Controlling Spam through Lightweight Currency *

David A. Turner and Daniel M. Havey
Department of Computer Science
California State University San Bernadino
San Bernardino, CA 92407
(dturner@csusb.edu, dhavey@yahoo.com)

November 4, 2003

Abstract

Spam is an ongoing problem on the Internet today, and an increasing body of literature from research papers to the popular press addresses the problem. The solutions generally fall into the categories of payment-based, legislative, and filter-based. In this paper, we present a payment-based solution to the spam problem that can reduce the level of unsolicited emails to reasonable levels without imposing legal restrictions on free speech, and without the use of email filters. Specifically, we propose that mail transfer agents (MTAs) make payments with the Lightweight Currency Protocol (LCP) when sending email into other mail domains. Mail service providers specify payment requirements that discourage spam, but encourage the free flow of other email. Thus, our proposal makes it possible to control spam without end user involvement or modification to user agents. The advantages of using the LCP for mail delivery include its conceptual simplicity, security, scalability, flexibility and low cost. With LCP, mail domains have the ability to issue their own currency or use currency issued by either other mail domains or by other LCP-based service providers outside of the email system. The ability for domains to issue their own currency ensures the system will scale to meet demand, and enables domains to operate spam-reduced mail services without paying an outside authority for needed currency.

Keywords: spam, micro-payments

1 Introduction

For the purpose of this paper, we make a distinction between the terms *spam* and *junk mail*. By spam, we mean email messages that possess one or more of the following characteristics:

1. indiscriminately copied to millions of inboxes, rather than targeted to users that have "opted in" in some way
2. contains false return addresses, or other false envelop data
3. contains material that is widely held to be objectionable or embarrassing that is sent without consent
4. was generated by a computer virus
5. has been specifically formatted to pass through email filters

By junk mail, we mean unsolicited email that doesn't fit into the definition of spam, but contains advertising targeted to the recipient. In other words, junk email should resemble junk mail that people receive non-electronically by post. Obviously,

*The support of the Associated Students Incorporated at CSUSB and the National Science Foundation under award 9810708 are gratefully acknowledged.

these definitions lack absolute precision. Nevertheless, they work reasonably well when describing how our payment-based system works in relation to other methods.

Approaches to solving the spam problem generally fall into three categories: litigation, filtering and payments. Under the litigation approach, governments pass laws against sending spam, and enforcement of these laws reduces spam after spammers begin to fear punishment. This approach is naively supported by a large segment of the population, because people are unaware of both the social costs of increasing government control of speech, and the economic costs of enforcement. Payment-based systems are superior to litigation approaches, because they reduce spam without adding legal restrictions to speech in the form of email, and avoid increasing government taxation.

The argument presented so far against a litigation-based approach assumes that government expenditures will be sufficient to solve the problem; however, this is not likely to be the case. Another consideration is that email can originate from any country, and so a litigation-based approach to spam would require international agreements, and countries would need to provide relatively equal levels of enforcement. Clearly, litigation is an overly complex and expensive solution, and ought to be reconsidered by policy makers in the government.

Another general approach to spam is the use of email filters, which automatically block spam from entering user inboxes. The main problem with the filter-based approach is that it is not 100% accurate in detecting spam; both false positives and false negatives occur. A false positive means the filter has mistakenly marked a legitimate email as spam, and thus does not get delivered to the user (or gets delivered to a bulk mail folder). A false negative means that an email that is spam is marked as not being spam. While this may not result in too much of an inconvenience to the user that receives an occasional spam, it encourages spammers to send more emails in an attempt to bypass the filter. Thus, filters have the negative effect of exacerbating the problem of resource consumption originating from spam. Clearly, filters are a temporary solution to the problem at the current time, and do not represent a long-term, cost-effective solution as do payment-based systems.

Payment-based approaches rely on cooperating email systems to create economic disincentives to spam. To accomplish this, these mail servers require a small payment in exchange for delivering an email to the recipient's inbox. The payment is kept small enough to allow legitimate email to pass into user inboxes, but large enough to make sending large numbers of junk emails unprofitable. While a payment requirement will deter spam, it will most likely not deter junk email. Advertisers still deliver junk mail by post even though the cost is roughly 20 cents per letter. Although it is possible to set email delivery payments at this level, a payment on the order of one cent per email is more likely. Thus, it is also likely that advertisers will simply accept this cost and continue to deliver junk email. It is also likely that advertisers will expend extra effort to narrow their mailing lists to avoid sending unnecessary emails in order to reduce costs. In this paper we propose a solution to the problem using Lightweight Currency Protocol LCP. The idea is that each SMTP mail server will use LCP pseudo-currency to make a payment every time that they send email and will receive a payment when they receive a piece of email.

