Controlling Spam by Secure Internet Content Selection

Amir Herzberg, herzbea@cs.biu.ac.il, http://amirherzberg.com

Computer Science Dept., Bar-Ilan University, Israel

Abstract

Unsolicited and undesirable e-mail (spam) is a growing problem for Internet users and service providers. We present the **Secure Internet Content Selection (SICS)** *protocol, an efficient cryptographic mechanism for spam-control, based on allocation of responsibility (liability). With SICS, e-mail is sent with a* **content label**, *and a cryptographic protocol ensures labels are authentic and penalizes falsely labeled e-mail (spam). The protocol supports trusted senders (penalized by loss of trust) and unknown senders (penalized financially). The recipient can determine the compensation amount for falsely labeled e-mail (spam)). SICS is practical, with negligible overhead, gradual adoption path, and use of existing relationships; it is also flexible and appropriate for most scenarios, including deployment by end users and/or ISPs and support for privacy and legitimate, properly labeled commercial e-mail. SICS improves on other crypto-based proposals for spam controls, and complements non-cryptographic spam controls.*

1 Introduction

E-mail main (and initial) use is professional and personal communication. However, e-mail is very efficient and low-cost; therefore, a growing fraction of e-mail messages contains other types of content, mostly advertisements. Many users, and providers, find themselves wasting substantial resources dealing with such messages, which are often undesired. There are few conventions for identifying advertising or other potentially undesired messages, e.g. prepending the string `ADV` to the subject line, allowing mail servers and user agents to quickly discard them. Unfortunately, most messages containing potentially-undesired content do *not* contain appropriate label for the type of content; indeed, the authors often use different evasive techniques to make it hard to distinguish between the messages and desirable professional/personal communication. We use the term *spam* for messages containing potentially undesirable content, without proper identification. Spam, and our solution (SICS), apply also to other forms of `push` content such as pop-up web pages and instant messaging (where spam is sometimes called *spim*), although we mention mostly e-mail.

Spamming wastes considerable machine and human resources – most notably, the recipient's time. Indeed, spamming is reducing the usefulness of e-mail as a communication mechanism these days. Many users reduce or avoid using e-mail, most limit the distribution of their e-mail address, and many desirable messages are lost by aggressive (human and automated) filtering. As a result, there are many proposals and mechanisms trying to control and prevent spam; we categorize them in Table 1 (and briefly review them in the appendix).

Unlike most existing and proposed mechanisms, the **Secure Internet Content Selection (SICS)** protocol focuses on ensuring that messages contain truthful labels. Namely, SICS allows messages containing advertising or otherwise potentially offensive or potentially undesirable content, as long as they are properly labeled, allowing trivial, efficient filtering by mail servers and recipients. The goal of SICS is to allow practical, efficient, flexible and secure usage of *content labels* identifying the type of content in messages (web pages, files, or other objects). In particular, SICS allows identification of (known) senders or ISPs that provide or endorse falsely labeled messages (i.e. spam). SICS also allows us to accept e-mail from unknown, new senders, by ensuring predefined financial compensation (via some trusted intermediary) if the e-mail contains false label (i.e. is spam).

	Non-Cryptographic Spam Controls	Cryptographic Spam Controls
Trust-based Spam Controls	Unlisted recipient e-mail address Unique unlisted address per trusted sender Senders whitelist Senders / domains blacklist Identification of source domain (`caller-ID`)	Source (sender/server) authentication Pay to Send Message signing with E-mail Policy Indication **Secure Internet Content Selection (SICS)**
Spam Controls without any Trusted Entity	Pattern, Heuristic or Human content filtering Return address validation Proof of human work	Proof of computational work (computational puzzles)

Table 1: Categorization of Spam Controls by use of cryptography and of trusted entity

While we focus on e-mail, content labeling is relevant to many other Internet applications such as the Web. Indeed, SICS extends the Platform for Internet Content Selection (PICS) W3C recommendations [MRS96, MK*96], and could be used to provide improved security for PICS-compliant applications (e.g. browsers limited to `non-offensive` content, used mostly for children). Currently, PICS appears to have moderate adoption, mainly for different forms of potentially offensive content (e.g. nudity); we argue that secure labeling could be much more useful, especially for filtering e-mail and other `push` content such as pop-up windows and scripts in web pages, instant-messaging, broadcast and other applications. In this paper, we do not discuss specific format for labels and signatures; implementations could use the PICS formats for labels as described in [MK*96] and DSIG signatures as in [DC*98], or other formats, e.g. as suggested in [CSRI04, sections 8-9]. We tried, as much as possible, to reuse and be consistent with the PICS terminology (content label, rating service, etc.).

1.1 E-Mail Operation, Security and Threats

E-mail originates at a *source mail client (user agent)*, which transfers it to a *source mail server* (also referred to as MTA, Mail Transfer Agent). The source mail server sends the e-mail message to the destination's mailbox server, directly or indirectly (via additional mail servers, acting as e-mail relays); the communication between mail servers (and between some source mail clients and their mail servers) is using the Simple Mail Transfer Protocol (SMTP). Throughout this process, the e-mail is sent in a printable, easy to read and process format, and usually without any encryption or any other cryptographic protection.

