CentMail: Rate Limiting via Certified Micro-Donations

Sharad Goel Jake Hofman John Langford

David M. Pennock Daniel M. Reeves

{goel, hofman, jl, pennockd, dreeves}@yahoo-inc.com
Yahoo! Research
111 W. 40th Street, 17th Floor
New York, NY 10018

ABSTRACT

We present a plausible path toward adoption of email postage stamps—an oft-cited method for fighting spam—along with a protocol and a prototype implementation. In the standard approach, neither senders nor recipients gain by joining unilaterally, and senders lose money. Our system, called CentMail, begins as a charity fund-raising tool: Users donate $0.01 to a charity of their choice for each email they send. The user benefits by helping a cause, promoting it to friends, and potentially attracting matching donations, often at no additional cost beyond what they planned to donate anyway. Charitable organizations benefit and so may appeal to their members to join. The sender's email client certifies each outgoing message with an unforgeable stamp issued by the CentMail server. The recipient's email client verifies with CentMail that messages are appropriately stamped, and have not been queried by an unexpectedly large number of other recipients. More generally, the system can serve to rate-limit and validate many types of transactions, broadly construed, from weblog comments to web links to account creation.

1. INTRODUCTION

The Internet has reduced the cost of communication to near zero, benefiting billions of people around the world. One consequence, however, is that unsolicited and unaccountable commercial communication, or *spam*, is also sent indiscriminately in massive quantities at low cost, imposing a large burden on recipients and on systems. Spammers have infiltrated nearly every form of online communication, including email, instant messaging, blog comments/trackbacks, and web pages/links. We propose a system for rate limiting Internet communications broadly, emphasizing the case of email, the first and still most widespread form of online communication.

Many people, most notably Bill Gates [13], have observed that adding a modest cost to sending email, for example by requiring postage stamps like ordinary mail, could significantly deter spam. Researchers have proposed and analyzed several such systems, including variations where the recipient keeps the payment, the recipient has the option of either keeping or refunding the payment [14], the sender "burns" human time or CPU cycles [3, 9], or the sender pays to a charity of his or the recipient's choice [5].

Although an equilibrium where senders and receivers all adopt email stamps benefits nearly everyone—and in fact can lead to a higher social optimum than is possible even with a perfect automated spam filter [14]—there is a serious *flag day* problem,[1] or coordination failure, that makes the equilibrium hard to reach from the status quo. Senders do not want to spend money buying stamps if recipients are not checking stamps, and recipients would not bother to check stamps if few senders use them. The hurdle for senders is heightened by the very real possibility that spammers who already hijack other people's computers may now in addition drain the users' stamp accounts of money to send spam or, worse, to funnel the money to themselves. It seems that the prospect of some day reducing spam is not enough to convince a critical mass of both senders and recipients to adopt a new protocol and monetary accounting infrastructure.

Most economic anti-spam approaches focus on penalizing senders, rewarding recipients, or both. This makes economic sense, in effect repaying the recipient's time that senders (including spammers) impose upon. However this leaves senders with little or no reason to join and plenty of reason not to, especially with the prospect of black hats stealing their funds.

Figure 1 CentMail stamps appear as an email signature that promotes the sender's cause.

```
--
Alice certified this email by donating $0.01 to Sierra Club
This donation was matched by Bob's Widgets
Powered by CentMail.net -- Do good. Fight spam.
http://centmail.net/stamp/1234567890AbCdEf
```

Instead, we advocate the "sender pays charity" variant [5] that directly rewards senders regardless of what recipients do. CentMail begins as a charity fund-raising tool allowing senders to regularly donate to causes they care about and to promote those causes in their email signatures (Figure 1). Charities may encourage their members to join and companies may agree to fully or partially match senders' donations,[2] increasing the unilateral benefit to senders. Senders

Submitted to CEAS 2009 - Sixth Conference on Email and Anti-Spam July 16-17, 2009, Mountain View, California USA [version 2009-04-16 11:58]

[1] `http://en.wikipedia.org/wiki/Flag_day_(software)`

[2] Although users may attempt to defraud sponsors by effec-

who already donate to the charities represented in CentMail can, up to a point, buy stamps at no cost by diverting funds that they already allocated for donation. Indeed, in the United States 89% of households make annual donations, on average $1620 [4] (with the median on the order of $100).

Once senders join, recipients may begin to use stamps as an additional feature for spam filtering, presumably whitelisting any message above a threshold donation level per authenticated recipient. Any improvement in the ability to recognize good email allows for more aggressive filtering of spam email which, in turn, increases the incentive to join CentMail, in a virtuous cycle.

A similar and independent effort at IBM Research in 2004 aimed to promote "charity seals" in email [5]. To our knowledge, while the concept was previously discussed [17], a formal protocol was never developed, nor a working system implemented. Our main contribution is to make the idea concrete by defining a formal protocol, implementing a working prototype of the service, and analyzing the benefits and drawbacks of the approach and how we envision the service's promotion and adoption. In addition we are actively pursuing the adoption of CentMail within Yahoo!, the world's largest email provider.

In the next section, we provide background and survey related work. Section 2 describes the CentMail protocol in detail for email and in general for any online electronic "document," including blog comments and links. Section 3 discusses the formal guarantees of CentMail, and presents our answers to a number of common questions and potential criticisms.

