

how it works

OPEN-SOURCE STATISTICAL
SOFTWARE ELIMINATES
UNWANTED E-MAIL.

Squashing spam

By Brian Dipert, Technical Editor

LIKE, I SUSPECT, many of you, I get a lot of spam. On weekdays, I average around 400 e-mails per day, many of

them consolidated digests containing numerous postings to the discussion lists to which I subscribe. The 75 to 100 spam e-mails out of that total that I'm receiving daily (the number seems to grow each month) are serious impediments to my ability to efficiently process the flood of incoming information. On weekends, the total amount of e-mail I receive decreases, but the volume of spam doesn't. Roughly three-quarters of the 100 e-mails I receive on an average weekend day are spam.

This ever-growing spam deluge is the impetus behind my recent evaluation of various antispam software packages and the detection algorithms that they run. The good news is that I've found a spam-swatting technology that works well for me, and, even better, it's free. The bad news is that I had to sort through a lot of poorly implemented products before I stumbled across a mention of my eventual winner in a magazine article. By first understanding the techniques and shortcomings of these also-rans, the superiority of the Bayesian-based heir ascendant becomes easier to comprehend in comparison.

UNFULFILLED EXPECTATIONS

A friend of mine likes to say, "If your only tool is a hammer, every problem looks like a nail." His analogy refers to FPGA-design tools, but it could apply equally well to traditional spam-fighting software. Reflective of just how invasive, offensive, and resource-sapping spam is, folks are frantically installing antispam utilities on their client PCs and e-mail servers in spite of the well-documented shortcomings of these crude first-generation tools.

These products' primitive algorithms employ combinations and variations of two general techniques in attempting to differentiate between spam and valid e-mail, or "ham." First, they search for of-

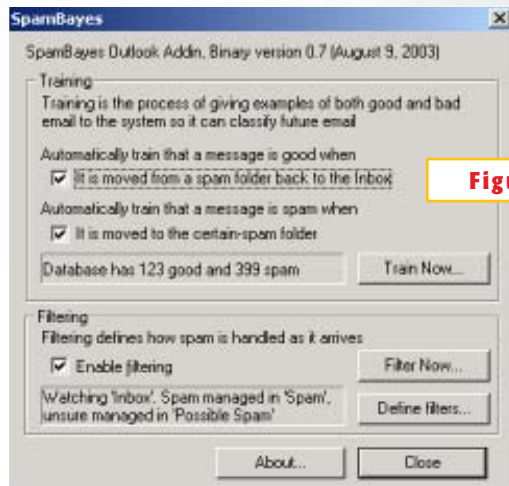
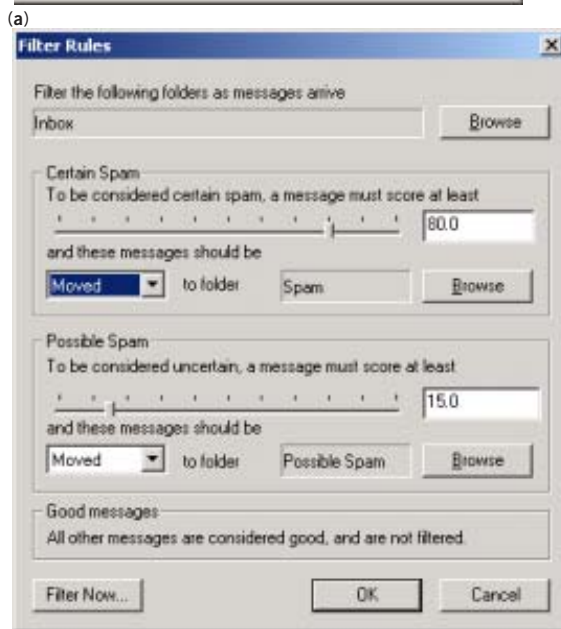


Figure 1



(b) SpamBayes can run either stand-alone or as a plug-in integrated within Outlook (a). You can customize its responses when incoming e-mail arrives, as well as the probability scores that determine whether each message is "ham," potential spam, or spam (b).

fensive keywords within the e-mail's subject line and (sometimes) body text. The limitations of this approach are almost immediately apparent; the "breast" that one person may find offensive, for example, may provoke a different reaction from someone who's subscribed to an online cancer-support group. Spam senders can also nimbly sidestep keyword searches by using synonyms for offensive terms or by intentionally misspelling words; replacing o's with zeros and i's with ones; or putting asterisks, dashes, spaces, or other symbols between consecutive letters.