Several other researchers have proposed payment-based schemes. One idea is to require a *proof of work* (POW) payment [6, 8]. These payments are one-time payments that the sending domain sends to the receiving domain; they are not reusable or transferable. After the sending domain connects, the receiving domain presents a mathematical puzzle to the sender that requires a significant amount of time for the sender to solve. After the sender computes the solution to the puzzle, it sends the solution to the receiving domain. While computing the solution to the puzzle consumes significant resources, verifying that it is correct can be done relatively easily. After the receiving domain verifies the response is correct, it accepts delivery of the email. The idea is that spammers will not have the resources to compute the solution to the millions of puzzles that will be presented to them. There are two essential problems with POW schemes when compared to LCP: they waste the resources of senders by requiring a meaningless computation to be performed, and compute

time requirements vary too much across the range of CPU speeds.

Fahlman and Wegman of IBM have proposed the use of charity stamps for controlling spam [10]. In this system, senders of email purchase stamps that are required in order for delivery of email, and the proceeds are given to charity. One problem with this system is the requirement of a central authority to decide on the price of stamps, and on which charities earn the proceeds. The LCP-based system, in contrast, operates in a self-regulated, fully open market with multiple currencies. Power is distributed among participants, rather than concentrated in a central authority. Additionally, LCP currencies are not restricted to email services, but are redeemable for services outside of email.

In section two of this paper, we present an overview of the LCP, and highlight its features as they pertain to our solution. In section three, we present our solution to the problem of spam, and define the LCP-based email architecture. In section four, we investigate the cost of controlling spam, and detail the economic ramifications of the use of Lightweight Currency payments. In section five, we illustrate the relationship of our spam solution to the larger peer-to-peer resource market, and describe how we fit into the big picture. Section six is devoted to the investigation of security issues, and in section seven, we describe deployment strategies. We conclude in section eight.

2 Overview of LCP

The lightweight currency protocol is a relatively simple mechanism by which an organization can issue a generic currency that can be used as a medium of exchange independent of any particular application. The term lightweight reflects the simplicity by which an organization can issue such currencies, and the ease by which implementers can integrate lightweight currency payment mechanisms in their applications. Lightweight currency is an effective alternative to real-world currency for micro-payment schemes, because it is easy to integrate into applications, and it is not directly tied to real-world currencies.

Under the lightweight currency paradigm, an organization issues a currency by generating a public/private key pair, and distributes the public key as its identifier. Alternatively, an issuer can publish a certificate that binds a domain name to its public key, and then be identifiable through the domain name.

Currency holders also generate a public/private key pair, and use the public key as an identifier. A currency holder holds a particular currency when the issuer of that currency has a record for the number of units of currency owned by the holder of the private key.

The use of public keys as identifiers has two benefits: it provides a means for entities to generate globally unique identifiers without the need of a central naming authority to control naming collision, and it allows entities to authenticate and establish secure communication channels.

The LCP is a request/response protocol in which the issuer plays the role of server and the holder (of currency) plays the role of client. There are two request messages and two corresponding response messages. An entity spends a particular currency he/she holds by sending a transfer-funds message to the issuer of the currency that identifies the recipient's public key identifier, the amount to be transferred, and a transaction-id. If the sender of funds has a sufficient balance of funds, the issuer will debit the senders account by the amount requested (and optionally a small transaction charge), and will credit the account associated with the recipient. The recipient verifies that payment was received by connecting to the issuer and sending a get-activity request message. The issuer responds with an account-activity-statement that lists the deposits made to his/her account since he last inquired. See [1] for an introduction to the LCP, and visit lightweightcurrency.org for a complete definition of the protocol.

3 LCP-Based Email Architecture

We propose a payment mechanism whereby email servers require a payment to be made in a lightweight currency in order to accept an incoming email. The process is transparent to the sender: the mail forwarding agent of the sender's domain is responsible for making this payment, and no changes need to be made to user agent software. Thus, responsibility is placed on the email service providers to ensure that spam is not passing through their systems. Methods by which service providers monitor and control their user accounts are not covered in this paper, but it is an important component of an overall solution based on the lightweight currency model.