1.1 Background and Related Work

Simple Mail Transfer Protocol

The backbone of the email system is the Simple Mail Transfer Protocol (SMTP), the de facto standard for one-to-one electronic messaging. SMTP operates independently of the message contents, which consists of the body of the email and the header fields (e.g., `From`, `To`, `Cc`, `Subject`, etc.) Senders connect to their local, outgoing SMTP server, specify a list of recipients—which need not be those listed in the email header—and then submit the message contents. The outgoing SMTP server looks up the domain of each recipient in the Domain Name System (DNS), connects to the appropriate receiving SMTP servers, and transmits the message.[3] SMTP does not provide any guarantees regarding the authenticity of the contents or the sender. In practice it is straightforward to forge the header fields to give the appearance, for example, that a message was sent from another individual—a common tactic of spammers. The simplicity of the underlying mail protocol, while providing for a great deal of flexibility, has also opened the door to the proliferation of spam.

Approaches to Mitigating Spam

Many anti-spam techniques, both economic and otherwise, have been proposed [7], including dozens in the annual *Conference on Email and Anti-Spam* (`ceas.cc`) alone. We detail some of these approaches below.

Domain Filtering. The most straightforward and computationally inexpensive approach for detecting spam is to blacklist known offending domains while whitelisting reputable ones. One problem with this technique, however, is that it is often difficult to verify the IP address or domain name of the originating email server. DomainKeys Identified Mail (DKIM)[4] is a protocol for digitally signing emails which attempts to remedy this shortcoming. Each participating domain is first assigned a public/private RSA key pair [16]. To sign an email, the sender's domain encrypts a hash of the email contents (the standard header fields and the body) with its private key, and includes this signature in the *DKIM-Signature* header field. Recipients verify the signature by decrypting the included signature with the domain's public key and confirming that it matches the message hash. In particular, DKIM provides two cryptographic guarantees: The contents of the email have not been altered in transit; and the email was in fact sent from the domain it was claimed to have been sent from. While DKIM facilitates domain-based filtering, it does not directly prevent spam or other abusive behavior.

A related filtering technique is the Sender Policy Framework[5] (SPF), which allows recipients to determine if messages were sent from machines authorized to use a particular domain name, and consequently makes it more difficult to forge sender addresses. For example, `mydomain.org` would publish a list of machines authorized to transmit email whose sender email address ends with "`@mydomain.org`." A recipient can then check that a message purporting to be from `alice@mydomain.org` was in fact sent by an authorized machine.

Content Filtering. Content-based filtering of email works by scanning the body and headers of a message for features, usually textual, that are predictive of spam. Like domain filtering, content filtering has the advantage of requiring little input from end-users, generally running quietly behind the scenes. However, this approach is prone to a constant cat-and-mouse game, with spammers regularly modifying their messages to evade filters. One of the most common implementations of this approach is the open source project SpamAssassin,[6] which in turn has been incorporated into several commercial and non-commercial spam solutions. SpamAssassin tests each incoming message against hundreds of rules. For example, one rule tests for the presence of the word 'viagra', and another tests whether the recipient list is sorted by address, indicative of a mass-mailing. Each rule is associated with a score—which can be either positive or negative—and a message's cumulative spam rating is the sum of the scores for the rules which are satisfied by the message. Typically, a message is marked as spam only if it has been flagged by several different rules.

tively stamping fake messages, we think the likelihood of and potential damage from this type of behavior are small and can often be detected.

[3] Although senders can specify a list of recipients when talking to their local, outgoing SMTP server, this outgoing server initiates separate connections for each receiving domain. Hence, SMTP is ultimately a one-to-one rather than a broadcast protocol.

[4] `http://www.dkim.org/`

[5] `http://www.openspf.org/`

[6] `http://spamassassin.apache.org/`

Economic Approaches. Content and domain filtering are by far the most widely used anti-spam techniques. However, several alternative, fundamentally economic, approaches have also been proposed. The key idea in these proposals is to require that users expend a certain amount of effort for each email they send, effectively limiting the rate at which users can send messages. Here effort can, for example, correspond to CPU cycles [3, 9], memory cycles [2, 8], "human cycles" (e.g., solving a CAPTCHA [18, 19]), or monetary payments (e.g., Goodmail[7]). Microsoft's Penny Black project [1], named in honor of the world's first official adhesive postage stamp, is a general protocol for issuing and validating certified stamps that serve as proof of the sender's effort. One delicate issue with these systems is calibrating payments so that spammers are deterred but legitimate users are not overburdened. Spammers, for example, often have significant computational resources at their disposal via compromised *zombie* computers, and can consequently afford to "pay" hefty fees.

One implementation of a proof-of-work approach for authenticating emails is Hashcash [3], where users pay in CPU time. For each recipient of an email, the sender computes a stamp of the form `version:date:recipient:salt` where `salt` is chosen such that the first k hexadecimal digits of the SHA-1 hash[8] of the stamp are 0. For any given value of `salt`, the first k digits of the hash is one of 16^k possible strings, and the most efficient way known to calculate a satisfying salt is comparable to random trial-and-error;[9] hence the sender is expected to make 16^k attempts to generate a legitimate stamp. The sender inserts the stamp in the header of the message, for example:

```
X-Hashcash: 0:080626:bob@somewhere.org:6470e06d773e05a8
```

While generating the stamp is computationally difficult, verifying the stamp is easy: The recipient simply hashes the stamp and checks that the first k digits are 0. To prevent reuse of stamps, recipients store recently received stamps and reject duplicates.

There are also several *challenge-response* spam filtering implementations that require payment in terms of human time, for example Default Off Email (`doemail.org`). Generally, a recipient challenges unknown senders to solve a CAPTCHA test [18, 19] that is difficult for computers but relatively easy for humans.

