WHITELISTING IN PRACTICE

A RESEARCH REPORT
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE WITH DISTINCTION IN RESEARCH

David Erickson

June 2008

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Master of Science with Distinction in Research.

(Nick McKeown) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Master of Science with Distinction in Research.

(John Lockwood)

Approved for the University Committee on Graduate Studies.

Acknowledgments

First I would like to thank my wife who has been exceptionally patient with me while working many late nights on this project, as well as always being there for help and loving support. I would also like to acknowledge Martìn Casado for his initial work and exploration of the Default Off Email idea, as well as his continued support and ideas. Many thanks to my adviser, Nick McKeown, for giving me the opportunity and resources necessary to work on this project at Stanford, and for his great advice, support, and friendship. Lastly, a big thanks to all the past and present DOEmail users, without your support this project would not have been possible! Thanks also to John Lockwood for his willingness and time spent critiquing this work.

Abstract

While whitelists are common in social networking and instant messaging (*e.g.*, buddy-lists), they have not caught on in email systems. Instead, most people use spam filters that try to identify all the senders they do not want to accept email from. With whitelists these people only need to identify the much smaller set of individuals who *can* legitimately send them email - generally most of these individuals are known to them ahead of time. In this study I built and deployed a whitelisting email service - named DOEmail - along with a Thunderbird add-on to try and make whitelists easy to manage. During the past two years, over 120 users have used DOEmail. As expected, almost no spam makes it to users' inboxes, and less than 1% of legitimate email is mis-classified. Measurements are performed to determine how hard it is for users to manage their whitelists, with the conclusion that after initial setup, the burden is very low. DOEmail uses challenge-responses to identify whether unknown senders are legitimate, this work also beings to explore their effectiveness in an operational system.

Contents

Chapter 1

DOEmail

There are several existing email whitelisting services such as [13, 15, 16, 17, 3]; and in many ways DOEmail is similar to these. When designing DOEmail, the goal was to make it easy to use, and collect usage data for this study. DOEmail is not particularly novel, and shares many features with the systems above. The benefit of implementing this system was that it could be tailored for research purposes, specifically to collect the data presented in the results section below.

1.1 Overview

DOEmail provides a basic forwarding service. Users sign up for a DOEmail account and redirect their existing email through it.[1] Each DOEmail user has a doemail.org email address (*e.g.*, `derickso@doemail.org`), and their own whitelists and blacklists. Each user owns four whitelists:

- **Individuals.** Individual email addresses to always accept email from, *e.g.*, `person@domain.com`.
- **Domains.** Domains to always accept email from, *e.g.*, `domain.com`, or `.edu`.

[1] Section 3.2 describes how this works in more detail.

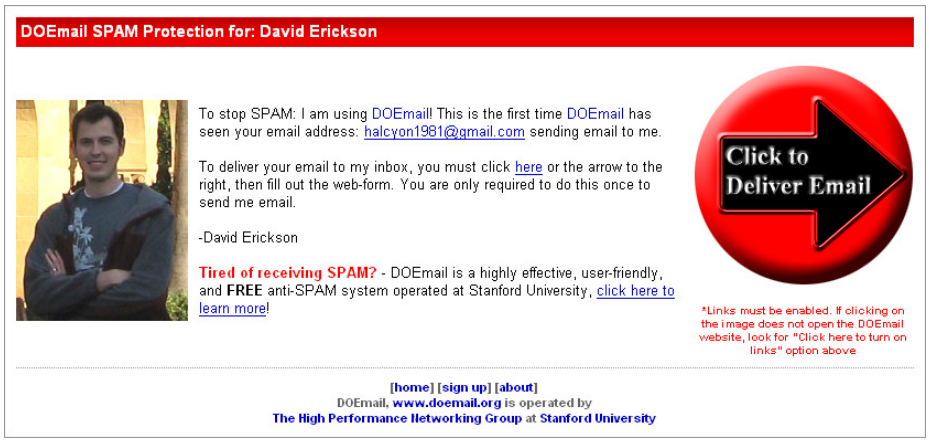


Figure 1.1: Example DOEmail challenge

- **Mailing lists.** Mailing lists to always accept email from, *e.g.*, `interesting_list@mailman.stanford.edu`. Mailing lists need to be handled differently (and carefully), and are discussed in more detail later.

- **Disposable addresses.** A list of addresses that can be given to an individual (or a web service) to reach us, regardless of the address they are sending from, *e.g.*, derickso+united_airlines@doemail.org; similar to those used in Mail Avenger [14]. Disposable addresses are easily revoked if a sender abuses them.

Similarly, each user also owns two blacklists: one for individuals, and one for entire domains.

In DOEmail, whitelists can be populated in three ways:

1. **Manually by recipient.** Users can manually add entries to their whitelist. The user interface is described below.

2. **Automatically add outgoing email addresses.** When an individual sends email, they typically desire to receive email back from the recipient. To assist this process, DOEmail allows users to automatically add all outgoing email addresses to the whitelist.

3. **Manually by sender.** If email is received from an address that doesn't match a whitelist or a blacklist, DOEmail sends a *challenge* to the sender, a sample

challenge can be seen in Figure 1.1. The challenge consists of clicking on a URL, taking the user to a website which (optionally)[2] contains a CAPTCHA[3]. When the sender completes the challenge, their email address is whitelisted and their email is delivered to the DOEmail user. The basic idea is that almost all SPAM is generated by computers; so SPAM-generating computers won't be able to get onto a whitelist.

This study examines many of the pros and cons of the last item on the list: the *challenge* that is sent to the sender. On one hand, it is expected that a CAPTCHA will raise the bar high enough to shut off almost all spam - and this study shows that it does. On the other hand, it may raise the bar too high for some senders. Friends, family and colleagues might be put-off, offended or confused when asked to fill out a CAPTCHA, or they might be unable to do so (*e.g.*, the visually impaired). This study tries to scientifically evaluate this tradeoff.

In summary, figure 1.2 shows the decision process DOEmail follows when an email arrives. First, DOEmail checks the blacklist; if it matches, the email is dropped.

Second, DOEmail checks to see if the email comes from another DOEmail server. There is a danger that two DOEmail users never whitelist each other because the *challenge* never reaches their inbox. Therefore, outgoing DOEmail challenges are marked,[4] so the challenge bypasses all checks and is delivered directly to the inbox.

Third, DOEmail checks to see if the email matches any of the recipient's four whitelists. If it does, it is delivered to the inbox.

Finally, if the email doesn't match a blacklist or whitelist, it is added to the user's *pending list*, and a *challenge* is sent to the sender. If the sender correctly completes the challenge, the email is removed from the pending list and delivered to the inbox; the sender's email address is added to the user's whitelist.

[2]DOEmail users can specify whether new senders must complete a CAPTCHA or not

[3]Essentially a distorted image that is easily recognizable by humans, but not by computers [20]. Currently DOEmail only supports reCAPTCHA [2] which contains a visual and optional audio-based turing test, but there is no reason this approach cannot be extended to other tests.

[4]The mark is a X-DOEmail-sig email header containing an RSA encrypted hash covering the body of the message and a timestamp.