Because lightweight currency can be issued by any organization, including the sending and receiving domains, there is a lot of flexibility during payment negotiation. In the remainder of this section, we break down payment negation into the important cases, and explain how both sides might come to an agreement. For each of these cases, we assume that a user Alice is sending an email to user Bob. Alices email address is alice@aaa.com, and Bobs email address is bob@bbb.com. Thus, for this email to be delivered, payment is made by domain aaa.com to domain bbb.com. We refer to these domains as simply A and B, respectively.

Consider the case that A and B have a history of mail exchange, and that based on this history, both sides have agreed to accept payment in the others currency as long as the accepting end does not hold what it considers to be an excessive amount of the others currency. If A holds B dollars, then the payment is clear: A pays B one B dollar for the delivery of Alices email.

If B holds A dollars, then A will propose the payment of another one of its dollars for the delivery of the email. If B does not hold (what it considers to be) too many A dollars, then it accepts this payment. On the other hand, if B holds too many A dollars, it will reject A's offer, and ask for payment in another currency. In this case, B may send a list of currencies that it is seeking. This list will likely contain both powerful currencies with wide acceptance in the larger resource market, and the currencies of domains that hold large amounts of B's currency. The reason B would accept a currency with wide acceptance include the ability of B to redeem this currency for resources that support other applications of interest to its stakeholders, or to redeem this currency for real-world dollars. The reason B would accept a currency from a domain that holds large amounts of Bs currency is that B wants to avoid getting into the situation that A is in. To understand this, suppose that domain C holds a large number of B dollars, and that B wants to deliver email to C. If B gets a C dollar from A, then B is assured of having the email delivered to C by paying with a C dollar. On the other hand, if B does not have a C dollar, then C may reject B's offer to pay another B dollar, forcing B to spend a dollar of a widely accepted currency, which B may need to purchase with real-world dollars.

Now we consider the case that A is sending email to B for the first time. It is possible, although unlikely, that A holds currency issued by B. If it does, it pays in B dollars. A may try to pay with A dollars. If A is using a certificate issued by a certificate authority that B trusts, then B may accept a limited number of A dollars. In this manner, B lets new email into its user accounts, but only up to a limit. If B's users respond to email coming from A's domain, then A dollars will be spent by B to send mail back to A. However, there is the possibility that spammers could abuse the system if they are able to obtain new certificates at a cost less than the profit their spam may generate. Thus, the recommended practice is for email systems to require payment from unknown domains in a widely accepted currency that is redeemable for valuable resources or real-world dollars, or to accept currency issued by domains for which the recipient domain needs funds, as described earlier.

Spammers send out millions of emails from their domains without receiving a commensurate level of responses. Thus, a spammer can not acquire the lightweight currency needed to make so many deliveries. The spammer is thus forced to earn lightweight currency by selling useful resources in the resource market, or to purchase widely-accepted currency using real-world dollars. While this may not eliminate

all email that is judged to be spam, it will greatly reduce the large numbers of unsolicited emails that are currently flooding user inboxes. Mail service providers have the flexibility of adjusting their pricing policies to reduce spam to acceptable levels while allowing welcome email to enter into user inboxes with little or no cost.

4 The Cost of Controlling Spam

Currently, bulk email providers charge roughly $100 per million pieces of email, which is 0.01 cents per email. If these bulk email providers were required to purchase lightweight currency that results in a cost of one cent per email, the cost for sending one million emails becomes $10,000. Thus, it is reasonable to expect that a large percentage of today's spam would be eliminated with a pricing scheme resulting in a charge of one cent per email.

On the other hand, companies now distribute advertising by postal mail at a rate of roughly 20 cents per piece. At a cost of one cent per email, advertisers may decide to spend one cent per email to deliver their message. We expect the result to be an improvement in the quality of unsolicited email. That is, advertisers will only send email that is expected to generate a sufficiently positive response. So, a lightweight currency based solution to email would reduce unwanted emails, but would not eliminate them altogether. This is necessary if email is to remain an open communication channel in which one user can send an unsolicited (but welcomed) email to a new user.

With a lightweight currency approach, individual email service providers have the flexibility of tuning their payment acceptance policies to reduce the frequency of unwanted email to acceptable levels. As spamming behaviors change over time, the systems flexibility allows for system operators to continually respond to these changes by adjusting their pricing policies.