A serious downside of proof-of-work is the sheer inefficiency. Human time and CPU cycles are valuable resources. This problem is not inherent to economic approaches, however. ReCAPTCHA [20], for example, improves the process of digitizing books by challenging humans to decipher text that could not be read by optical character recognition software. An interesting twist on sender-payments are sender-posted bonds [11, 14]. Senders post a bond for each email they send that is redeemable by the recipient in the case of spam. In this way, while offering to make a payment, legitimate senders never in fact have to pay the recipient. Bonds are essentially warranties issued by the sender that the recipient will not be dissatisfied.

Several companies facilitate the economic metering of email. Goodmail[7] allows legitimate bulk emailers to certify their mail, enabling "trusted class" treatment by major email providers including AOL and Yahoo!. Seriosity[10] offers an email add-on for enterprise use that allows senders to attach a virtual currency called Serios to their email to signal relative importance. Boxbe[11] bundles prioritization tools, smart whitelisting, and challenge-response functionality for use in Outlook, Yahoo! Mail, or Gmail.

The Charity Seals project at IBM [5] is the closest to our own. The authors proposed a similar system where email senders donate to charity. We present a formal protocol, implement a working prototype, and analyze the prospects of the concept in detail.

2. CENTMAIL

Domain and content filtering are currently the first line of defense against spam. However, domain filtering is difficult to apply when spammers use legitimate domains to send messages (e.g., by opening email accounts at Yahoo! and Gmail), and places an onerous burden on new domains to establish themselves as legitimate senders. Content filtering requires considerable effort to maintain as spammers constantly evolve to circumvent the latest filters. In this regard, economic approaches to spam are an attractive alternative and offer the possibility of an elegant solution. Despite being proposed several times over the last decade, however, economic tactics to fighting spam have failed to gain popularity. In large part, this is because economic approaches often lack appropriate incentives. Early adopters incur costs for participating in those systems yet see no initial benefits: End-users do not see a reduction in spam until a significant fraction of other users participate in the system, posing a substantial coordination problem. As one illustration of the severity of this chicken-and-egg problem, we note that DomainKeys, an open standard that requires little cooperation from end-users, is still not universally supported even after several years of industry backing by Yahoo! and Google. While the equilibrium in which all users have adopted the economic approach is certainly a strong deterrent against spam, the path toward this equilibrium appears formidable.

CentMail is a general economic framework for rate-limiting that directly addresses the issue of incentives, providing tangible net benefit to even the earliest adopters with no need for coordination. This tailoring of incentives is threefold. First, users receive stamps in exchange for making donations to charitable organizations of their choice. In this way, many users would incur no added financial burden since they already make these donations regardless of their participation in CentMail. In fact, 89% of U.S. households already make annual donations, with an average household contribution of \$1620 (or 3.1% of income) [4] and a median on the order of \$100. (A donation of \$100 yields enough stamps to send email to 10,000 people.) Second, the stamps themselves are implemented as email signatures (see Figure 1) that promote both the sender's cause and the sender's support of that cause. Demand for such a feature is evidenced by the nearly 20 million people who use the Facebook application *Causes* to advertise their favorite charitable organiza-

[7] `http://www.goodmailsystems.com/`

[8] The SHA hash functions are a set of cryptographic hash functions designed by the National Security Agency (NSA).

[9] Although recent work has shown it is possible to find collisions faster than through trial-and-error [21], an exponential number of attempts is still required.

[10] `http://seriosity.com`

[11] `http://boxbe.com`

tions. Moreover, a recent survey estimates that "people are 100 times more likely to donate when asked by a friend or family member than an anonymous solicitation" [6]. Third, users amplify their impact via matching donations, either by a corporate sponsor acknowledged in the message signature, or by the mail provider who may eventually see a reduction in spam-associated costs, estimated to be on the order of billions of dollars per year worldwide [12]. For many potential users, these design choices in sum yield net benefit, even in the absence of other participants in the system.

Although CentMail offers benefits even for early adopters, wider coordination is still required for it to function as a spam deterrent. We note that in this regard, CentMail improves upon existing proposals in that CentMail stamps serve as advertisements for the system itself. This allows us to leverage the latent social network of email contacts to further adoption of the system. If enough senders join, recipients may take notice and begin to whitelist stamped email, allowing them to tune their content-based filters more aggressively, increasing the incentive for senders to stamp email, and so on.

We do not expect to see complete adoption of CentMail. However, even with limited adoption, we believe stamping to be an effective tool that works in conjunction with aggressive domain and content-based filtering to detect and deter spam.

2.1 The CentMail Protocol

The CentMail protocol supports authentication of both emails and arbitrary text documents. For example, a "document" could correspond to a comment on a weblog, or a listing of links on a web page. In each case, CentMail certifies that the content was validated by a charitable donation.

The two key operations are certification and verification:

```
Centmail.certify(amount, digest)
    return null

Centmail.verify(digest)
    return {amount, queries}
```

These function calls are authenticated, and in particular, the user making the call is identified by the global parameter `Centmail.user`.

The `certify` function takes as input an `amount` to donate and the `digest`, or SHA-1 hash, of the content.[12] It is generally in the sender's interest to append a *nonce* (i.e., a randomly generated string) to their content to ensure each message is unique—although this is not explicitly required by the protocol. When a message is certified, the CentMail server debits `CentMail.user` and stores the `digest` for later verification. In order to efficiently scale, message digests are eventually expired, and hence are maintained on the CentMail server only temporarily.