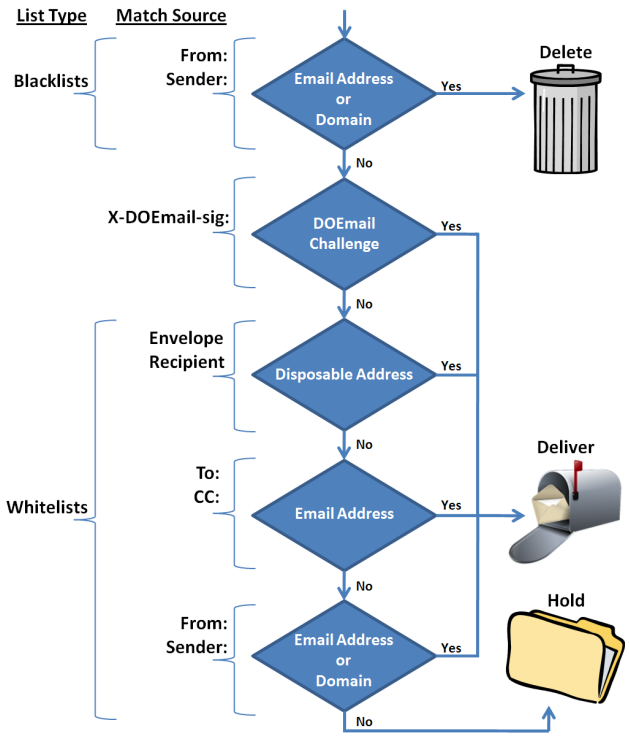


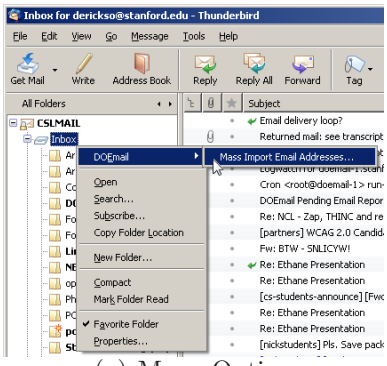
Figure 1.2: Decision flow diagram

1.2 Challenges and pending email

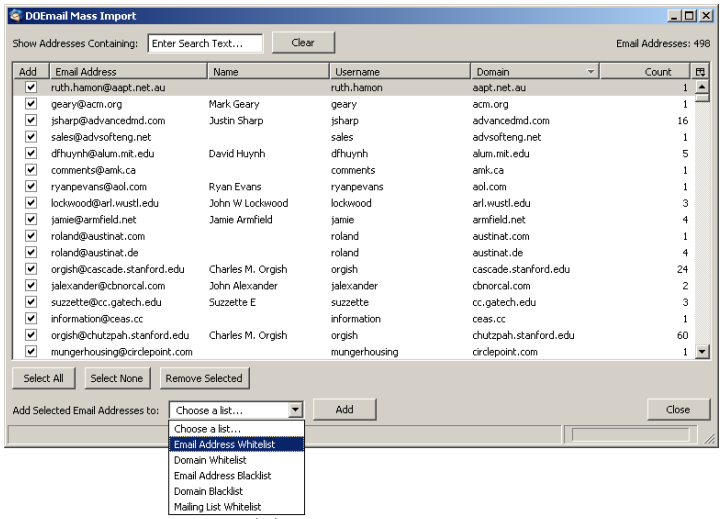
The challenge allows a human sender to add themselves to the recipient's whitelist, but makes it difficult (and not cost effective) for a spammer. While a spammer could perform the required functions to satisfy the challenge, it is unlikely they will take the effort to do so. Of the over one million emails DOEmail processed, it has only been verified as happening twice, both times from fraudulent Nigerian-based scams [18].

Similar to a spam folder, the user is able to view the contents of the pending email folder and can manually whitelist (or blacklist) email addresses and domains associated with waiting messages.

DOEmail holds pending email for up to three weeks from the receive date, and then deletes it.



(a) Menu Option



(b) Dialog Window

Figure 1.3: Mass Import

1.3 User Interface

When designing DOEmail, the main goal was to make it easy and appealing enough to use such that it would be feasible to encourage 100-200 users to participate in this study. It was quickly apparent that DOEmail needed an easy-to-use user-interface. Building good, intuitive user-interfaces is not easy and consumed a large part of this study.

Two user-interfaces were build to help users manage their whitelists, blacklists and pending email. The first is a Mozilla Thunderbird add-on, the second a web-based interface.

1.3.1 Thunderbird add-on

Thunderbird is a powerful, free, and widely-used email client, and was used by 44% of DOEmail's users.

The Thunderbird add-on has many features:

1. **Mass import of addresses to whitelist or blacklist.**

Figure 1.4: DOEmail toolbar button

To help new users of DOEmail quickly take advantage of whitelisting, and cause as little inconvenience as possible to pre-existing email senders, the ability was added for Thunderbird users to scan through existing email, and import email addresses and domains into whitelists and blacklists. Figure 1.3(a) shows a user selecting a mail folder in Thunderbird, right clicking to access the DOEmail context menu option, and selecting Mass Import Email Addresses.

After selecting this option, the Mass Import dialog window is displayed, an example is shown in Figure 1.3(b). From this dialog window, users can view all email addresses contained within the headers of the email folder they selected. Sortable columns displaying the email address, a name (if specified), the username and domain portions of the email address, as well as the count (or number of occurances) of the email address within the selected folder, are available to help navigate the list. A full text search of the addresses can also be performed. The ability also exists to remove addresses from the main working set, allowing a user to drill down and work with a specific subset of addresses. For example, if a user wanted to examine all non-Stanford email addresses, they could search for "Stanford", select all of the resulting addresses, click the "Remove Selected" button, and the working set would then contain all email addresses that did not contain the text "Stanford". Finally, when the user has selected email addresses of interest, they can be added to any of the whitelists or blacklists by selecting the type of list from the drop-down, and clicking add.

2. **Whitelist and blacklist management.** Figure 1.4 shows the DOEmail toolbar button used to access the add-on's main dialog window. The value in parenthesis on the button is the current number of pending emails stored for that user, which is updated periodically. The main dialog window, shown in Figure 1.5, has a tabbed interface where the first six tabs each contain a specific whitelist or blacklist and an interface to add and remove entries from

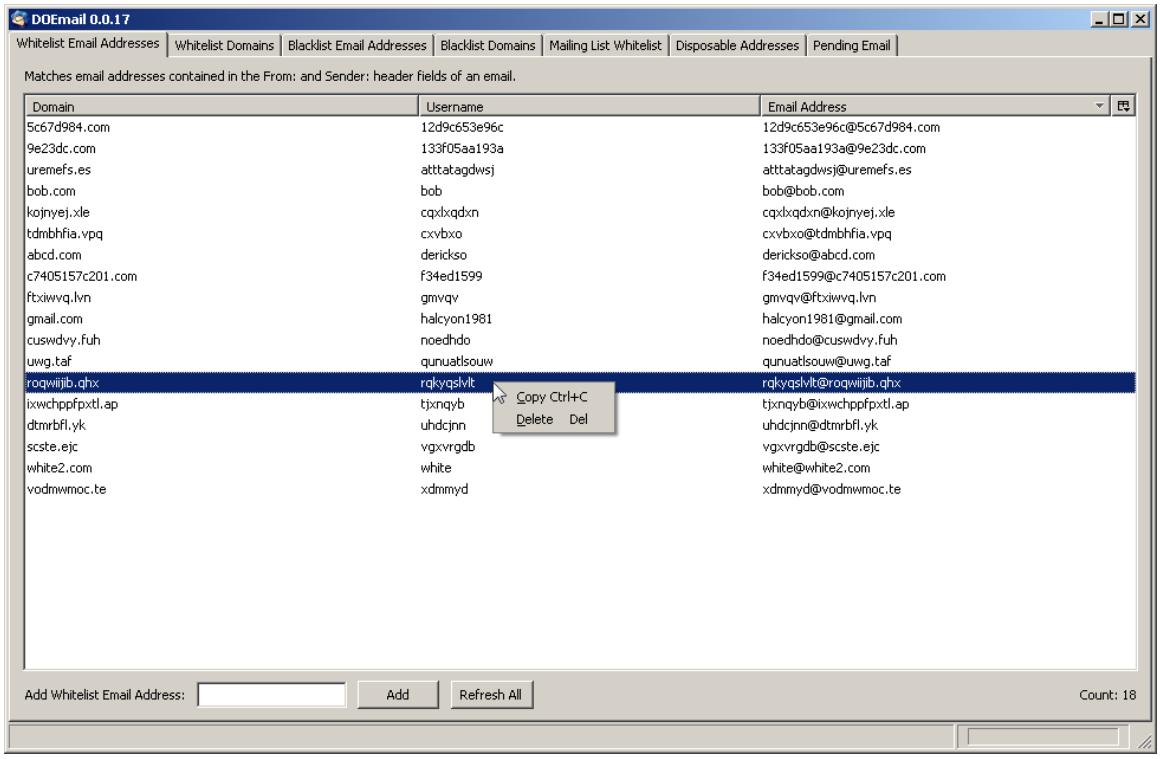


Figure 1.5: Email whitelist

that list. All the column headers are sortable, and can be multi-selected for quick deletion, or copying to the clipboard.