Therefore, e-mail may be eavesdropped and modified while it passes the source network, the target network, and the intermediate links, SMTP server gateways and routers. Furthermore, some SMTP servers are willing to receive e-mail messages from any SMTP server or client, often without authenticating them[1]; as a

[1] A growing fraction of e-mail servers are authenticating incoming e-mail, e.g. by using SMTP over TLS, often with opportunistic authentication (i.e. not insisting on trusted CA) for interoperability; this is good. On the other hand, many mail servers, e.g. in Internet cafés and hotels, accept e-mail from arbitrary senders (and cannot authenticate source address).

result, it is easy to forge the sender (source) address in e-mail messages. Indeed, spammers usually use false sender address – which could be an invented address, an address picked from public sources (e.g. web pages, forums), an address from an address book of a penetrated computer (a common technique for viruses), or an address of a specific victim (to `frame` the victim, who will be blamed for issuing the spam e-mail).

In order to allow secure use of e-mail, and deal with such threats, several standards and systems offer encryption and source authentication mechanisms for e-mail messages, including S/MIME, PGP and PEM. The S/MIME standard is implemented in many e-mail products, and PGP has enjoyed wide recognition and substantial number of installed clients. However, only relatively few e-mail messages are protected.

There are different explanations and reasons for the limited use of encryption and authentication of e-mail. One reason may be that the widely implemented e-mail security solutions, in particular S/MIME, are based on the use of identity public key certificates. In particular, source authentication is done by sending the e-mail signed, together with an identity public key certificate; the receiver validates the certificate and retrieves the sender's public key, name and e-mail address from it, and then validates the signature of the sender over the e-mail. However, most users do not obtain an identity certificate for e-mail. The main reason is probably the complexity, effort and cost involved in obtaining such identity certificate. In many countries, there is also a substantial liability risk to owners of identity certificates, due to digital signature laws that make them responsible for any signed message; considering the insecurity of most personal computers, such liability is unacceptable. Finally, e-mail security requires adoption by both sender and recipient, reducing motivation for early adopters. For these and other reasons, almost the entire e-mail traffic is neither encrypted nor authenticated.

In this work, we are especially concerned with the lack of source authentication, which means that spoofing of the identity of the e-mail source (the sender) is trivial. We observe that receivers do not always need to authenticate the *identity* of the source of e-mail. More frequently, receivers care mostly about the *properties* of the sender or of a particular message. We focus on identifying and filtering *spam* – unsolicited and undesirable e-mail, including offensive content, malicious content (e.g. virus) and/or advertising content, when not clearly labeled. We do *not* consider e-mail messages that contain properly labeled advertising content as spam.

We notice that spam is closely related to other threats to computer security, and in particular to different forms of malicious software (`malware`) such as viruses and Trojan horses, and also to spoofed web sites and other online scams. As shown in Figure 1, there is a `vicious cycle` in which spam plays a central role. The cycle begin when a spammer buys a domain and begins to send spam (left side). The critical point in the process is the response of the user (or his agent) to the incoming spam. When a user reads a spam message, there are four main possibilities:

1. The user may simply discard the spam; here, the damage is mainly the waste of time by the user, as well as the bandwidth, storage and processing costs.

2. When spam contains advertising content, the user may actually respond favorably to the advertisement, e.g. by buying the advertised product or service.

3. Often, spam contains link to a malicious web site used as part of a scam; we use the term *phishing* for the method of luring users to a scam web site by spam e-mail. Scam web sites usually trick users into providing them with sensitive, personal information, in different ways; one of the most common is by misrepresenting (`spoofing`) the scam web site as a trustworthy web site, e.g. a large bank, and asking users to login to their account, thereby collecting user accounts and passwords. See solutions to web-spoofing in [HG04]. Some of the sensitive information collected in this manner may be of direct financial value, e.g. credit card numbers and other details, which the attacker abuses, among other things, to finance spamming operations (e.g. purchase new domain names). Attacker can use other personal information, such as names, e-mail accounts and areas of interest, to form new spam messages to this user and users related to her, increasing the effectiveness of the scam and making automated filtering harder. Finally, such web sites may try to cause

execution and/or installation of malicious programs such as viruses on the client's machine, by including them as active content on the web page (scripts, applets, etc.), or by convincing the user to download them. This is similar to execution or installation of malicious content attached to the spam, as we discuss next.

4. Spam messages often contain malicious programs (malware) such as viruses and Trojan-horse programs. Such malware, running on the user's computer, often use this computer's resources, including e-mail accounts and e-mail address books, to send additional spam messages. Finally, malware (`spyware`) and malicious websites (often spoofed sites, i.e. clones of `good` sites), often collect personal information, used for fraud (esp. identity theft) and for composing harder-to-filter spam messages.