To verify content, the user passes the document `digest` to `Centmail.verify`. This call returns the `amount` which was donated, and `queries`, the number of times the content has been verified. The return value `queries` is a crucial piece of information since the certifier's "payment" (i.e., donation) is less meaningful when the content is consumed by multiple individuals. For example, in the case of email, donating $0.01 for an email which is ultimately sent to 1000 people is less of a commitment than donating $0.01 for an email sent to a single individual.[13] Typically, recipients would accept messages when `amount/queries` is at least $0.01, and treat messages not meeting this threshold as effectively unstamped. In this latter case of unstamped—or effectively unstamped—messages, existing domain-based and content-based techniques could still be applied to classify email. The recipient, however, is free to enforce any filtering policy of their choice.

A complete description of the protocol is given in the Appendix.

2.2 General Rate Limiting

Aside from certifying messages, it is often useful for applications to rate limit requests, for example for account creation, or for posting comments to blogs. CAPTCHAs are typically used in these contexts, and work by requiring users to burn "human cycles." CentMail facilitates an alternative, economic approach to rate limiting which allows third parties to ask users to burn (i.e., donate) money. This feature is intended for web-based applications, and is implemented through an additional function call:

```
Centmail.request(amount, transID, returnURL)
    return null
```

When the requester makes this call, the end user (i.e., the individual being asked to make a donation) is redirected to the CentMail website to confirm the donation. Afterward, CentMail posts an authenticated response to `returnURL` and redirects the user back to the originating site. The requester receives confirmation that a donation was made, but no other identifiable information about the user. A more detailed protocol description is given in the Appendix.

Web links are another form of online communication beset by spam that may benefit from CentMail certification. For example, blog posts commonly feature a list of *trackbacks*, or the context of what other blogs that link to the post are saying. Spammers will flood a blog with a deluge of bogus links in the hopes of appearing in its trackback section. Spammers also manufacture or buy links to manipulate their search engine ranking; search engines cannot easily distinguish "natural" links from paid or spam links, except for the crude `NOFOLLOW` link attribute.

Web content creators could certify a link by donating money in association with a particular pair of source and destination URLs. Blog owners could whitelist certified trackbacks and search engines could use the certification as a quantitative measure of how much trust to place in the link.

3. ANALYSIS OF CENTMAIL

3.1 Correctness Properties

[12] In the case of a plain text document, computing the message hash is straightforward. For email, however, care must be taken so that determining which header fields to include, and their order, is unambiguous.

[13] Often one can avoid the problem of multiple recipients consuming the same content by using different nonces for each recipient. Then instead of sending 1000 people the "same" certified email message, each recipient would in fact be verifying their own unique copy.

The CentMail protocol provides two guarantees: 1) a stamp issued for one message cannot be applied to another; and 2) recipients will accept a message if and only if the sender has made a donation that exceeds a certain minimum threshold per recipient. These guarantees, explained in more detail below, rely on the standard assumption that the underlying hash function (e.g., SHA-1) is *collision resistant.*

Definition 1. *A hash function h is collision resistant if it is computationally difficult to find two messages m_1 and m_2 (with $m_1 \neq m_2$) such that $h(m_1) = h(m_2)$.*

Since CentMail stamps are indexed by the hash of the associated message m_1, certifying another message $m_2 \neq m_1$ with the same stamp requires $h(m_2) = h(m_1)$. By collision resistance, it is computationally difficult to find such a message, establishing the first property.

Property 1. *A stamp issued for a document m_1 cannot be applied to another document $m_2 \neq m_1$.*

Under the CentMail protocol described in Section 2.1, malicious users could in theory sniff a network and verify messages not intended for them, effectively depleting the value of a legitimate stamp. In this scenario, while the malicious user does not realize direct monetary benefit, the tactic could be used to undermine the CentMail system itself by making legitimately stamped mail indistinguishable from mail stamped by a spammer with reused stamps. We view this to be an unlikely situation, and further note that only authenticated users can increase `queries`, so abuse is likely detectable. However, to provide a stronger correctness guarantee, we modify the CentMail protocol so that senders now specify a list of intended recipients, and recipients, upon verification, are told whether or not they are on that list. If they are, then `queries` is incremented and the recipient can safely accept a message if `amount/queries` exceeds a minimum threshold; otherwise `queries` is left unchanged and the recipient should consider the message as effectively unstamped.

In particular, suppose there is a universally agreed upon threshold $t > 0$ on `amount/queries` for accepting a message. As long as the sender donates $t \cdot n$, where n is the number of intended recipients, then all the intended recipients will accept. This is because `queries` is only incremented for intended recipients, and the last intended recipient will see $t \cdot n$ as the amount (this is constant for all recipients) and n as the number of queries; thus `amount/queries` will meet the threshold t. (All previous recipients will see `amount/queries` safely exceeds t.) For completeness, we provide a rigorous statement of this property.

Property 2. *Fix $t > 0$, and suppose that each message m is associated with a set of intended recipients R_m. Call a user u* compliant *if u verifies each message m it receives with CentMail, and accepts m if and only if*

1. *$u \in R_m$*
2. *`amount/queries` $\geq t$.*

Let A be the set of compliant users that accept a message m. Then

1. *`amount`$/|A| \geq t$.*
2. *If `amount`$/|R_m| \geq t$, then each compliant $r \in R_m$ accepts m.*

Proof. If a compliant user u_i accepts m, then necessarily $u_i \in R_m$ and consequently u_i causes `queries` to be incremented. Hence, the i^{th} compliant user to verify m sees `queries` $\geq i$ (`queries` includes the current verification). Note that we do not have equality since a non-compliant user u_i could also increment `queries` (assuming $u_i \in R_m$). In particular, the final compliant user to accept m sees `queries` $\geq |A|$. By the second condition of compliance, it follows that amount$/|A| \geq t$.