3. **Pending email folder.** The pending email tab (see Figure 1.6) displays a list of the user's pending email (*i.e.*, email for which a challenge has been sent, but not yet completed). The user can whitelist (or blacklist) the sender or their domain, and move the email to their inbox. Or the user can manually delete the email. To help the user decide, they can open and view the email in a Thunderbird email window, just like viewing an email from their inbox, or even view the email's raw source. Full text search of pending email is available, as well as hiding email above a selected spam score.

4. **Processing result.** Figure 1.7 shows the new column added to Thunderbird's main threaded email view, containing a graphic representing the type of rule that each email matched during DOEmail's processing

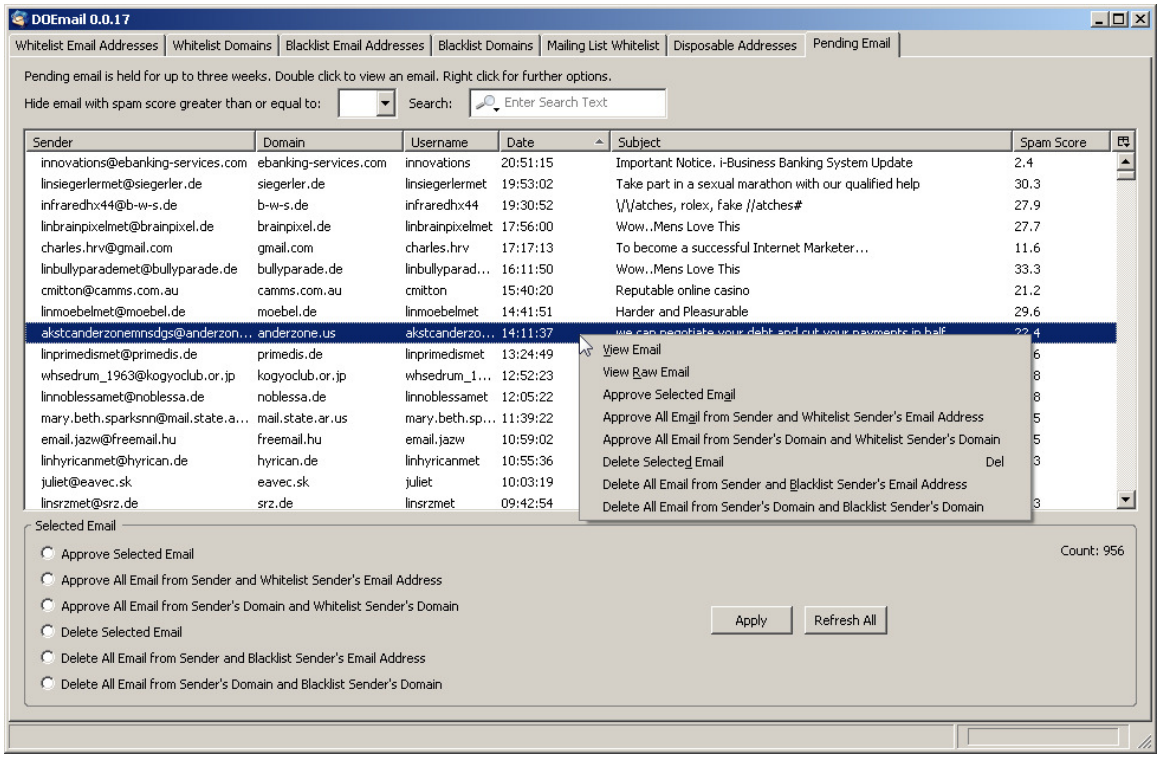


Figure 1.6: Pending email

5. **Dynamic context menus.** Throughout the majority of Thunderbird, dynamic menus were added to right-click context menus where the target of the context menu is an email address. When the menu is triggered, a request is sent to the DOEmail server to determine whether the target email address currently resides in any whitelists or blacklists, and the corresponding menus are built accordingly. An example can be seen in Figure 1.8.

6. **Compose dialog address status.** Graphics similar to those found in the new column on the Thunderbird main dialog have been inserted next to each email address in the compose email window. Each graphic represents the status of the email address, whether it exists within a whitelist or blacklist, or is unknown. As seen in Figure 1.9, right clicking on the graphic brings up a dynamic menu where the email address can then be immediately added to or removed from a whitelist or blacklist.

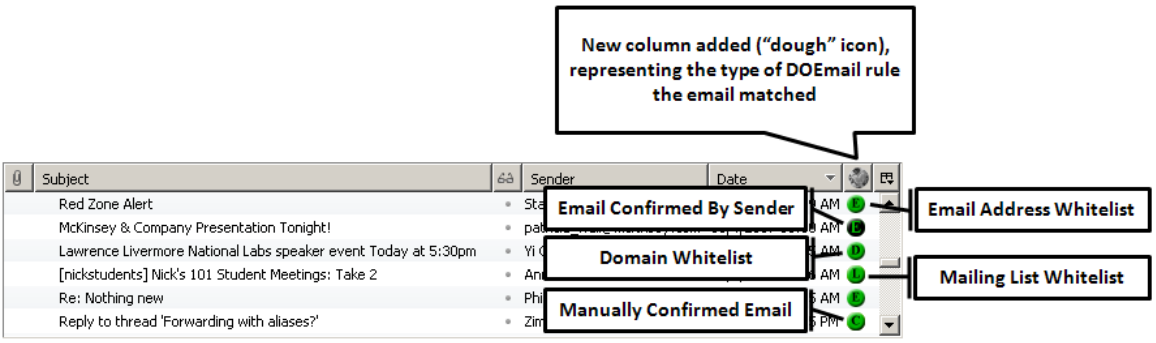


Figure 1.7: DOEmail rule type column

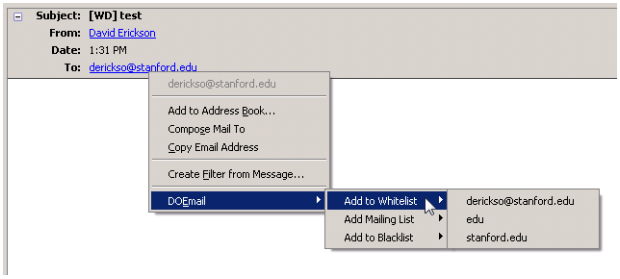


Figure 1.8: Dynamic context menu

The design and testing of all these features was performed very carefully so-as to make it easy for new users to migrate to DOEmail, while showing them the control and power they have over their whitelist and blacklist. Excellent feedback from actual users has also contributed to this process.

1.3.2 Web Interface

The web interface provides similar functionality to the Thunderbird add-on. Users can edit their whitelists and blacklists, and view their pending mail folder. Users can also personalize the text and/or html sections of their outgoing *challenge* email, and ask for a daily summary of pending email.