In some cases, senders of email that are welcomed by recipients naturally have a higher rate of outgoing email compared to their incoming mail. In this case, the sender will not be able to collect revenues from incoming emails. As an example, this may be the case with a mailing list operator, or with an e-commerce site that sends purchase confirmations.

A legitimate mailing list operator (an operator sending email that is welcomed by recipients) does not operate at the same levels as indiscriminate spammers. Perhaps a mailing list operator would send out one thousand emails per day as opposed to a spammer who might generate 10 million emails per day. At one cent per email, this mailing list operator pays $10 per day compared to the spammer who pays $100,000 per day. Thus, an appropriately chosen pricing scheme imposes sustainable costs to the mailing list operator, but not to the spammer.

Still, any amount of real-world dollars may be unacceptable to mailing list operators. In this case, these operators can earn the needed lightweight currencies through several alternative approaches. One solution is for the mailing list operator to request users to respond to emails it sends as an indication that the user desires to continue receiving emails from the list operator. Each time a user sends an email of support, the users domain must transfer funds to the mailing list operators domain. Users that truly value the list operators efforts will take the two or three seconds required to send such a reply. This approach has the advantage of discouraging overzealous list operators from sending unwanted emails.

A second solution is for the list operator to collect needed currencies by selling unused bandwidth, storage and compute power in the raw resource market. Similar to the example presented in the previous section, the list operator can request payment in various currencies that are acceptable to the domains to which the list operator wishes to send email.

A third solution is for the list operator to require lightweight currency payments for the services that it provides to recipients. In this case, the recipients will need to purchase, or earn lightweight currency to make these payments. Alternatively, the list operator can request lightweight currency funds from some of its users as a show of support, and thus to enable it to deliver emails to the larger community of recipients.

Another way to manage an imbalance of outgoing

to incoming emails is to charge more for incoming emails and pay less for outgoing emails. For instance if a domain sends 2 emails for every one email received then this domain could try to charge 2 times as much for receiving email. Relatively small differences in prices may be acceptable to email systems, because it will correct for any natural imbalances in flows between domains, but still not enable spammers to attain profitability.

Under the lightweight currency based payment system, E-commerce sites that send purchase confirmations, but do not receive the same level of incoming emails from customers, would also need to earn or purchase the lightweight currency needed to make email deliveries. However, unlike the mailing list operators, these confirmations correspond to revenues that renders the cost of email insignificant. For example, if the average sale at an e-commerce site is $20, the one cent cost they need to expend to deliver a confirmation by email is negligible.

5 Relation to the Larger P2P Resource Market

Rather than define a lightweight, micro-payment currency for the restricted use of making email delivery payments, we propose that email delivery payments be made with the Lightweight Currency Protocol (LCP) as defined in [1], because it is fully transferable within the context of other applications. Because LCP is SOAP-based, it is relatively easy for implementers to incorporate into applications. The benefit to using currencies that are fully transferable into other contexts is that these currencies will more easily acquire value. Additionally, email senders that send more email than they receive have access to additional methods of acquiring necessary currency. It also provides a way for email systems that receive more currency than they consume to redeem it for other desirable resources, or trade it more easily for real-world dollars.

We illustrate with an example how the email system would interact with the broader resource market. Suppose that user Alice maintains an email account with yahoo.com, and she is the recipient of a newsletter. In the raw resource market, autonomous agents sell surplus CPU cycles, storage and bandwidth for their users. These agents provide a means for users to specify currency targets. This is useful for the user who needs to acquire specific currencies for the purchase of certain services, as is the case with Alice. So, Alice could input into her interface that she needs one hundred units of Yahoo currency sent to domain operating the newsletter she desires to receive. The agent would then seek to sell unused resources for Yahoo dollars. The resources do not need to be sold directly to Yahoo, but can be sold to anyone who holds Yahoo currency. Perhaps an organization has holds some Yahoo currency with which it wishes to purchase CPU cycles, or perhaps a provider of streaming video has some Yahoo currency with which it wishes to purchase bandwidth for the distribution of its content. In either of these cases, Alice's resource selling agent may earn the desired yahoo currency by providing the sought after resources. After the yahoo dollars are earned, Alice's agent transfers the required amount to the domain of the mailing list operator. Now, the mailing list operator can delivery emails to Alice without acquiring funds through any other activity other than delivering its content. It should be kept in mind that this is a futuristic scenario that does not necessarily need to occur for the LCP-based payment system to succeed, but it is feasible, and illustrates the flexibility that results from the use of LCP.

