paulgraham.com/spam.html*

THE AUTHOR

Marc André Selig spends half of his time working as a scientific assistant at the University of Trier and as an ongoing medical doctor in the Schramberg hospital. If he happens to find time for it, his currenty preoccupation is programing web based databases on various Unix platforms.