The web interface displays a variety of usage statistics and graphs, including: the total number of email received by the user per day, the number of emails that matched whitelists and blacklists, how many matched neither and went into the pending folder,

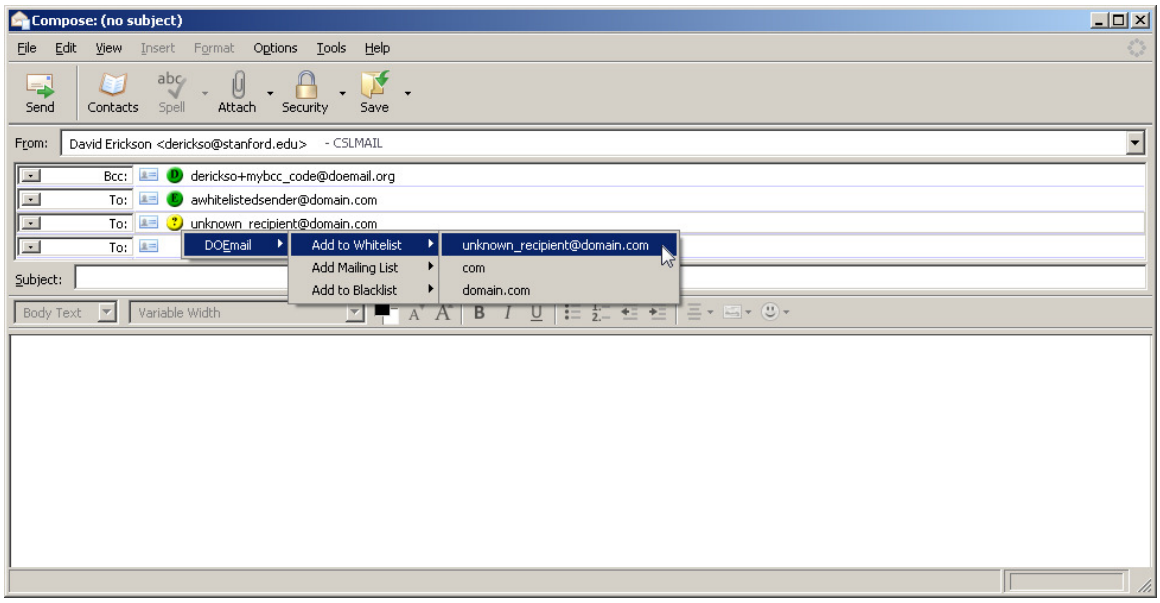


Figure 1.9: Email address status in the Thunderbird compose email window

the number of CAPTCHAs completed by senders, and the number of manually confirmed emails by the recipient. An example user's statistics page can be seen in Figure 1.10.

1.4 System Setup

Figure 1.11 is a high-level overview of the DOEmail system. Users forward email from their existing accounts to their DOEmail address. Incoming mail is handled by Mail Avenger which acts as the incoming SMTP server. Received emails are post-processed by a number of Python scripts which perform the blacklist and whitelist filtering, store pending email locally, and forward accepted email out to user accounts. Integrating with existing email accounts requires processing to determine whether or not incoming email has already been passed through DOEmail.[5]

The Thunderbird add-on is based on JavaScript and XUL [6], and communicates with a PHP service page using HTTPS, and transmits data encoded in JavaScript

[5]This was discovered to be a non-trivial hurdle when migrating new users which is discussed further in section 3.2.

Object Notation (JSON) [1]. The DOEmail servers run an Apache web server with access to the DOEmail configuration state, the pending email list, and the whitelists and blacklists. The web interface is built using PHP.

Incoming email is handled by a single DOEmail server with redundant backup servers for fail-over. If the primary server goes down, a backup server is automatically allocated the active IP. Account information and all whitelists and blacklists are stored in a MySQL database. Pending emails are stored on a filesystem with snapshot support.

1.5 DOEmail Deployment

DOEmail has been operational at Stanford University for nearly two years and has hosted over 120 users. DOEmail has been deployed in the Electrical Engineering and Computer Science departments at Stanford University. Many professors, staff and students actively use it on three large departmental and university email servers, including `stanford.edu`. Nearly 70 users off campus used it too - forwarding their email from a variety of email servers through DOEmail.

DOEmail currently receives around 9,000 emails per day during the week, of which roughly half do not have a corresponding whitelist or blacklist and are thus held pending approval. On average, the system receives 60 challenge responses from senders per day.