6 Security Issues

The lightweight currency protocol was designed as a fully secure method of micro-payment transactions. LCP messages are transported over SSL using client and server authentication. However, possible security vulnerabilities emerge because of the ability for spammers and other criminals to generate any number of public key identities at will. We call this the throwaway identity problem. We will look at the throwaway identity attack as well as a man in the middle attack in this section.

A naive policy that accepts currency from anyone

up to a certain limit is susceptible to the throwaway identity attack. In this attack, a spammer generates a public key identity, issues a currency through it, and uses this currency to send emails until the recipient domains currency limit is reached. The spammer generates as many public keys as necessary to deliver the desired number of emails to the recipient system. A man-in-the-middle attack is possible if a recipient system accepts public key identities that are not bound by a certificate authority to the domain name of the sender. In this attack, a man in the middle impersonates the parties on each end of the communication, issues currency that is worthless, and obtains currency issued by the sender. For example, suppose that A is sending email to B, and B has decided to accept A dollars as payment for delivery of the email. The man in the middle delivers email to B for A, but in the process of doing so, it makes B think that its public key is from A, so that B accepts worthless currency from the attacker. While this is happening, A transfers to the attacker a worthwhile currency. Although this kind of attack can not result in the attacker obtaining very much currency from A, it can discredit A to B, and result in a disruption of normal email service.

There are two solutions to the throw away identity attack, depending on whether the recipient system has a small or large number of inboxes. If the recipient system has a small number of inboxes, then it is possible for a spammer to purchase a certificate, and use this to deliver a small number of emails to all small-scale systems. In this way, the spammer distributes the expense of the certificate across a large number of small systems. Thus, if a small system does not send email into the domain of the originating email, it should not accept that domains currency, even if that domain presents a valid certificate. If on the other hand, it does send email into the domain, it may accept the senders currency. However, it should only accept its currency if presented with a valid certificate, in order to avoid exposure to the man-in-the-middle attack.

The second solution for the throw away identity attack is appropriate for domains with large numbers of inboxes. In this case, a spammer must purchase too many certificates in order to reach a large number of inboxes, and so there is a built-in economic disincentive by virtue of the domains size. In this case, the recipient domain can simply accept currency from an issuer who presents a valid certificate.

Security is a matter of policy. A nave security policy while easy to implement and use is also easy to cheat. A strong policy requires more thought to design and implement but can be just as easy to use while still providing powerful protection for its users.

7 Deployment

There are two avenues through which deployment can take place. In the first approach, system administrators install email server upgrades that incorporate the LCP based payment mechanism, and then configure and turn on the new functionality so that the system accepts both payment based mail delivery and ordinary delivery. We call this the partial deployment strategy. In the second approach, new spam-reduced email services are introduced into the market, and users maintain at least two email identities: one for their ordinary email accounts, and one for the spam-reduced account. We call this the full deployment strategy. Both approaches are promoted by the development of free open source libraries that provide the new functionality, and by incorporating the new functionality in existing open source code, such as sendmail, postfix, etc.

Under the partial deployment strategy, mail service providers are encouraged to migrate to the new system incrementally. When LCP functionality is turned on in partial deployment mode in a mail domain, then its incoming mail server announces its support during the EHLO greeting when a mail sender connects. If a connecting client does not support LCP, then the incoming mail server simply accepts mail from the sender in the ordinary manner without payment. If the connecting client does support LCP, then both sides transition into the new paradigm. However, for this transition to be secure, both ends should authenticate to each other by presenting certificates issued by a trusted authority. If this precaution is not taken, then a man-in-the-middle attack is possible.

The connecting mail domains can immediately start using each others currency with some reasonable upper limit on the number of dollars one end will hold of the other ends currency. This limit may be a function of the number of inboxes the system supports. However, this approach exposes small-scale mail systems to the throw-away identity attack described in the section on security. Thus, the upper limit of currency held by one end needs to be a fraction of the number of inboxes. After mail flows in both directions between the two domains, then the upper limit on the number of dollars one end holds can be increased, because outgoing responses are demonstration that the incoming mail from the other system is welcome. Of course, users should not respond to spam email for this to work.

The partial deployment strategy is made more feasible if the currency being transacted between domains is used in other application domains, such as peer-to-peer content distribution, distributed backup, grid computing, content sales, etc. In this way, mail service operators will become cognizant of the larger LCP resource market, and thus be more apt to turn on LCP payment functionality in their systems. The benefit for partial deployment is that the delivery of spam can be detected and eliminated from a growing list of cooperating domains.