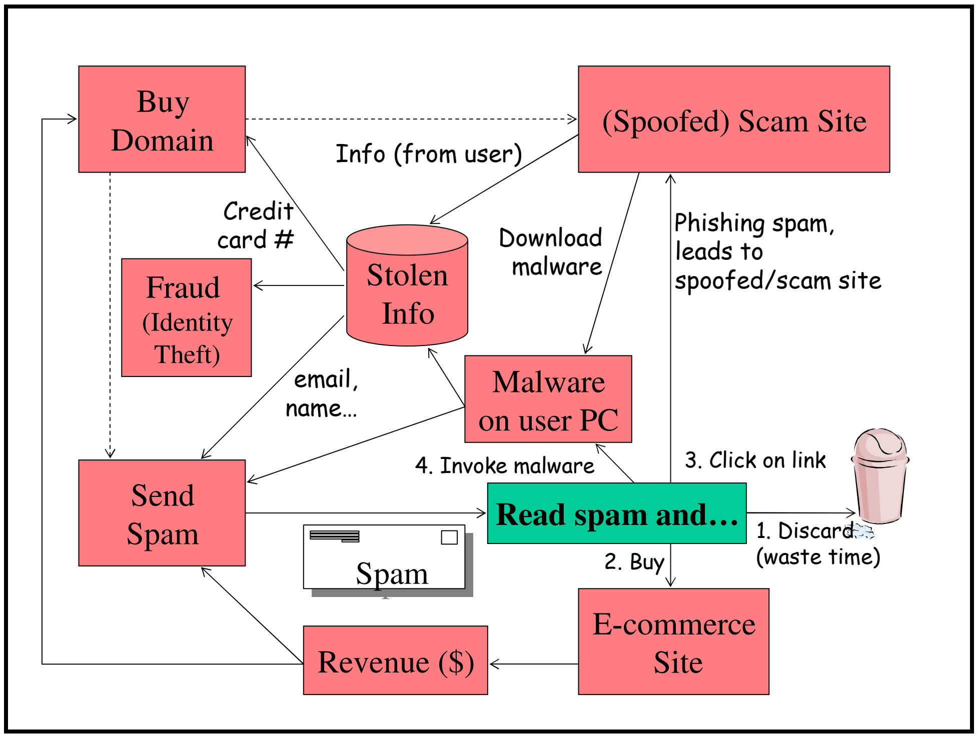

Figure 1: The Vicious Spam Cycle

2 Secure Internet Content Selection in a Nutshell

We begin by describing the basic operation of the SICS protocol, in typical scenarios; we postpone more detailed discussion of advanced features to later sections. Internet content selection involves at least two entities: an originator, e.g. *Alice,* and a recipient, *Bob*. Alice sends some content or message *m* to Bob. To help Bob decide if and how to use *m*, Alice attaches to *m* a *content label l* (or *rating*). Before the message reaches Bob, it passes a *Content Filter (CF)* agent. Bob defines to the Content Filter some *policy*, specifying acceptable vs. unacceptable content labels; the content filter should discard or retain messages with unacceptable content labels, and deliver to Bob only messages with `desirable` content labels. The content filter CF may also add some additional label, e.g. classifying messages based on likelihood of being spam.

For simplicity, we assume that all parties agree on a universal mapping (`oracle`) *Label* from messages to their `correct` content labels (extensions to multiple mappings are trivial, e.g. see `Rating Systems` in PICS). However, content-filtering agents cannot compute *Label* efficiently and precisely (if such an `ideal` content filtering software is possible, SICS is not needed, of course). The content filter may use some efficient estimate of *Label*, such as content-based heuristics, to confirm the content label. Therefore, the content filter may still err and pass to Bob messages that arrive with incorrect content labels. A message *m* sent with incorrect content label *l≠Label(m)* is called *spam*; message *m* sent with correct label *l=Label(m)* is *non-spam*. An e-mail message is *unlabeled* when it does not contain any content label (i.e., `legacy` e-mail message).

To encourage correct labeling (and discourage spam), it is critical to *endorse* the content label, by attaching a cryptographic *content label authenticator* (or simply *authenticator).* We call the endorsing party a *Rating Service (RS)*, and denote the authenticator by A_{RS}; the authenticator is the result of public key signature or shared-key Message Authentication Code (MAC), applied by the Rating Service (the preferred choice depends on the scenario, as we explain later).

The Rating Service may generate and endorse (authenticate) the label *l* on its own, or endorse (authenticate) a label received, with the message, from the sender, possibly after some `sanity check` and/or modification. The Rating Service could be provided by the sender's outgoing mail server, e.g. to support labeling when the sender runs legacy e-mail software (without SICS support). The Rating Service could also be done on the sender's machine (e.g. when the sender and recipient both use SICS-compliant systems, while the sender's ISP does not use SICS-compliant systems, does not trust the sender, and/or is not trusted by the recipient).