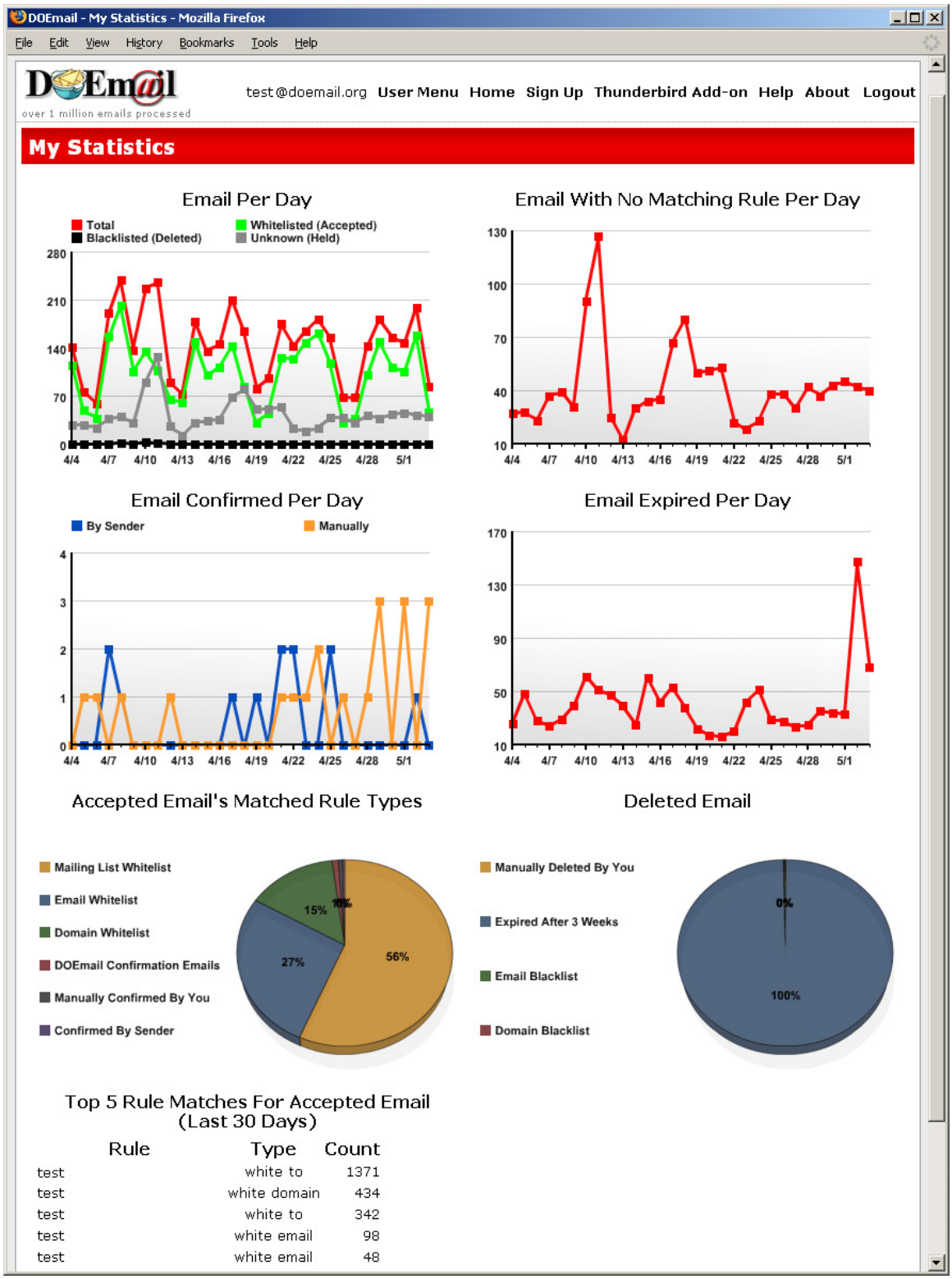


Figure 1.10: Example user's DOEmail statistics webpage

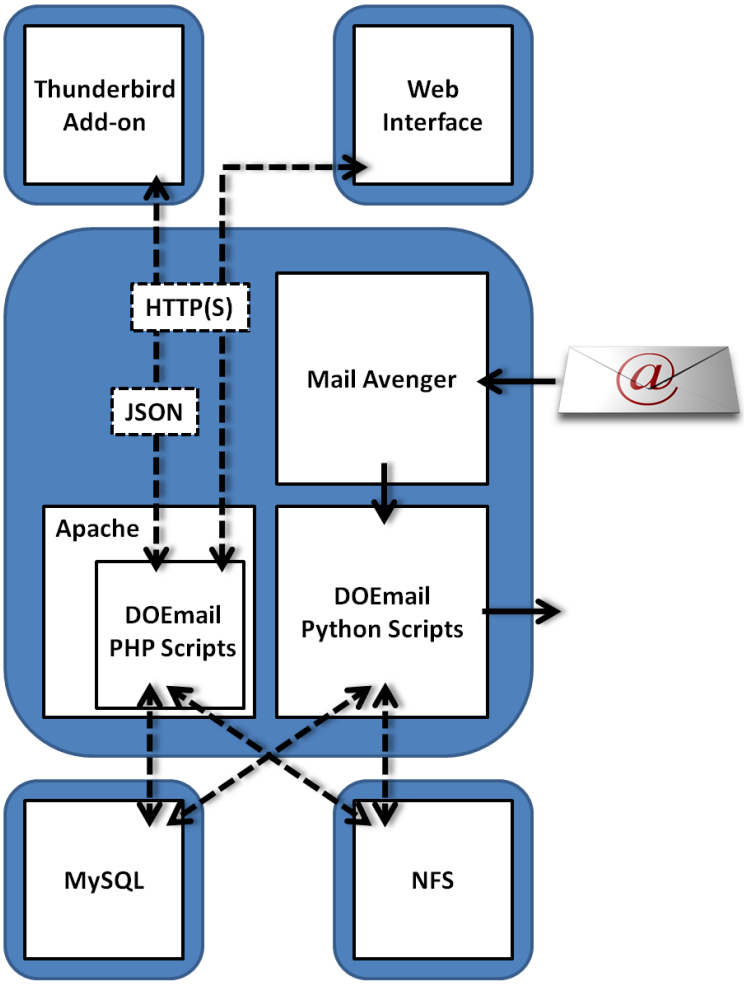


Figure 1.11: System diagram showing how mail flows in and out of DOEmail denoted by solid lines, as well as inter-system communication denoted by dashed lines. Dark squares with rounded corners denote single physical systems.

Chapter 2

Results

Lots of live data were collected with DOEmail from real users, and an attempt is made to try and make sense of it here. However, it is a little hard to make "apples with apples" comparisons, because there is no definitive test of whether an email is really spam or not. In most cases, comparisons are made with the decisions made by SpamAssassin or from hints from what the user does (*e.g.*, if they manually accept an email from their inbox).

2.1 Our Data

The data reported in this section was collected from 07/13/07 through 02/29/08, encompassing a total of 592,794 emails received by 112 user accounts. An overview of the data is presented in table 2.1.

DOEmail logs the headers of all incoming email, all manual edits of whitelists and blacklists, and events involving sender challenges and responses. For comparison, all incoming email was run through SpamAssassin [4][1].

DOEmail divides email into three categories: accepted, deleted, and outstanding. "Accepted" emails are those that matched a whitelist entry, were confirmed by the sender through challenge-response, or confirmed manually by the DOEmail user. 55.55% of email was "accepted".

[1] A very popular content-based spam filter.

	# of emails	% of total
Total Email	592794	
Accepted	329277	55.55
Whitelist	320097	53.40
Sender Confirmed	4382	0.74
Manually Confirmed	4798	0.81
sent challenge	3864	0.65
not sent challenge	934	0.16
Deleted	224320	37.84
Blacklist	84065	14.18
Expired	79297	13.35
Manually Deleted	60958	10.28
Outstanding	39197	6.61

Table 2.1: Number of emails received between 07/13/07 and 02/29/08 categorized by the action taken by DOEmail

During the measurement period, DOEmail deleted just 37.84% of received email – much lower than the 90+% spam reported elsewhere [7, 8]. This is for three reasons. First, users forward email from other systems (*e.g.*, `stanford.edu`, or Gmail), and these sytems already filter some email that is marked as "very high likelihood" of being spam. Second, many of our users are students and have recently created email addresses. It takes time for an email address to be publicized and end up in spam lists. Last, several of DOEmail's users are system administrators who receive large quantities of (non-spam) auto-generated reports from systems they manage.

Email is "Deleted" for one of three reasons: if it is blacklisted, if it expires while on the pending list (the default timeout is three weeks), or manually deleted from the pending list by the user.

When the experiment ended, 6.61% of received email was still sitting in users' pending folders.

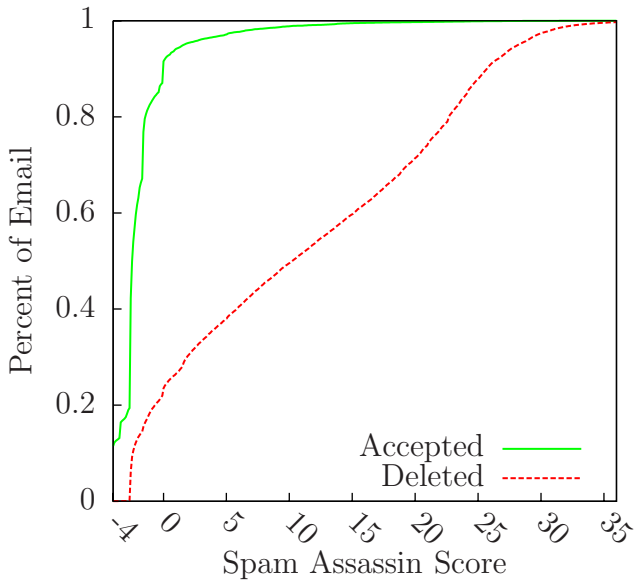


Figure 2.1: CDF plots of the SpamAssassin scores of accepted and deleted email.

2.2 False positives

A false positive means an email was rejected by DOEmail, but should have been accepted. There is no definitive measure of whether or not an email should have been accepted, but it can be estimated. If a user manually accepts an email in their pending folder, it is assumed that it should have been accepted and was therefore a false positive. Our users manually accepted 0.81% of the total received email, which was 1.46% of accepted email.

This is an underestimate of the real false-positive rate: a user might not check the pending folder and miss an email that should have been accepted. However, it was apparent that - particularly when they were new to DOEmail - most users checked their pending folder quite regularly.

2.3 Comparing with SpamAssassin

SpamAssassin calculates a score for each email — the higher the score, the more likely it is spam. If it crosses a threshold it is dropped. As a sanity check that DOEmail's filtering is not way off base, it was important to do a comparison with another existing (albeit imperfect) filtering solution. DOEmail tracked the score SpamAssassin gave to each processed email, and the results are presented in Figure 2.1. The figure shows the CDF of the scores, specifically for two types of email, those that were "Accepted", and those that were ultimately "Deleted".

First look at the "Accepted" curve. As expected, email that makes it to the inbox also has a very low SpamAssassin score.

The "Deleted" curve tells a very different story: almost all scores in the range 0-30 are equally likely, showing there is no good threshold for SpamAssassin to use as a cutoff value, such that email above that value would all be deleted. For example, if email is filtered with a score greater than 2, then 90% of legitimate messages will be accepted, but nearly 30% of spam will be received. If the goal is to only accept 10% of the spam (SA score of -2.4), then the tradeoff will be to only receive 54% of the legitimate email. This quite surprising result illustrates the extent of the continued arms race between spam generators and spam filter creators.