The other and even cruder method for screening out spam involves blocking (that is, blacklisting) specific e-mail addresses or, more generally, all e-mail sent from certain domains. The more encompassing approach has on several occasions caused me problems as an e-mail sender. For example, overaggressive spam-screening software may decide that *all* e-mail coming from a Yahoo server is undesirable. (This kind of software often broadly blacklists AOL- and Hotmail-sourced e-mail, which spammers commonly use, in the same manner.) Although my e-mail address defines a "pacbell.net" domain, the server it comes from has evolved over the years from Pacific Bell to SBC Communications, and, via the SBC Yahoo partnership, to Yahoo.

Earthlink and some other ISPs (Internet-service-providers) implement another twist on the e-mail-screening practice. Enabling their spam-blocking software allows only e-mail from addresses you specify in your account setup (that is, a "whitelist") to flow through the ISP's server to your inbox. When the server receives an e-mail from an unapproved address, one or multiple things happen. Sometimes, the intended recipient receives a short administrative notice indicating a pending message. If the recipient responds in the affirmative, the software delivers the e-mail and adds the sender's address to the approved list. Other times, the software sends an autoresponse, requesting detailed contact information, to the sender; it then transfers this data to the intended recipient for perusal, so that he or she can supposedly more intelligently screen the message and its source. For someone like me, though, with such an extensive set of contacts and such a large volume of daily e-mail, this tedious back-and-forth approval cycle is a far bigger hassle than simply hitting the "delete" key when I get an e-mail I don't want. Also, autoreponses to e-mail senders have the unwelcome consequence of alerting spammers that they possess a valid e-mail address, which they'll subsequently sell to other spammers, further exacerbating your spam problem.

FALSE POSITIVES AND UNFINISHED BUSINESS

Version 4 of McAfee's SpamKiller, the product I

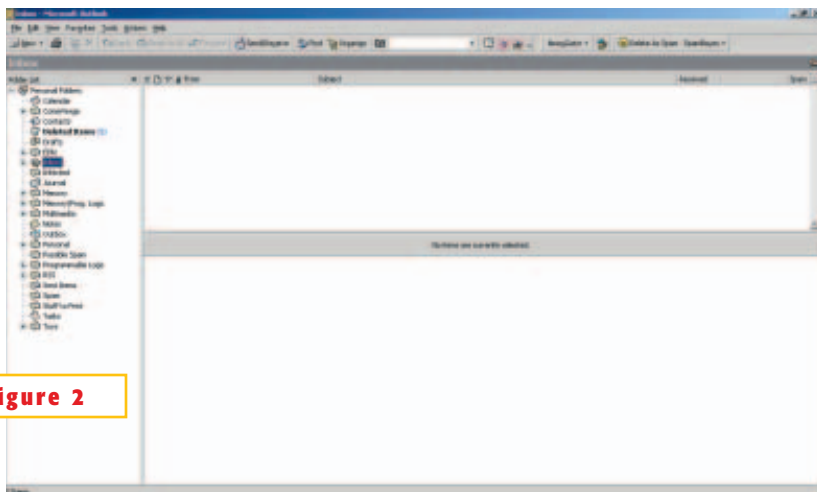
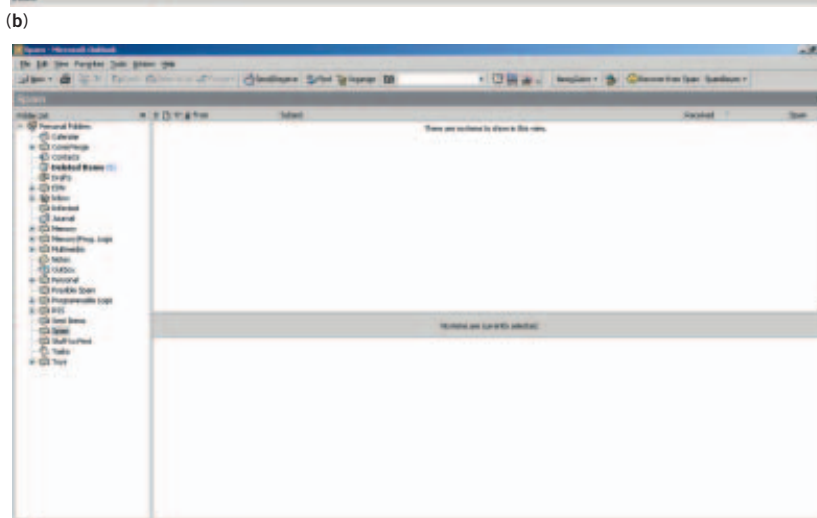
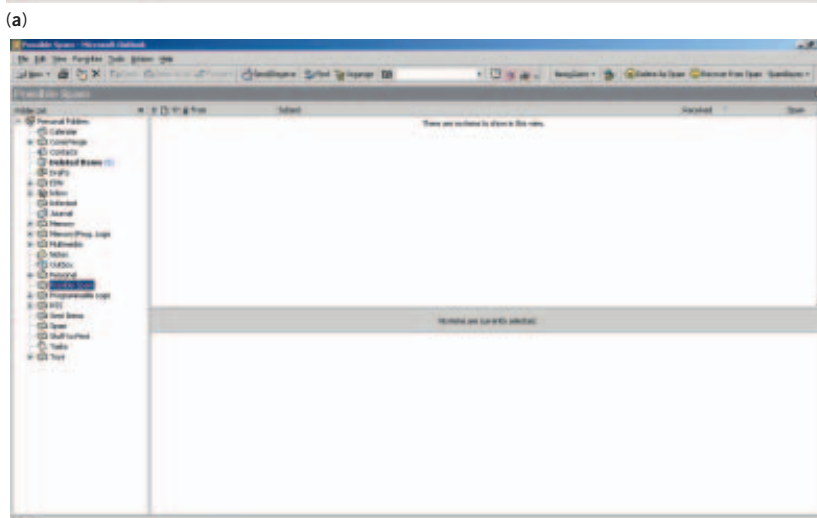