Now that we introduced the sender (Alice or A), recipient (Bob or B), Rating Service (RS) and Content Filter (CF), it may be helpful to refer to Figure 2, which presents an overview of the Secure Internet Content Selection (SICS) protocol. The BL (black list) and CA (certification authority) parties are optional; and the dashed arrows are used only when a `false negative` labeled message arrives. Further details follow.

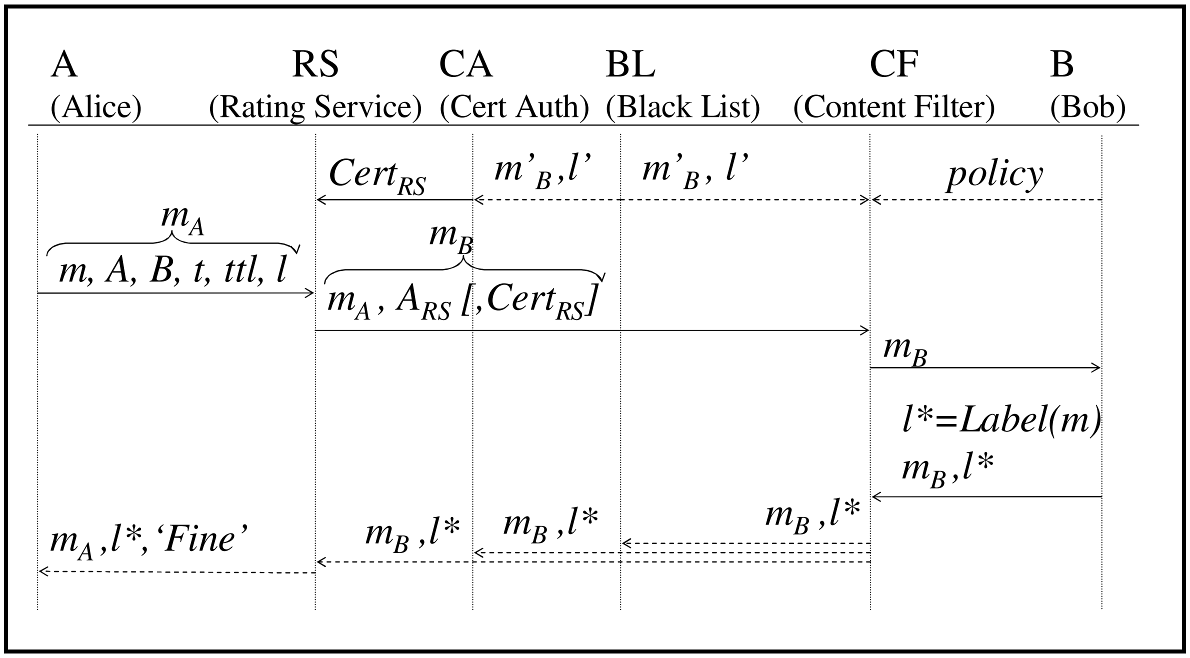

Figure 2: Overview of the Secure Internet Content Selection (SICS) Protocol

Alice sends to the Rating Service the message *m,* the identities of Alice and Bob (*A* and *B*), the current time *t*, the maximal acceptable delivery time *ttl* and the content label *l*. Let m_A denote the set of all the fields sent by Alice, i.e. $m_A=\{m,A,B,t,ttl,l\}$. The label *l* and time to live *ttl* are optional (if absent, the RS adds them).

The Rating Service *RS* may locally validate (or generate) the label *l*, and attaches the *authenticator* A_{RS}. The authenticator signals the endorsement of the label *l* by the rating service *RS*. We use the following **dot notation:** fields (e.g. keys) belonging to a party are denoted by the name of the party followed by dot and the name of the field, e.g. the secret signature key *s* of *RS* is denoted by *RS.s*. The authenticator is therefore

A_{RS}=$Sign_{RS.s}(m_A)$, when using digital signatures. We later discuss efficiency improvements allowing the use of shared key message authentication code (MAC) instead of digital signatures.

The *RS* next sends the complete e-mail message, together with the attached authenticator, to the recipient, using the standard e-mail protocol (SMTP). The *RS* may also attach to the message its public-key and/or attribute certificates, $Cert_{RS}$= $Sign_{CA.s}(RS.v,attr)$, using X.509 or another certificate format. The certificate contains the public signature-validation key *RS.v* of *RS*, and a list of attributes *attr* assigned to *RS* by a *Certification/Attribute Authority,* which we denote *CA*. The attributes *attr* describe properties that *CA* attributes to *RS*, e.g. `this ISP has been spam-free for the last year`; we later show how *attr* could be used to establish relationships between *CF* and *RS*, when *CA* acts as a trusted, guaranteed payment service provider.

The certificate is needed when the *CF* does not trust the *RS*, and in particular when receiving a message with a label from a Rating Service operated by an individual sender, which the recipient (or *CF*) do not trust. The certificate should convince *CF* to accept the content labels authenticated by *RS*. Usually, this implies financial risk to *RS* if it endorses (authenticates) false labels (spam), and/or compensation to Bob and/or *CF*.