One cannot make a perfect comparison of false negatives with DOEmail, because of a lack of understanding of the user's behavior in relation to email. For example, one might assume that if a user deletes email, it was spam. However it is impossible to actually know the motivation behind why a user deleting that email. One possibility to better understand the false negative rate would be to explicitly prompt the user at deletion time as to whether or not the email was spam, future work in relation to DOEmail may explore this possibility.

However, it should not be surprising that DOEmail has a very low false negative rate: the user is explicitly involved in deciding what email to accept, and whom to accept it from. This is clearly the main advantage of whitelisting. Anecdotally, as a user of DOEmail myself, and closely involved with many other DOEmail users, I can report that DOEmail almost never accepts spam – far less than 1% of my accepted

email was spam, and they are almost always the result of forged source addresses (which as was said earlier, is becoming more difficult to do). Many of our users have reported that DOEmail has saved their inboxes, and that they cannot imagine going back to life without it.

Despite the difficulties in measuring DOEmail's false negative rate, a rough comparison between DOEmail and SpamAssassin can still be drawn. While DOEmail almost never has a false negative, it appears to have about a 1% false positive rate. On the other hand, from our data, SpamAssassin can have 30% false negatives and still have 10% false positives.

DOEmail also makes it very clear that - even to human users - the definition of spam is not always precise. Is a chain letter from a friend spam or just a slight nuisance? When we receive five identical announcements for the same event (*e.g.*, a conference or a talk), are four of them spam? If we are not interested in the event, should all five copies have been identified as spam? The answer is not obvious. Another example of hard-to-classify email - familiar to any faculty member - is the email received from prospective students around the world just prior to a university admissions deadline. If the same email is sent individually to every faculty member, most professors would describe it as spam. If an email is sent to just one faculty member, perhaps it was not spam. Clearly, the definition of spam is context specific. In this particular example it was observed that DOEmail's challenges had the interesting and desirable effect of slightly "raising the bar" for the sender; and in general only the legitimate senders took the time to respond to the challenge.

2.3.1 User Overhead

Managing whitelists A critical element to understanding the viability of a system akin to DOEmail is understanding what degree of burden DOEmail imposes on its users – how much work it takes to maintain whitelists and blacklists. Figure 2.2 shows the average number of times users modified their whitelists and blacklists per day. An action here is defined as adding an entry to, or removing one from a list. The user interface allows users to add and remove multiple entries at a time from whitelists and

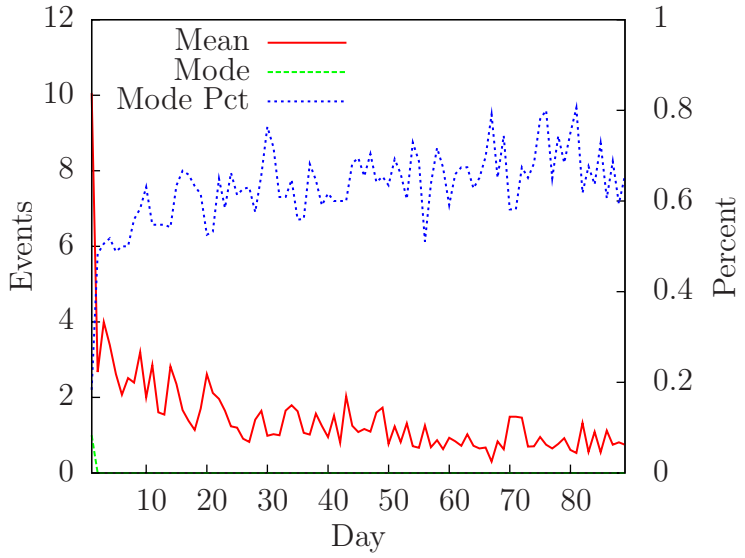


Figure 2.2: Shows the median, mode, and what percentage of users the mode accounts for, of events performed by a user per day after registering for DOEmail

blacklists; this graph considers such operations as a single action. Users are evaluated starting from the day they sign up until they cease being active, or hit the 90 day activity mark.

As expected, maintenance is highest to start with - when users are setting up and tuning their whitelists and blacklists. It is interesting how quickly the maintenance drops. On the first day of use, the average number of actions performed is 10. This decreases to half by the second day. After three weeks users are interacting with the system less than twice per day, and eventually to just once on average.

It was discovered that a minority of users took a very active role in managing their mail accounts and their interactions do slightly inflate the results. On the other hand, the mode indicates that the majority of users don't interact with DOEmail at all on a day-to-day basis. As is shown in the graph, after the first day, over 50% of the users don't perform any manual modifications to their whitelists, and this number increases to greater than 60% over the first two months.

Challenge emails and the pending folder Another important metric for DOEmail is how much effort users devote to managing the pending folder, and its direct correlation being how effective the challenge-response emails are.

Table 2.1 shows that 1.55% of email was accepted, but required action by either the sender or the DOEmail user. If DOEmail *didn't* send challenges, this is the amount of email users would need to manually confirm.

With *challenge* emails, DOEmail users manually confirmed 0.81% of accepted email; *i.e.*, challenge emails roughly halved the number of manual confirmations. This was surprising, it was expected that the challenge emails would reduce the amount of manual work by more than this.

The estimate is that only 53.14% of legitimate senders reply to a challenge. But there is low confidence in the accuracy or meaningfulness of this result – it requires further study. The problem is that one cannot tell why the sender didn't respond. It might be that most of them are legitimate, but are auto-generated senders (*e.g.*, from an online commerce site). Futher work remains to be done in exploring ways to make it easier for users to whitelist auto-generated emails when they sign up for an online service; for example, by adding DOEmail support to Firefox. Other factors contribute to the low response rate: challenges are sometimes filtered as spam before reaching the original sender. As is discussed later, after many iterations of the challenge, DOEmail has not completely avoided being tagged as spam by all major email providers. Another problem is that users are often distrustful of emails containing links. To try and mitigate this fear, DOEmail users can personalize challenges, for example by adding a photograph of the user.

Blacklists It was very surprising that users blacklisted 14.18% of their received email. But on closer inspection 96.4% of all blacklisted email was for one user who was on many unwanted mailing lists. From the user's perspective, blacklisting email addresses and domains to eliminate this email was faster than hunting down and attempting to unsubscribe from all of the disparate sending locations.

If this user is removed from consideration, the remaining users blacklisted less than 1% of their received email.

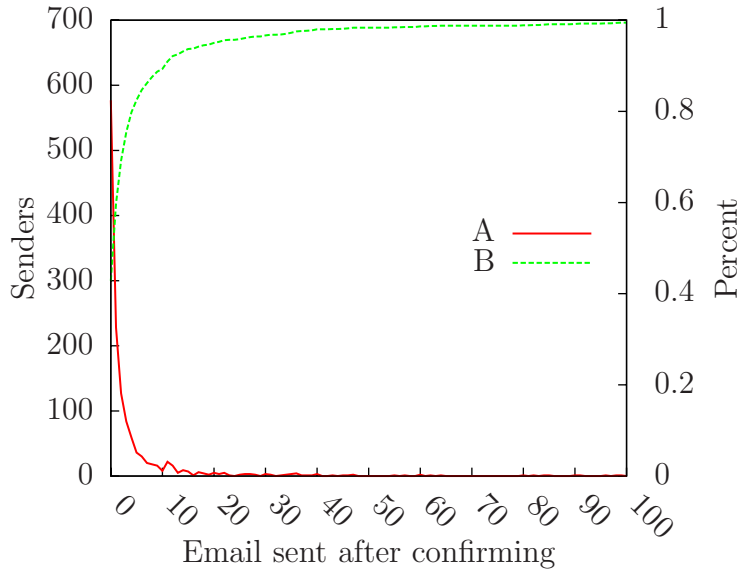


Figure 2.3: The number of senders sending specific quantities of email to a DOEmail recipient for 90 days after completing the challenge response process, along with the cumulative percentage

2.4 Sender Overhead

It is hard to quantify how painful or burdensome DOEmail challenges are. Anecdotally, many responses were heard — some senders don't mind, and are happy to fill out the challenge (once). Some senders simply ignored it. Some were offended to be asked to verify they are humans.