Figure 2



(a) If spam makes it through SpamBayes, a single mouse-click moves it to the appropriate folder and trains the software to reduce the probability of a repeat occurrence (a, see upper right-hand corner). The software just as easily tags potential spam as "ham" or spam (b) and quickly corrects false positives should you ever encounter one; I haven't (c).

most extensively evaluated before discovering my current champion, puts another twist on e-mail inspection. As part of its installation, it attempts (if you allow it) to import all of the e-mail addresses for the contacts in your Eudora, Outlook, or Outlook Express address book, subsequently treating all e-mail coming from any of these senders as “friendly.” Unfortunately, this feature didn’t work for me, because SpamKiller couldn’t find my Outlook 2000 data, perhaps because I store my Outlook database files in a nonstandard location for a Windows XP-based system, due to their Windows 98 legacy. But, given the diversity of my incoming e-mail, this “friends” feature would have been of limited value even with nearly 3000 names in my address book. And anyway, who among you regularly gets spam from “friends”?

SpamKiller runs in a stand-alone fashion, separate from your e-mail program. It regularly monitors Hotmail-, MAPI-, MSN-, or POP3-based e-mail accounts you define, and it downloads messages that its e-mail- and keyword-based filters identify as suspect, subsequently deleting them from the server so that, ostensibly, they won’t clutter your inbox. McAfee claims to regularly release filter updates, and the program’s nonintegrated approach has the advantage of making SpamKiller generic to any e-mail program you might use (aside from AOL). Once an e-mail’s sitting in the SpamKiller window, you can tell the software (if it has misidentified the message as spam) to in the future accept either all messages from this sender or all messages from any sender in this domain. You can also resend the message from within SpamKiller, so that the next time your e-mail program checks the server, it will be able to download it.

If SpamKiller worked well, it would represent a compelling option to solving the spam problem. Unfortunately, it doesn’t. Therefore, I found it to be more trouble than it was worth. For one thing, you need to disable automated periodic server-checking from within your e-mail program so that SpamKiller has a chance to do its housekeeping. More generally, any lag time between when SpamKiller last cleans out your server and when you check for messages represents an opportunity for spam to sneak into your inbox.

Any antis spam-software developer will tell you that users are far more willing to tolerate the occasional spam message that slips through the filter than a “false-positive” result in which software incorrectly tags e-mail as spam. SpamKiller inappropriately yanked at least 20 valid e-mails per day from my server (sometimes as much as three times this number on weekdays), and it didn’t tone down its overaggressive behavior to any noticeable degree through the three weeks I used it. These false positives were particularly irritating because of

SpamKiller’s stand-alone nature; I was perpetually using ALT-TAB to move from Outlook to SpamKiller, training SpamKiller that each nonspam message’s sender was valid and, in a separate operation, sending the e-mails back to the server so that I could deal with them in Outlook.

McAfee claims that you can customize the filters, and I suppose that if I spent enough time doing so I could diminish the volume of false positives to a certain degree. But, just as with voice-recognition software, unless the user gets a reasonably robust “out-of-box” experience, he or she will be unmotivated to take the next step. Coupled with the disturbing volume of false positives, an equally disturbing high number of spam messages (roughly half) consistently slipped through SpamKiller’s trap. Simple keyword filtering is too easy to circumvent.

A SOUND OF THUNDER

Fans of author Ray Bradbury will recognize this section’s header as the title of one of his science-fiction short stories, which happens to be my favorite. It’s about a group of people who go back in time. (I won’t reveal any more of the plot.) After a frustrating search for a robust keyword-based antis spam algorithm, programmer Paul Graham (www.paulgraham.com) also went back in time, turning to the equations of Thomas Bayes, an obscure (except, perhaps to statisticians) 18th-century mathematician and Presbyterian minister.