Now assume `amount`$/|R_m| \geq t$ (i.e., `amount` $\geq t|R_m|$). Suppose m is verified by the (ordered) sequence of users $u_1, u_2, \ldots, u_n$, who are not necessarily compliant. The set of users who cause `queries` to be incremented (i.e., users $u_i \in R_m$) form a subsequence $u_{k_1}, u_{k_2}, \ldots, u_{k_s}$, where $s \leq |R_m|$. The user u_{k_i} sees `queries` $= i \leq s \leq |R_m|$, and consequently u_{k_i} computes

$$\texttt{amount/queries} \geq t|R_m|/|R_m| = t.$$

Hence, if u_{k_i} is compliant, u_{k_i} accepts m. Since $\{u_{k_i}\}_1^s$ contains the set of compliant users, it follows that each compliant user $r \in R_m$ accepts m. □

Property 2 guarantees recipients that a message will not be accepted by too many people; and guarantees senders that if they donate enough money, their intended recipients will accept their message.

Observe that the modified CentMail protocol does not compromise the anonymity of recipients: Since requests are authenticated, third parties—including legitimate recipients—cannot query CentMail to determine who else is an intended recipient; each user can only determine if they, themself, are an intended recipient.

3.2 Frequently Asked Questions

We raise and discuss a variety of questions and potential criticisms regarding CentMail.

1. Why should I use CentMail?

   First, CentMail promotes your favorite causes with every email you send. Second, since CentMail donations may be matched by sponsors, your financial impact is directly amplified. Third, as the system gains popularity and becomes incorporated into spam filters, your emails are less likely to end up in the junk mail folder.

2. If only a few people are using CentMail, why should I start using it? Is there a chicken-and-egg problem?

   To become an effective spam deterrent, many people need to use CentMail. However, regardless of how many other people are using CentMail, each participant gets the immediate benefit of having their causes promoted and their donations matched. In terms of spam deterrence with partial adoption, CentMail's guarantee—that the sender paid at least one cent per recipient—is diluted but still meaningful.

3. How much does it cost to use CentMail?

   Nothing. More specifically, the vast majority of users who already make charitable contributions incur no additional monetary expense. Furthermore, there is also no significant computational expense (e.g., spent CPU cycles) or significant human expense (e.g., time spent solving CAPTCHAs). The system does require users to open a CentMail account and, in some cases, to install a simple application. With support from large email providers (e.g., Yahoo!, Google and Microsoft), however, this barrier to entry can be significantly reduced.

4. Can spammers reuse stamps, sending many messages for the cost of one stamp?

   Since stamps are associated with a particular piece of content (as determined by its hash), a malicious user can only reuse a stolen stamp on identical content (see also Section 3.1). This security guarantee makes it effectively useless for third parties to steal stamps since they would have no control over the content. A related scenario is when a user attempts to reuse a single legitimately obtained stamp to validate a single message sent to thousands of people. This is in fact considered to be acceptable behavior from the perspective of CentMail, similar to the use of blind carbon copy (bcc) for emails. Recipients are informed, however, of the number of times a message has been verified, alerting them to down-weight the value of a donation appropriately.

5. Will CentMail deter web applications from sending email (e.g. Evite, Facebook, LinkedIn, etc.), and hence hamper innovation? What about other legitimate bulk mail senders?

   We suspect that any long-term solution to spam will continue relying in part on domain and content-based filtering. In particular, organizations with a history of legitimate bulk mailings that verify their identities (e.g., via DomainKeys) can be whitelisted and avoid using CentMail altogether. CentMail is tailored for instances when individuals—not organizations—send messages to recipients for the first time. This type of correspondence is particularly hard to classify as spam or ham (i.e., legitimate email) via traditional methods.

   Furthermore, CentMail would not pose any barrier to as-yet-unestablished bulk senders unless it were so ubiquitous that recipients commonly rejected unstamped messgages from non-whitelisted senders. See Questions 18 and 19 for more on the case of legitimate senders who cannot afford sufficient donations.

   When making sufficiently large donations is not an issue, the protocol accommodates bulk senders by allowing arbitrary postage amounts on a stamp. That way, when sending a single message to many recipients, the sender would attach a stamp with higher postage. Each recipient when verifying the stamp would divide `amount` by `queries` to determine the amount donated per recipient and decide if that amount is high enough to be considered legitimate. For example, if you get a dollar's postage then the first 100 recipients will see a per-recipient postage of at least one cent. The rest would, if using one cent as a threshold, reject it as spam. This distinguishes spammers from legitimate bulk mail senders if we take as a definition of a spammer someone with sufficiently low value per recipient.

6. How will CentMail work with mailing lists (listservs)?

   Mailing lists traditionally pose difficulties for economic approaches to reducing spam, as a sender's message to the mailing list address is redirected to the (potentially large) set of subscribers. While CentMail encounters some difficulty in dealing with mailing lists, a simple solution exists via whitelists. Although one could ask that either the original sender or the operator of the mailing list incur a relatively large fee to cover the number of subscribed users, we instead recommend that users whitelist relevant mailing lists to deal with this issue. Recent work on improving the management and useability of whitelists [10] has confirmed that this is an effective strategy.

7. For a message with $1 postage sent to millions of people, the first 100 CentMail users will receive it with no indication from CentMail that it's spam. Has CentMail failed these initial recipients?

   Not at all. CentMail has correctly informed those 100 recipients that they were one of at most 100 CentMail users to receive the message. We take as a definition of a legitimate message one for which the sender paid at least one cent per actual recipient. Whether there were millions of *attempted* recipients is irrelevant. And in fact the would-be spammer would not attempt to send to more than the initial 100 since it wouldn't work.