When receiving the message m_A , authenticator A_{RS} and optionally certificate $Cert_{RS}$, the *Content Filter (CF)* validates the authenticator A_{RS} , the time-to-live *ttl* and the label *l*. Only if all of these are Ok, and in particular if Bob's policy is to receive messages with label *l,* then *CF* passes the message to Bob.

If Bob finds out that this message is spam, i.e. *l* is incorrect, then he should complain and send the correct label *l*=Label(m)* to *CF*. To support legacy recipients, *CF* may add text to the message (possibly in separate MIME part) instructing Bob how to manually file the complaints, e.g. by simply replying to the message (with the contents of the message); SICS enabled mail agents will automate this functionality. Bob sends the complaints all the way to the sender (Alice), but makes sure it passes via *CF* and *RS*; to support legacy senders, the complaints will also contain text in simple language explaining that Alice sent spam. The *CF* forwards the complaints to the *RS* and possibly also to the *CA*. As result of complaints, the *CF* may reduce his trust of the *RS* and the *CA*. If the *RS* (and/or the *CA*) is committed to pay some compensation for spam, then the *CF* demands such compensation for itself and/or for Bob. Let us consider here only the simplest case, where the *RS* pays some pre-agreed compensation amount to the *CF* for each falsely authenticated label (spam). We deal with the other scenarios later. Therefore, in this case, falsely authorized content label (spam) is a direct financial risk to the rating service *RS*.

In theory, the rating service *RS* could eliminate all risk of paying penalties due to having authorized spam (false content labels), simply by (manually) inspecting each message for validating or for assigning it a correct content label. However, this is clearly too expensive (and very intrusive on sender's privacy). Therefore, the rating service *RS* must manage some risk due the potential of authorizing spam inadvertently. To allow *RS* to manage this risk, we limit its liability and financial exposure to spam messages identified as such before their time-to-live (*ttl*) expired. Rating services could use different mechanisms to protect themselves from the risk due to messages whose time-to-live is still valid. This includes (manually) inspecting some messages, i.e. evaluating *Label(m),* on a random basis (which may be expensive and intrusive), using heuristic approximation of *Label(m)* and requiring sufficient deposit from the sender (Alice) for compensation in cases of spam. We believe this last option would be appropriate for many ISP-customer relationships. In another common scenario, *RS* is simply a software agent of the sender (Alice); in this case, the only risk to *RS* (or, more precisely, to Alice) is if the *RS* receives messages from attacker acting as Alice, e.g. a virus (we explain later how Alice can manage this risk). In any case, when *RS* receives a complaint claiming *m* was spam sent by Alice, it also forwards this complaint to Alice.

When detecting a false digitally-signed label, the content filter *CF* may also inform one or more *Black List (BL) servers.* The Black List servers provide pairs of `falsely signed spam` to content filters and certificate/attribute authorities. This allows the content filter to ignore the labels and certificates authenticated by non-trustworthy rating services and certificate authorities, and the *CA* to revoke (or not extend/issue) certificates for non-trustworthy rating services.

Mail servers and clients may provide one or more of the content filtering and rating services. The sending mail client, his outgoing mail (SMTP) server and other mail server along the route, including the recipient's incoming mail (POP/IMAP) server, can provide rating service (RS). Similarly, any mail server, as well as software running on the recipient's machine, could provide the content filtering service. The mail servers, and other trustworthy entities, could also provide the black list (BL) service and the certification (CA) services.

3 Secure Content Selection: Design Criteria

We begin by identifying the design criteria and requirements from a secure (internet) content selection solution, roughly in order of (diminishing) importance. We believe that it is most critical to ensure usability and acceptability of the solution to e-mail users and providers, and to identify the following ***usability and acceptability criteria****:*

- **Legacy support: no interference with `legacy` (unlabeled) mail:** the solution should allow complete, transparent interoperability of the content-selecting recipient user agent with all `legacy` senders and mail servers (which do not support content labels). Similarly, content-labeling by sending user agents and/or by mail transfer agents (servers), should not cause any difficulties to `legacy` standard mail recipient user agents (which do not support content labels).

- **Smooth migration path:** any labeling solution cannot provide substantial value until there are at least two adopting entities (one making a label and the other using it). However, this should be sufficient; the solution should provide value immediately when any two of the parties involved in mail transfer adopt it (including sending and receiving user agents, and any of the mail transfer agents / servers). Preferably, the solution should also offer some advantages when adopted only by a single entity (this, in fact, is achieved by our solution).

- **Minimize manual user work:** the solution should make the minimal requirements from the users. An example of a reasonable requirement is to ask users to provide feedback on false negatives (undetected spam), and of course to provide (simple) mechanisms to allow users to define their filtering policy. An example of unreasonable burden, is requiring SICS senders to know whether the recipient supports SICS.

- **Standard mail clients support: allow use of existing mail clients, with minimal new UI, and no usage change, including e-mail addresses, mailing-lists, etc.:** the design should allow customers to continue using existing e-mail clients without change in usage or in e-mail addresses. In particular, the design should allow users to send mail to multiple recipients, and should have minimal or no impact on mailing lists.