Realizing that it wasn’t so much individual keywords themselves but their *combination* and *context* that defined whether an e-mail was desirable or not, Graham’s work leveraged Bayes’ theorem’s use (published in 1763 as an “Essay Towards Solving a Problem in the Doctrine of Chances”) of statistical inference to update probability estimates of the truth of different hypotheses, based on observations and a knowledge of each observation’s likelihood given each hypothesis. Graham’s breakthrough, like many good ideas, seems intuitively obvious in retrospect! His work, which other developers have subsequently enhanced and optimized to result in algorithms such as the Spam-Bayes Outlook filter I use (<http://spambayes.sourceforge.net>), isn’t *strictly* Bayesian; it’s fine-tuned for the specific task at hand (Figure 1).

Graham’s and successors’ programs come with predefined set of hypotheses derived from a prior

JUST AS WITH VOICE-RECOGNITION SOFTWARE, UNLESS THE USER GETS A REASONABLY ROBUST “OUT-OF-BOX” EXPERIENCE, HE OR SHE WILL BE UNMOTIVATED TO TAKE THE NEXT STEP.

analysis of spam that he and others had received. You then initially train SpamBayes by exposing it to two sets of e-mails: those you've already determined to be "good" and a folderful of spam. To optimize performance and memory usage, the algorithm does not calculate a spam probability score using all constituent elements, or "tokens," within each analyzed e-mail. Graham's initial code instead analyzes the 15 tokens whose individual scores are furthest away from an assumed neutral value of 0.5. (Subsequent iterations of his and others' work varied this exact number.)

Again, to optimize speed and system-resource use, Graham's code also focuses its analysis on text in the subject line and body, along with an application-tailored set of HTML tags within the e-mail. Interestingly, according to Graham, the presence of HTML-text color code "ff0000," referring to bright red, indicates a high probability that the message being scanned references pornography. Spam and potential spam, if you desire, go into different folders, based on threshold scores you can adjust. As you use SpamBayes, the iterative training process continues, and the program's accuracy improves. It's easy to tell SpamBayes that an e-mail that has made it through the filters is spam, that a message in the spam folder is ham, or that a potential spam message is spam or ham (**Figure 2**).

The first day I ran SpamBayes, about one-third of the spam I received ended up in my inbox, another third found its way to my potential spam folder, and the final third went directly to spam. Less than two weeks later, using the threshold settings shown in **Figure 1b**, iterative training has led to approximately 85% of my spam's ending up in the spam folder; roughly 10% is defined as potential spam, and only about 5% evades SpamBayes and ends up in my inbox. Equally important, I've yet to have a *single* valid message land in my spam folder, and SpamBayes tagged only *two* valid e-mails, both graphically rich, bright-red text messages, as potential spam. Words can't fully describe the elation I feel when I boot up my computer in the morning, launch Out-

look and watch 75 or so e-mails download, only to see two-thirds of them immediately and automatically leave my inbox and enter the spam folder!

AN ESCALATING ARMS RACE

Just as spam senders adapted their junk e-mail to dodge detection by crude first-generation filters, they'll do their best to evade Bayesian-based algorithms, too. But their options are far more limited this time around. Data on Graham's Web site indicates that, although misspellings, synonyms, and other alterations might make it through simple filters, such as those in SpamKiller, they won't trip up SpamBayes—especially after a bit of training. Ironically, one means of counteracting Bayesian filters is by *simplifying* the message—sending only a sentence or two containing generic, inoffensive wording, coupled with a URL. Although the volume of spam you see might not decrease if such a transformation occurs, the content of each message—arguably, the more disturbing aspect of spam, especially when children are involved—will diminish.

The other spam-sender counterattack, which HTML-friendly e-mail programs unfortunately allow, is to embed within the spam a link to a Web-site-hosted graphic containing the text and images. Blocking the display of e-mail images and reading e-mail only when you're offline and therefore unable to download the graphic are probably the only means of combating this approach, at least until more sophisticated anti-spam algorithms emerge that can analyze the zero-and-one patterns of JPEG and GIF images in real time. □

You can reach Technical Editor Brian Dipert at 1-916-454-5242, fax 1-617-558-4470, e-mail bdipert@edn.com.



ACKNOWLEDGMENT

Thanks to Mark Hammond, the developer of the SpamBayes-based Outlook plug-in.

AUTHOR'S BIOGRAPHY

Technical Editor Brian Dipert is a happier and more efficient computer user with his spam crisis under control. Reach him at 1-916-454-5242, fax 1-617-558-4470, bdipert@edn.com, and www.bdipert.com. But don't send him spam.