8. Does CentMail magnify the damage from email viruses?

   A virus that infects a user's computer could deplete that user's CentMail account by sending out emails on their behalf. In this case, however, three factors mitigate the potential damage: (1) An increase in the number of stamps requested by a user would alert CentMail to a potential security issue, and the user could then be alerted to a possible infection; (2) since stamp proceeds are donated, stolen stamps amount to the user donating more money through CentMail than they had intended—while still not ideal, this is perhaps a better scenario than money being lost outright; and (3) if one's computer is infected with a virus, the monetary loss of CentMail stamps—on the order of $5-$10—is likely not the primary concern given the costs associated with corrupted data and other threats associated with viruses.

9. Is Hashcash better, achieving the same goal with less user involvement?

   We're not proposing to eliminate Hashcash. Any viable solution to spam will likely draw on myriad techniques. That said, we believe CentMail is intrinsically more socially efficient in the sense that nothing (e.g., CPU cycles) is wasted.

10. Don't micropayments exert a large cognitive cost on users?

    Using CentMail does not require making a decision every time you send an email about whether or not to donate a penny. In contrast, users make upfront donations (typically $5-$10) that often amount to enough stamps to last several months or even years.

11. Will demanding too much from a sender dampen the free flow of information? Could it even be viewed as a restriction of free speech?

    By analogy, postage stamps on paper mail are generally not considered an infringement on free speech. More to the point, CentMail is coercion-free. Senders choose to attach stamps proving they made charitable donations and recipients choose to read those messages based on any criteria they like, including potentially the existence of a stamp. See also Question 18.

12. Does forwarded mail need to be restamped?

    If the message is *remailed*, then a new stamp is required since the sender's address is included in the message hash. On the other hand, if the message is *bounced* (i.e., the header—including the sender address—and message body remain unchanged), then a new stamp is not required. This latter scenario is akin to use of blind carbon copy (bcc). See also Question 4.

13. Are there privacy implications in the fact that when a recipient verifies a stamp they learn how many times the stamp was previously verified?

    It is true that CentMail is giving the recipient a clue about the number of other recipients. If the sender wishes to fully conceal that information they should not stamp the message. Indeed, much of the point of stamping a message is to prove to the recipient that the message was not sent to them indiscriminately.

14. Who will run CentMail? Will there be competition for providing it?

    In addition to implementing CentMail ourselves (`centmail.net`), we are publishing the API as an open standard (see the Appendix). We welcome other organizations to implement it and provide the service as well.

15. How can we incentivize the big players to agree to this?

    DomainKeys was adopted by Google and Yahoo! because of its promise to curb spam in the long run if it became a standard. We expect they have no less incentive to adopt CentMail.

16. Will companies pay for their employees' emails?

    Most corporate email is internal and need not be certified. However, even if employees send ten thousand emails per year, they need only one hundred dollars worth of stamps. Large companies often already donate that much per employee.

17. Is the cost of false positive spam detection high enough to get the average user to bother verifying CentMail stamps?

    This depends on how many people are stamping their mail. If stamping were universal then stamp verification would likely become the primary spam filtering attribute. At the other extreme, with very few CentMail users, there's little incentive. Somewhere in between is a threshold. Since senders have an independent reason to use CentMail (to promote their charities) it remains to be seen whether that motivates enough early adoption to cross that threshold. See [15] for a recent survey of the cost of false positives.

18. What happens to people who continue to send unstamped email?

    CentMail does not shut out users who send unstamped email. Especially at first, CentMail's spam-fighting value will be in avoiding false positives. Sending unstamped mail just means forfeiting that protection against your message being falsely labeled as spam. Only when CentMail becomes so popular that a lack of a stamp is strong evidence of spam is there pressure to stamp all mail.

19. Would CentMail be a barrier to emerging email markets in poorer countries?

    It is true that CentMail stamps are "free" only when users already intended to make charitable donations, and in fact CentMail may be prohibitively expensive for users in poorer countries. CentMail, however, does not shut out users who leave their messages unstamped (see Question 18), and we imagine CentMail to work in conjunction with myriad other spam fighting tools and techniques. In particular, CentMail eases the transition to alternative economic approaches, such as sender-posted bonds [14], which ultimately may be better suited for poorer users, but which initially lack the appropriate adoption incentives.

20. Can malicious users undermine CentMail by verifying stamps before the intended recipients do?

    In the standard CentMail protocol (described in Section 2.1) malicious users could in theory sniff a network and verify messages not intended for them, giving the impression that a legitimate user was trying to send spam. Although we believe this type of attack on the system is unlikely, Section 3.1 describes a modified protocol that closes this loophole by requiring senders to specify the intended recipients.

21. Can spammers cancel payment after procuring stamps?

    It is paramount that CentMail participants pre-pay for stamps, and do not in effect steal stamps by defaulting on their financial commitments. To a large extent, this situation is avoidable through no-refund policies and enforcing waiting periods to confirm that donations are in fact processed and debited from users' accounts.

22. Can spammers get stamps by donating to themselves?

    Users receive stamps in exchange for donations to charitable organizations *of their choice*. Here, care must be taken to guarantee that spammers are not simply funneling payments to a "charitable" organization which they ultimately control. To address this problem, we restrict donations to known reputable organizations which, for example, have been given a Charity Seal from the Better Business Bureau Wise Giving Alliance (`bbb.org/charity`) or granted non-profit tax status by the U.S. government.

23. Will legitimate users with extra stamps have incentive to sell them to spammers?

    For those who make charitable donations, CentMail stamps are effectively free up to the amount of their donations. This creates the possibility of a black market for stamps in which users resell their stamps for potentially far less than their face value. While not inconceivable, this scenario seems implausible since the seller would still be associated with any stamps he sold (i.e., the stamps are traceable to the seller, regardless of who actually sends the message).