- **Allow any legitimate content (including explicit advertising and `potentially offensive` content):** a major concern with many existing spam controls is that they may cause, inadvertently or intentionally, censorship of legitimate content [DGH*04]. Furthermore, complete prevention of advertising content may not always be desirable, and may interfere with society goals (under some conditions, advertising may be acceptable or even desirable), and with legitimate, desirable business models (e.g. providing free networking services in return to adding advertisements to the mail content). It is therefore essential that the content selection mechanisms restrict content strictly on the basis of the recipient-defined policy, and not using some mandatory or third-party (`big brother`) defined limitations.

- **Easy interoperability across providers (without central authority):** to ensure interoperability among all content-labeling senders and/or servers, and all content-filtering recipients and/or servers, it is desirable to make it easy for different content selection and filtering service providers to interoperate, including rating services, content filtering services, and black list services. Like many other problems related to security and trust, e.g. payment systems, a centralized solution where a single entity is responsible for rating, filtering and blacklisting is simpler to implement and deploy, provided all users use this particular entity (or services chartered by the `central` entity). However, this assumption may not be reasonable, and certainly is only an option for a single, determined (and probably well funded) entity; we also believe that such a centralized solution is not in the benefit of society, and in particular it may be abused and result in censorship.

Next in importance are requirements of ***security***, i.e. making it impossible or not economical to send content with false labels (spam), and protecting all conforming (honest) participants.

- **Facilitate use of e-mail security and privacy mechanisms (encryption):** while, as noted before, currently e-mail security (and in particular encryption) is not widely used, definitely secure content selection mechanism should not prevent users from protecting (e.g. encrypting) their e-mail. In particular, the solution should not force senders to make their messages readable to any mail servers along the route to the destination. On the contrary, it is desirable that security mechanisms introduced for content selection, would also facilitate other security features for e-mail, such as encryption and authentication.

- **Mail received should (usually) conform to the recipient's policy:** clearly, the content filtering should block almost all of the messages that do not conform to the policy defined by the recipient. It is probably impossible to completely block all non-conforming messages, when allowing sufficiently flexible and powerful policies.

- **Prevention of `Framing` conforming participants:** the SICS protocol, as well as some other approaches, includes `penalties` for malicious, non-conforming parties, by fines and/or by blacklisting. It should not be possible for an attacker to `frame` an innocent, conforming party, i.e. cause it to be `punished`.

- **Allow filtering and rating by intermediate mail servers as well as recipient:** spam wastes resources of all mail servers it passes through, therefore the protocol should allow all of them to filter out spam (messages with false or unacceptable content labels). Intermediate mail servers may also provide filtering and rating as a service to the recipient or to the mail servers along the path to the recipient.

- **Limit the damage to and from broken-into computers:** a realistic spam solution cannot ignore the sad reality that currently most computers connected to the Internet are quite insecure, and as a result, spammers can obtain control over numerous computers belonging to honest, naïve, users (see Figure 1). The solution should provide security in spite of this unfortunate situation, taking into account that many of these users will not notice even significant levels of overhead due to malware activity, and many may even willingly allow spammers to use significant computing resources (e.g. to enjoy free use of some `shareware`). On the other hand, the solution should limit damages to the owners of such broken-into computers.

- **Limit the damage due to breach of trust:** content selection always depends on some trust among the participants. In SICS, the content filter *CF* needs to trust either *RS* or *CA* to some extent; in particular if *CF* is obliged to pay some compensation to Bob for each unfiltered spam, then it may need to limit the damage due to *RS* endorsing spam (and refusing to compensate *CF*). The protocol should allow the operator to set monetary bounds for damages from such breach of trust, and inform the operator whenever some entity breaks the agreements between them.

Finally, we mention ***performance and implementation criteria***.

- **Performance:** the solution should have minimal, acceptable overhead in computation, communication, messaging or otherwise.

- **Stateless, replicable services/servers:** it should be possible to implement all services, including content filtering and rating services, without requiring the server to maintain any state for messages. Namely, servers should be able to perform correctly, based only on the information they receive, and some general local storage, which the server uses for all messages, senders or recipients. This allows better efficiency and use of multiple servers for scalability and resiliency.

4 Bootstrapping SICS: Providing Value to Early Adopters

In the overview description of SICS in Section 2, we assumed that SICS is deployed by (at least) the content filter (*CF*) and the rating service (*RS*); furthermore we assumed that the content filter (*CF*) trusts, and has the public key, of the rating service (RS) and/or of the certification authority (CA). When SICS is established, we expect that these assumptions will be usually or often valid. However, an important design goal is to plan a `bootstrap process` for SICS, providing value to early SICS adopters; this is the subject of this section.

We recommend that SICS server and client deployments would always contain both Rating Service and Content Filter services. In this section, we show how this will allow any two SICS-enabled parties along the path from sender to recipient, to use SICS rating and content-filter services. However, in the first and second subsections, we show that even when only a single party along the path from sender to recipient deploys SICS, then there are significant value to this party from SICS. In the third subsection, we explain how a SICS sender and recipient can establish trust, even in the absence of a trusted certification authority.