We also believe that an educational campaign about the proper use of the system and the benefits that can be achieved from operating in a spam reduced LCP environment should be launched targeting system administrators and corporate information officers.

While the benefits of partial deployment are perhaps not so clear or convincing, the benefits of full deployment are more obviously significant. Under full deployment, users are expected to establish new email identities, which they maintain along with old email identities. Managing multiple email accounts is a behavior that is commonly observed, and so the extra burden on users that this presents should not present an insurmountable barrier.

Users can use their LCP accounts to send and receive email to and from users that also maintain LCP accounts. The approach will work well for small groups of users the exchange email frequently. One example would be a group of coworkers who frequently exchange messages. In this case, each member of the group agrees to use an LCP account for communication with other group members. Group members will be able to check their inboxes without needing to weed out spam. Another good example is a family, in which family members agree to use an LCP account for correspondence with each other. Still another example is a company that decides to set up an LCP system for it employees to use. A company-based system can be used for employees to communicate with each other, but also for emails sent to and received from other companies that do the same.

For the full deployment strategy to succeed, LCP-enabled mail domains should have a mechanism that allows participating domains to discover each other without human intervention. Such a mechanism will support the automatic aggregation of LCP domains into an infrastructure capable of replacing SMTP-based email. Thus, some combination of partial and full deployment strategies is called for. This hybrid strategy would work as follows. Suppose company A establishes an LCP mail domain in addition to its ordinary mail domain. When mail arrives from company B, company As mail server announces support for LCP payments. If B is not an LCP domain, then A accepts the incoming email and delivers it to the users non-LCP inbox. On the other hand, if B is an LCP domain, then A and B initialize a payment-based relationship, and mail is now delivered to the users LCP inbox. Over time, as more users adopt LCP accounts, increasing amounts of quality mail are delivered to LCP inboxes, while spam is quarantined to inboxes that are shrinking in importance.

References

[1] D. Turner and K. Ross. A Lightweight Currency Paradigm for the P2P Resource Market. Submitted, 2003.

[2] D. Turner and D. Havey and J. Ewart. Alocating Resources in Storage Cooperatives with Pseudo Currencies. International Conference on Computer Science and its Applications, San Diego, CA, Jul 2003.

[3] I. Androutsopoulos, J. Koutsias, K. Chandrinos, G. Paliouras, C. Spyropoulos. An Evaluation of Naive Bayesian Anti-Spam Filtering. In Proceedings of the workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning, 2000.

[4] X. Carreras and L. Mrquez. Boosting trees for anti-spam email filtering. In Proceedings of RANLP-01, Jth International Conference on Recent Advances in Natural Language Processing, Tzigov Chark, BG, 2001.

[5] D. Madigan. Statistics and the War on Spam. Statistics, A Guide to the Unknown, 2003.

[6] M. Jakobsson and A. Juels. Proofs of work and bread pudding protocols. In Proceedings of the IFIP TC6 and TC11 Joint Working Conference on Communications and Multimedia Security (CMS 1999). Kluwer, 1999.

[7] D. Bleichenbacher, E. Gabber, M. Jakobsson, Y. Matias, and A. Mayer. Curbing Junk E-Mail via Secure Classification. Financial Cryptography, 1998.

[8] A. Back. Hashcash: A Denial of Service Counter-Measure. Tech Report, Aug 2002. http://www.cypherspace.org/hashcash/hashcash.pdf

[9] M. Franklin and D. Malkhi. Auditable Metering with Lightweight Security. Proceedings of Financial Cryptography 1997, Springer-Verlag, LNCS 1318.

[10] A. Robbins. You Spam, You Pay. PC Magazine, April 30, 2003. http://www.pcmag.com/article2/0,4149,1040763,00.asp

[11] R. Rivest, A. Shamir. PayWord and MicroMint: Two Simple Micropayment Schemes. In Proceedings of the Fourth Cambridge Security Protocols Workshop, Springer LNCS v 1189.

[12] C. Dwork and M. Naor. Pricing via Processing or Combatting Junk Mail. Proceedings, Lecture Notes in Computer Science, Vol. 740, SpringerVerlag, 1992.

[13] Brad Templeton. E-Stamps. http://www.templetons.com/brad/spam/estamps.html