24. Does CentMail have an adverse selection problem, in essence "over-solving" the spam problem and encouraging users to purposefully present themselves as spam bait?

    That is a more serious problem for economic approaches that pay the recipients of spam. CentMail minimizes this concern by leaving the choice of charity to the senders.

25. Is CentMail vulnerable to spammers who have zombie networks at their disposal?

    Zombie networks typically rely on being able to compromise computers undetected, to relay spam as a background process. With CentMail they would have to not only execute code on their host machines but steal users' CentMail credentials and deplete their balances. This type of attack is harder for spammers and, since CentMail generates an audit trail of stamp requests, more easily detectable.

4. THE IMPLEMENTATION

A beta implementation of CentMail is available at `CentMail.net`. In addition to an initial implementation of the CentMail API on the server side, we have developed a CentMail plug-in for Thunderbird, the popular open source email client, a Firefox plug-in for web-based email services, including Yahoo! Mail and Gmail, an Apple Mail plug-in, and Perl scripts for clients such as Pine, Mutt, and Evolution that support filtering email through arbitrary scripts.

5. CONCLUSION

Despite the aggressive development of domain-based and content-based filtering, unsolicited email continues to proliferate. Economic solutions offer the possibility of a near-complete solution to spam, yet are often stymied by the need for coordination. The would-be early adopters, that is, have little incentive to stamp messages when others are not doing the same. To overcome this so-called "flag day" problem, we develop, analyze and implement CentMail (`centmail.net`), a system through which users donate $0.01 to a charity of their choice for each email they send. The user benefits by helping a cause, promoting it to friends, and potentially attracting matching donations, often at no additional cost beyond what they planned to donate anyway. In particular, CentMail facilitates adoption by providing users with immediate, tangible benefits, while obviating the need for initial coordination. More generally, CentMail is a framework for rate-limiting many types of web-based transactions, including the posting of weblog comments and links, and account creation. While we foresee spam detection and deterrence relying on myriad tools and techniques, we believe economic schemes like CentMail will mature to play a prominent role in the fight against spam.

Acknowledgments

We thank Tom Gulik, Tom Maher, and Sergiy Matusevych for help with implementing CentMail. We also thank Mark Delaney, Miles Libbey, and Mark Seiden for providing key input and for extensive comments on this manuscript.

6. REFERENCES

[1] M. Abadi, A. Birrell, M. Burrows, F. Dabek, and T. Wobber. Bankable postage for network services. In *Proceedings of the 8th Asian Computing Science Conference*, December 2003.

[2] M. Abadi, M. Burrows, M. Manasse, and T. Wobber. Moderately hard, memory-bound functions. In *Proceedings of the 10th Annual Network and Distributed System Security Symposium*, February 2003.

[3] A. Back. Hashcash—a denial of service counter-measure (5 years on). Tech Report, 2002.

[4] M. S. Brown. *Giving USA: The Annual Report on Philanthropy for the Year 2007*. Giving USA Foundation, 2007.

[5] P. Capek, B. Leiba, and M. N. Wegman. Charity begins at ... your mail program. `http://www.research.ibm.com/spam/papers/charity-seals.pdf`, 2004.

[6] P. B. Carroll. Charity cases, July 14, 2008. The Wall Street Journal, `http://online.wsj.com/article/SB121554292423936539.html`.

[7] G. Cormack. Email Spam Filtering: A Systematic Review. *Foundations and Trends in Information Retrieval*, 1(4):335–455, 2008.

[8] C. Dwork, A. Goldberg, and M. Naor. On memory-bound functions for fighting spam. In *Proceedings of the 23rd Annual International Cryptology Conference*, August 2003.

[9] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *Proceedings of CRYPTO'92*, pages 137–147, 1993.

[10] D. Erickson, M. Casado, and N. McKeown. The Effectiveness of Whitelisting: a User-Study, 2008. Fifth Conference on Email and Anti-Spam.

[11] S. E. Fahlman. Selling interrupt rights: A way to control unwanted e-mail and telephone calls. *IBM Systems Journal*, 41(4):759–766, 2002.

[12] D. Ferris, R. Jennings, and C. Williams. The Global Economic Impact of Spam, 2005. Technical Report 409, 2005. `http://www.ferris.com`.

[13] S. Hansell. Gates backs e-mail stamp in war on spam. New York Times, `http://www.nytimes.com/2004/02/02/technology/02spam.html`, Feb. 2004.

[14] T. C. Loder, M. W. V. Alstyne, and R. Walsh. An economic response to unsolicited communication. *Advances in Economic Analysis & Policy*, 6(1), 2006.

[15] F. Research. The Cost of Spam False Positives. `http://www.ferris.com/2003/08/14/the-cost-of-spam-false-positives`, 2008.

[16] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

[17] R. Segal, J. Crawford, J. Kephart, and B. Leiba.

SpamGuru: An enterprise anti-spam filtering system, 2004. First Conference on Email and Anti-Spam.

[18] L. von Ahn, M. Blum, N. Hopper, and J. Langford. Captcha: Using hard ai problems for security. In *Advances in Cryptology—EUROCRYPT 2003*, 2003.

[19] L. von Ahn, M. Blum, and J. Langford. Telling humans and computers apart automatically: How lazy cryptographers do ai. *Communications of the ACM*, 2004.

[20] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. Captcha: Human-based character recognition via web security measures. *Science*, 321:1465, 2008.