4.1 Sending SICS-enabled e-mail to new or legacy (non-SICS-enabled) recipient

We first consider the scenario where the sender, or the sender's outgoing mail service (usually operated by the sender's ISP), is SICS-enabled, but the recipient may not support SICS. Even if the recipient does not use SICS, there may still be a value for the sender or her ISP to run SICS, to solve the following problem. Currently, e-mail clients and ISPs often receive `spam complaints` from different spam filters and e-mail recipients, containing spam messages sent using the client's e-mail address. However, e-mail source addresses are easy to spoof; it is therefore difficult for the sender, and/or her ISP, to determine whether the complaint is the result of an actual spam sent by the client (possibly by a virus on the client's machine), or the result of a spoofed source address (possibly generated by a virus on another machine). This distinction is important, to warn the client when her computer is infected, and to penalize clients who intentionally (or due to negligence) send spam.

Similarly, SICS could be used by the recipient's mail server, or by intermediate mail servers, to confirm `spam complaints` regarding mail sent via the mail server. This allows the mail server to confirm, easily, the mail server from which the spam came, more securely and conveniently than by consulting the content of the complaint and the log files kept by the mail server (which is the current practice). This will allow more accurate black list reporting and other penalty mechanisms, encouraging mail servers to be more careful in preventing spam thru them.

Let us consider the case of a SICS-enabled mail server. When the SICS-enabled mail server receives an e-mail message without a SICS label, it attaches a SICS label as a MIME attachment (`part`) to it before transferring it towards the recipient. If the original message contained a multipart MIME type, the SICS label is simply added as an additional part of the existing multipart MIME type. Otherwise, when the original message contains only a single (MIME or ASCII) part, then the server transforms the message into multipart MIME and includes the original message as one part, and the SICS label as another part. The SICS label itself consists of the following parts:

1. A simple textual message identifying this as a SICS label and recommending the use of SICS-enabled mail reader.

2. A SICS rating for the content, based on the agreement between the server and the server or client from whom it received the e-mail, and possibly on the result of a content-based filter run by the server. Typically, this will be a basic, broad indicator, e.g. `no advertisement`.

3. A (possibly compressed) copy of the original message headers, allowing precise identification of the server or client responsible for it. If the server maintains the headers in log files, it may simply include an index to the entry here.

4. Hash values for the two parts above of the SICS label, and for all the other parts in the (original) message.

5. A digital signature over the hash values, using the private signing key of the SICS-enabled server.

6. Optionally, one or more certificates of the server, and/or links to repository containing such certificates.

When a SICS-enabled mail server or client receives a message containing a SICS label, it immediately knows that this message was sent by a SICS-compliant sender (or server). If the certificates and cryptographic methods are acceptable, it may begin immediately filtering messages based on the SICS ratings. Otherwise, it can communicate with the SICS sender (or server) to negotiate acceptable credentials and cryptographic methods, as we discuss in subsection 4.3 below.

Formatting for interoperability with standard e-mail mechanisms. The definition of categories and formats for specifying the policy in the e-mail message, or in a separate document identified in the e-mail message, are beyond the scope of this work, and may use formats such as PICS labels [MRS96, MK*96], or the policy format of [CSRI04].

4.2 SICS deployed by recipient or recipient's mail server

We next consider the case where the recipient, or his mail server, is first to deploy SICS functionality, i.e. the content filtering service. For simplicity, we describe SICS (content filter) deployment by the recipient (Bob) himself; deployment by the recipient's mail server is essentially the same. Namely, we describe the behavior of a SICS-enabled e-mail client application, used to receive mail.

Upon initialization, a SICS recipient generates a secret key k for a pseudo-random function f, e.g. implemented by a block cipher such as AES or DES. It then uses this key, to generate pairs $n,f_k(n)$ where n is a unique identifier (counter or random), and f is a pseudo-random function. Then, either Bob himself or Bob's SICS-enabled mail reader, provide one pair to each new correspondent. SICS-enabled senders can use n as an identifier and $f_k(n)$ as a shared key with which they secure initial communication with Bob, as we discuss in the next subsection. Other senders can use the pair $n, f_k(n)$ as a `ticket` to one or more messages to Bob.

To use the pair $n, f_k(n)$ as a ticket, Bob – or his e-mail reader – provide the pair to the sender, typically by encoding it as part of a special e-mail address for Bob, which this sender should use (similar to [GJMM98]). Specifically, [RFC822] allows e-mail addresses in the form *phrase "<" addr-spec ">"*, where *phrase* is any sequence of words, which is ignored by the mail servers, and *addr-spec* is the `simple` e-mail address, e.g. *Bob@mail.org*. Therefore, we can encode $n, f_k(n)$ as one or two words in *phrase*. The length of n should suffice to ensure uniqueness, e.g. 32 bits, and the length of $f_k(n)$ should suffice to prevent guessing, e.g. 64 bits; with typical encoding, the length of the resulting pharse is in the order of 20 characters, which seems acceptable.