For those senders who completed challenges, DOEmail tracked the number of subsequent emails they sent to the same receiver over a 90 day period after the initial email.

The line labeled A in figure 2.3 shows a curve of the number of senders (left y-axis) and the quantity of email sent (x-axis). The line labeled B is a cumulative percentage of the same values. 43% of senders did not send a single email to the DOEmail user after the initial email. And 75% of senders sent 3 or fewer emails after confirming. For this group the challenges make up a relatively high percentage of the total communication (over 25%).

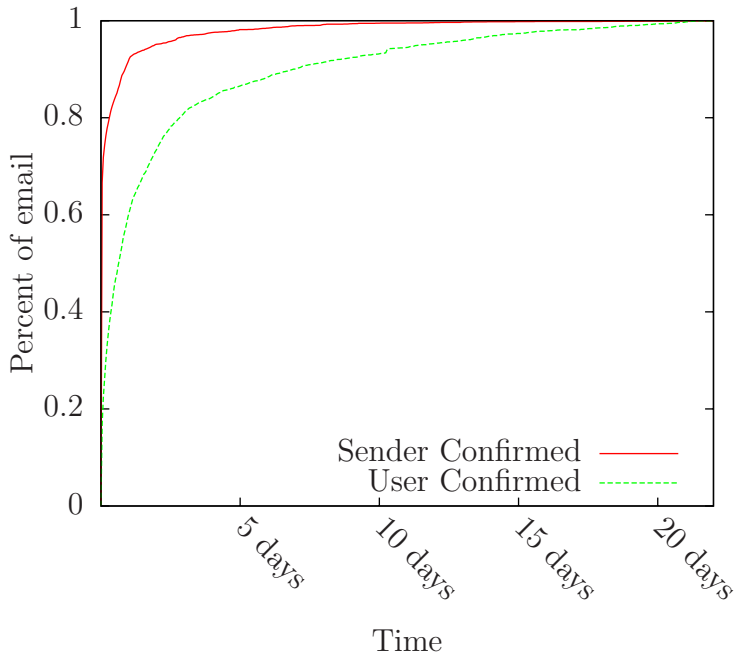


Figure 2.4: CDF plotting how long it takes senders to complete the challenge-response process compared with DOEmail users manually confirming email

But perhaps it shouldn't be surprising – could it mean that users are very good at creating whitelists that cover almost everyone they communicate with frequently? This indeed appears to be the case: only 4.4% of all user/sender pairs who exchanged valid email completed a challenge response. This is encouraging news for whitelisting. It appears that many users whitelist entire trusted domains (*e.g.*, `stanford.edu` or `my_employer.com`), drastically reducing the number of challenges that need to be sent.

2.4.1 Response times

Another interesting aspect to examine is the distribution of time it takes senders to respond to the challenges. Figure 2.4 shows the CDF of the delivery times of email from unknown senders that are eventually confirmed by the sender, or manually confirmed by the user. Over half of the email confirmed by senders occurs in under

15 minutes from the time it was initially delivered to the system, and 95% within the first 24 hours. As expected, the manually confirmed curve is delayed to give the sender time to confirm. However, email is confirmed by DOEmail users as long as the 3 week expiration date, suggesting that retention beyond 3 weeks may be necessary.

Chapter 3

Deployment Experience

Many practical challenges and unforeseen complications were faced while running the operational system. A few of them are listed here.

3.1 Mailing lists

Mailing lists created an interesting problem. DOEmail maintains a separate whitelist for mailing lists; instead of matching on the sender's email address, it matches on the `To` and `CC` fields. This is because emails from mailing lists typically set the `To` or `CC` field equal to the name of the mailing list. If the user is subscribed to the mailing list, it is assumed that all emails from the list should be accepted (unless the sender is blacklisted).

An alternative method would be to whiteliste the List-ID header (recommended by RFC 2919 [10]), but many mailing lists don't yet adhere to the RFC that calls for the List-ID email header.

DOEmail assists users in identifying mailing lists using the following algorithm. If an incoming email doesn't match a blacklist or whitelist, check to see if it contains List-* headers. If it does then extract all email addresses contained in the `From`, `Sender`, `To`, and `CC` header fields. Check if any of these email addresses are contained in the List-* headers. If a match is found, assume it is a mailing list address based on the observation that some mailing lists place their identifiable address into one of

the List-* headers. DOEmail keeps track of all the email addresses that match the List-* header, as well as addresses from emails with no match. The addresses seen most frequently will likely be mailing lists.

Potential mailing list addresses are presented to the user through the web interface. They can then be added to the whitelist, deleted, or ignored.

To avoid creating spam of its own, DOEmail does not send challenge emails to mailing lists.

3.2 Account integration

To attract a broad user base, DOEmail was designed to integrate with existing email accounts. This proved to be much more difficult than anticipated. Novice users had a hard time configuring their existing email accounts to forward email to DOEmail. Of those able to forward, some had mail servers which would rewrite the `From` or `Sender` email header of all forwarded email with the user's email address, behavior that is consistent with forwarding from a mail client, but not a server.

The ability to re-inject DOEmail filtered email back into the user's original email account turned out to be prohibitively difficult. Supporting this required the user's server to differentiate between email received for the first time, and email that has been processed by DOEmail. I have been unable to identify a free major email provider that can enable this in a clean way[1].

At Stanford this turned out to be much easier, the main stanford.edu domain forwards email to a final destination server, which enabled DOEmail to be interposed between the two. On other Stanford systems specially configured email addresses were used (eg. derickso+clean@adomain.com) that when sent to, would deliver cleaned email directly to the inbox, meanwhile all other email is forwarded to DOEmail. As a last resort it is possible to use two email accounts, one for receiving and forwarding to DOEmail, and one to receive all the clean email, although this is a cumbersome

[1] It is possible to do this with Google's Gmail, however it involved inserting a character string into each email's subject that could 'tag' it as having been through DOEmail, however the feeling was that this is too intrusive a solution to be viable.

solution.

3.3 Sender challenges

Challenges had to be crafted so that they wouldn't be tagged or dropped by the sender's spam filter. Multiple techniques were implemented to reduce the chance this would happen. DOEmail's outgoing challenges are composed of a text-only section of the email, along with an optional (on by default) HTML portion. Images are embedded in the email itself to decrease the likelihood that the email would be considered spam due to linking to external images. Also, an SPF entry for our domain was created, specifying which IP addresses are approved for sending email from doemail.org. Lastly DOEmail hashes and signs all outgoing challenges with DomainKeys signatures. Despite this it appears some major providers (*e.g.*, Yahoo!) classify our challenges as spam.

3.4 Performance

Anyone looking to implement a system such as DOEmail should rightly wonder what the performance characteristics of such a system would be. DOEmail's core processing performance depends heavily on the latency of requesting and receiving information from an external SQL server which is used for storing and querying users' state. Further, for incoming email that needs to be written to the pending folder, DOEmail will depend on the performance of the file system being used to store such email. Given this information I do not attempt to provide a full treatment of DOEmail's performance in this text, suffice it to say that DOEmail has received no performance tuning, and yet in practice we find that DOEmail can process email approximately 40% faster than our Spam Assassin installation which is configured to run locally in a client/daemon configuration.

Chapter 4

Limitations

Backscatter Challenges sent in reply to spam containing forged sender addresses are considered "backscatter". Backscatter is often considered as contributing to the spam problem because it can be sent to users that never originated an email, potentially causing a flurry of failed delivery status notifications to be sent if the email address was invalid.

I believe this problem can be overcome through wide-spread adoption of sender authentication. If a domain advertises its use of sender authentication, and an incoming email purports to be from the domain, but cannot be verified, then the system will not send a challenge email to the spoofed sender address. Thus it is in the domain administrator's best interests to deploy sender authentication to protect themselves from backscatter.