[21] X. Wang, Y. L. Yin, and H. Yu. Finding collisions in the full sha-1. In *Advances in Cryptology—EUROCRYPT 2005*, 2005.

APPENDIX

A. SPECIFICATION OF THE CENTMAIL API

Section 2.1 gives a high-level overview of the CentMail protocol, and in particular describes the two basic operations: certification and verification. Here we specify the protocol and working implementation in more detail. CentMail is built around a web services API, in which clients make requests through an http interface and results are returned as xml documents.

We use an authentication protocol that has been adopted by several web services APIs (e.g., those by Yahoo! and Facebook), and which is both lightweight and secure. In particular, compromising the security of the system involves inverting a hash function, a problem believed to computationally intractable. Users are assigned a numerical `USERID` and a `SECRET` that is shared only with the CentMail server. The parameters of each API call are concatenated together with `SECRET`, and the hash of this string is sent to CentMail, functioning as a secure digital signature ensuring that the specified client indeed made the call. Malicious third parties could try to intercept a legitimate, signed request, and then resubmit this call to CentMail. To prevent this type of *replay attack*, a `TIMESTAMP` is sent along with each request. The client's system time need not be perfectly synced with the CentMail server, but it needs to be self-consistent. Namely, the client's `TIMESTAMP` should increase with every request; CentMail stores and checks against the last `TIMESTAMP` submitted by the user, and hence prevents the same call from being executed more than once.

Finally, for the actual message digest, we piggyback on DomainKeys which adds a digest of the message in the DKIM-Signature email header.

Certification

For stamping an outgoing message, the sender can first call `mailsig` to fetch a plain text signature to append to the document:

```
http://centmail.net/v1/mailsig?ts=TIMESTAMP&user=
USERID&sig=SIG
```

where

- `TIMESTAMP` is unix time—the number of milliseconds since January 1, 1970 (it must be guaranteed to increase for every stamp request, and must be within 10 minutes of CentMail's time);
- `SIG` is the SHA-1 hash of `TIMESTAMP` + `USERID` + `SECRET`.

CentMail returns a preconstructed signature along with the components of the signature:

```
<centmail>
  <status>OK</status>
  <user>USERID</user>
  <org>ORGID</org>
  <nonce>NONCE</nonce>
  <sig>
    <line>NICKNAME certified this email by donating $0.01
          to ORG</line>
    <line>Powered by CentMail.net -- Do good. Fight spam.
          </line>
    <line>http://centmail.net/stamp/NONCE</line>
  </sig>
</centmail>
```

`NICKNAME` and `ORG` are associated with `USERID` and `ORGID` respectively. Upon sending the message, the sender makes another call to actually certify the message.

```
http://centmail.net/v1/certify?user=USERID&org=
ORGID&digest=DIGEST&amt=AMOUNT&ts=TIMESTAMP&sig=
SIG
```

where

- `AMOUNT` is the amount of money to be debited from the certifier's account, typically USD 0.01. For simplicity, we currently denote all monetary transactions in US dollars;
- `DIGEST` is the SHA-1 hash of the contents of the message to be certified;
- `TIMESTAMP` is unix time in milliseconds;
- `SIG` is the SHA-1 hash of `AMOUNT` + `DIGEST` + `TIMESTAMP` + `USERID` + `SECRET`. The '+' operator is concatenation.

Verification

Stamps can either be verified anonymously or verifiers can authenticate with their own CentMail user ID. Only in the latter case will the number of queries for the stamp be incremented. In other words, only authenticated queries count. In this way, ISPs or other intermediaries can check stamps without interfering with end users' verification of them.

The API calls to verify a stamp anonymously and as an authenticated user (with a CentMail `USERID` and `SECRET`) are, respectively,

```
http://centmail.net/v1/verify?digest=DIGEST
```

and

```
http://centmail.net/v1/verify?digest=DIGEST&user=
USERID&sig=SIG
```

where

- `DIGEST` is the SHA-1 hash of the message canonicalized per DomainKeys;
- `SIG` is the SHA-1 hash of `DIGEST` + `USERID` + `SECRET`.

Both versions of the `verify` call return the following:

```
<centmail>
  <valid>VALID</valid>
  <amount>AMOUNT</amount>
  <queries>QUERIES</queries>
</centmail>
```

where

- `VALID` is 1 if the stamp was purchased for the specified message and 0 otherwise;
- `AMOUNT` is a number like 0.01 denoting the amount of money in US dollars donated when the stamp was created;
- `QUERIES` is the number of authenticated queries that have been made to verify this stamp, a non-negative integer.

Request

In the case of requesting donations (e.g., during account creation), the API call is:

```
http://centmail.net/v1/request?amt=AMOUNT&rid=REQUESTERID&tid=TRANSACTIONID&url=RETURNURL&sig=SIG
```

where

- `AMOUNT` is the amount the requester is asking the user to donate;
- `REQUESTERID` is the `USERID` of the requester;
- `TRANSACTIONID` is an id generated by the requester to uniquely identify this transaction;
- `RETURNURL` is the location to which a response will be posted;
- `SIG` is the SHA-1 hash of `AMOUNT` + `REQUESTERID` + `TRANSACTIONID` + `RETURNURL` + `SECRET` where `SECRET` is the requester's secret key.

After the end user agrees or declines to make a donation, the CentMail server posts a response to `RETURNURL` with the following parameters set:

- `TRANSACTIONID` is the id generated by the requester to uniquely identify this transaction;
- `VALID` is 1 if the user made the requested donation, and 0 otherwise;
- `SIG` is the SHA-1 hash of `TRANSACTIONID` + `VALID` + `SECRET` where `SECRET` is the requester's secret key.