Automated senders Email sent from automated senders such as registration confirmations or newsletters cannot be confirmed through challenge-response. As mentioned earlier, DOEmail supports "disposable" addresses that could be used when the sender's address is unknown. Other methods of handling automated senders involve pre-whitelisting an entire domain, or waiting for the email to arrive and manually confirming it from the pending email.

Going forward I have high hopes that as whitelisting becomes more common, websites will be more explicit in listing the email addresses they plan to send to you

from, allowing users to pre-whitelist the addresses. Since beginning work on this project I have seen a large increase in the frequency of such disclosures.

Challenge collision Multiple users behind different whitelisting services can complicate the use of challenges. In the worst case, the original sender of the email will have to comb their pending list for the challenge sent by the recipient[1]. One approach to addressing this is for whitelisting services to standardize around sending challenges from a known address or domain (with source authentication). This way, senders can auto-whitelist the address or domain when sending email, in expectation of receiving a challenge. Another approach is to create a challenge-response authority which provides the credentials to allow senders to sign "legitimate" challenges and for receivers to verify them. Neither of these approaches are particularly elegant and I believe this to be a interesting area for future work.

Mechanical turk attacks In the context of this paper, a mechanical turk attack would strive to harness large numbers of users to solve the CAPTCHAs presented by the whitelisting system. Generally the users doing the work are "paid" via access to online content. While this may pose a real threat for CAPTCHAs used to protect sending accounts (from which a spammer may send millions of messages), it doesn't appear to be a realistic attack to gain access to the inbox of a single user. Because users may immediately blacklist the offending address, the cost to the attackers converges on answering one CAPTCHA per email.

[1]Note that there isn't danger of a loop as challenges are sent from addresses that will not issue a reply

Chapter 5

Related work

There have been numerous proposals for combating spam (and a relative paucity of user based studies). Related mechanisms are briefly discussed below.

Leveraging existing social networks A number of projects have proposed leveraging existing social networks to help create email whitelists. Re:[11] proposes a privacy preserving approach to using friend-of-friend social networks to expand a single user's whitelist. Johansen *et al.* [12] examine algorithms for identifying communities of interest (akin to social networks) from existing email traffic patterns. This can be used to aid in whitelist generation or identify networks likely to send spam. In [9], Boykin *et al.* use graph-theoretic methods to analyze email headers. The resulting data is used to create white, black, and gray lists based on the characteristics of existing email networks.

Sender reputation Taylor [19] explores the use of sender reputation in the Gmail email service. Sender reputation allows Gmail to classify most senders as good or bad, remaining email is run through traditional statistical classification. Sender reputation could provide a useful input to user (or administrator) automated white and blacklist entries.

Changing the cost model for senders Another class of proposals attempts to alter the current cost model of sending spam in which the marginal overhead of sending an additional email converges on zero. The Penny Black Project [5] suggests charging a small fee for every email. This would not effect low volume senders such as users, but could greatly impact spammers who rely on large volumes (and consequently list managers as well unless special measures are taken). Walfish *et al.* [21] propose limited quotas per sender and offers a scalable mechanism for enforcing it.

Challenge-Response for whitelisting Many open source and commercial anti-spam systems contain elements found in DOEmail such as whitelists, blacklists, and challenge-response. Examples from the open source community include, TMDA (Tagged Message Delivery Agent) [13], Active Spam Killer (ASK) [15], and Qconfirm [16]. There are also a number of commercial offerings such as Spam Arrest [3] and Clean My Mailbox [17].

Chapter 6

Conclusions

In the past, whitelisting probably didn't make sense, because it was so easy to spoof sending addresses. Nowadays, spoofing is much harder to do - and will continue to be so - making whitelisting an interesting option. This study suggests that whitelisting can be very effective, and significantly outperforms SpamAssassin in both false negatives and false positives. It is interesting to note that this is achievable on a simple prototype system with significantly less engineering effort than is devoted to creation of spam filters. But this shouldn't be surprising: like a buddy-list in IM, a whitelist tries to precisely identify the people we communicate with, or who we allow to send us email. Unless we make a mistake, we will not allow a spammer to send us email. One should expect a well-engineered whitelisting email service to behave almost perfectly.

But the controversy with whitelisting is not the performance, but the overhead imposed on the users. In our system both the sender and the receiver are involved in maintaining the user's whitelists. This study has mixed and inconclusive results about how much burden this system placed on the sender. Challenge-response emails were used to try and separate legitimate users from spam-generating computers. As many as 50% of the challenges to legitimate senders received no response; but it is suspected that this is because the sender was in fact a legitimate computer (not a human) – *e.g.*, an online commerce site. This can likely be solved through a better user interface allowing users to whitelist email addresses when they register for an

online service. On the other hand, this data indicates that the burden on the user (the receiver of email) is very low, with most users doing no manual maintenance after the first few days. A good user-interface is the key to low maintenance; this system, implemented by just me, placed very little burden on the user. In the future, you can expect that a finely-tuned whitelisting systems (based on more experience) will require even less maintenance.

In summary, while there is still more work to be done in this area, whitelisting deserves much closer attention than it has traditionally received in the past.

Bibliography

[1] JSON: JavaScript Object Notation. http://www.json.org/.

[2] reCAPTCHA. http://recaptcha.net/.

[3] Spam Arrest. http://www.spamarrest.com.

[4] SpamAssassin. http://spamassassin.apache.org/.

[5] The Penny Black Project. http://research.microsoft.com/research/sv/PennyBlack/.

[6] XML User Interface Language (XUL). http://www.mozilla.org/projects/xul/.

[7] Barracuda Networks. Annual Spam Report. http://www.barracudanetworks.com/ns/news_and_events/index.php?nid=232, December 2007.

[8] Commtouch Software. Q4 2007 Email Threats Trend Report. http://www.commtouch.com/downloads/Commtouch_2007_Q4_Email_Threats.pdf, January 2008.

[9] P. O. Boykin and V. P. Roychowdhury. *Leveraging Social Networks to Fight Spam*, volume 38(4):61-68. Computer, 2005.

[10] R. Chandhok and G. Wenger. RFC2919 - List-Id: A Structured Field and Namespace for the Identification of Mailing Lists. http://www.ietf.org/rfc/rfc2919.txt, 2001.

[11] S. Garriss, M. Kaminsky, M. J. Freedman, B. Karp, D. Mazieres, and H. Yu. Re: Reliable Email. In *NSDI*, San Jose, California, May 2006.

[12] L. Johansen, M. Rowell, K. Butler, and P. McDaniel. Email Communities of Interest. In *Conference on Email and Anti-Spam*, Mountain View, California, August 2007.

[13] J. R. Mastaler. Tagged message delivery agent. http://www.tmda.net.

[14] D. Mazieres. Mail Avenger. http://www.mailavenger.org/.

[15] M. Paganini. ASK: active spam killer. In *USENIX Annual Technical Conference*, pages 12–12, Berkeley, CA, USA, 2003. USENIX Association.

[16] G. Pape. Qconfirm. http://smarden.org/qconfirm/.

[17] Permission Technologies. CleanMyMailbox - Permission-Based Filtering Service. http://www.cleanmymailbox.com.

[18] R. G. Smith, M. N. Holmes, and P. Kaufman. *No. 121: Nigerian advance fee fraud*. Trends & Issues in Crime and Criminal Justice, July 1999.

[19] B. Taylor. Sender Reputation in a Large Webmail Service. In *Conference on Email and Anti-Spam*, Mountain View, California, July 2006.

[20] L. von Ahn, M. Blum, N. Hopper, and J. Langford. CAPTCHA: Using hard AI problems for security. In *Proceedings of Eurocrypt*, pages 294–311, 2003.

[21] M. Walfish, J. Zamfirescu, H. Balakrishnan, D. Karger, and S. Shenker. Distributed Quota Enforcement for Spam Control. In *3rd USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA, May 2006.