Bob's SICS-enabled e-mail client will use such an e-mail address for Bob, encoding a ticket $n, f_k(n)$, when sending mail to a recipient who may not be SICS enabled; if this recipient uses this address when replying to Bob, it will usually automatically contain the ticket. Bob can also provide manually his e-mail address with a ticket encoded as a phrase, e.g. by creating appropriate business cards (each containing a unique $n, f_k(n)$ ticket). Bob can specify how many messages, and possibly which kind of messages, are allowed from each unique address (or class of addresses he defined); once receiving spam from an address, it is invalidated and future messages from it are ignored. Or, if Bob still trusts that the sender is not a spammer, and the spam was by someone else (e.g. copied on a note from the sender to Bob), then Bob instructs the SICS-enabled mail reader to generate new ticket n', $f_k(n')$ and send it to the sender.

A SICS-enabled mail reader will normally not pass to Bob messages received without a ticket (or an appropriate SICS-label). Bob may allow some exceptions; in particular, Bob may want to permit incoming messages without a ticket, if the sender address appears in Bob's address book or if Bob earlier sent a message to this sender; usually such `waiver` will be removed upon identification of spam from this sender. When an incoming message without a SICS label or ticket is blocked, the SICS-enabled mail reader may respond by sending back a `ticket`, i.e. a pair $<n, f_k(n)>$, and a request to re-send the message together with the ticket (possibly by hitting `reply`). This will confirm, at least, that the sender is able to receive e-mail as well as to send e-mail, getting rid of much of the spam (using spoofed source address). Of course, such an address may still belong to a spammer, and Bob's e-mail reader may be more suspicious regarding messages coming from it, e.g. running more critical content-based filter, or restricting the number of such messages delivered to Bob during the same day.

4.3 Establishing `Web of Anti-Spammers`, or: trust without a certification authority

In the previous two subsections, we argued that SICS may provide some (limited) benefits even if deployed only by a single participant (sender, recipient or mail server) along the e-mail path. In this subsection we consider the case where two (or more) of the participants along the path have installed SICS enabled mail agents. In Section 2, we explained the operation of SICS between a rating service and a content filter that trusts it. We also briefly explained how the content filter can establish trust in an unknown rating service, using a certificate stating that this is a trustworthy non-spammer, from a trusted certification authority. In this section we show two methods for the content filter to establish trust in a rating service (or sender), without requiring a trusted certification authority.

The first method extends the technique in the previous subsection, namely the use of a per-sender e-mail address for the recipient, , e.g. $<n, f_k(n)>$ *Bob@mail.org*, where $n, f_k(n)$ is a *ticket* which the content filter generates using the secret, random key k (and the known pseudo-random function f). As before, the user may transfer the per-sender address (containing the ticket) manually to the sender, e.g. printed on `customized` business cards. Alternatively, the content filter may send the per-sender e-mail address in e-mail to the sender, possibly with a request to re-send the original message with a SICS label or ticket.

When a SICS Rating Service, run by the sender or ISP, receives an e-mail address $<n, f_k(n)>$ *Bob@mail.org* containing a SICS ticket, it considers $k'=f_k(n)$ as a shared secret key with the recipient *Bob@mail.org*; it then uses k' to authenticate the messages, and SICS labels, it sends to Bob, as explained in Section 6.

This simple mechanism allows pairs of SICS-deploying users and ISPs to establish a `secure pipe` between them, preventing spoofing of e-mail; it therefore allows the content filter to establish more and more trust in the rating service's SICS labels (as long as Bob doesn't authenticate spam!). In particular, the rating service RS uses k' to authenticate a special SICS message to Bob, containing the public signature-validation key of the rating service $RS.v$, authenticated by a message authentication code $MAC_{k''}(RS.v)$ using key k'' derived from k' as in Section 6. This allows Bob to gain trust in $RS.v$. This can be useful if the rating service RS, e.g. owned by Alice, starts using another client, to which she transfers her private key but not the shared secret key.

A possibly more important use for *RS.v* is to validate certificates signed by the rating service (or Alice) signature key *RS.s,* e.g. `non-spammer` certificates to persons trusted by the rating service (or Alice), in a `PGP-like` web of trust [Z95]. We now explain how users can use such user-issued certificates to share the trust observations with each other, to create a `web of trust` allowing automated, secure establishment of trust in SICS labels between individuals, building on the existing social networks connecting them.

Specifically, SICS client and server implementations will offer users to *publish signed (certified)* public keys of (all or selected) trusted rating services (including these belonging to specific organizations and individual senders). SICS implementations publishing public keys of trusted rating services, may sign either the entire list (for efficient signing and validation) or sign each public key separately (for more compact certificates). Preferably, the signed list or certificates include details on the level of trust in each rating service (e.g. details on the amount of non-spam communication with it). To allow other SICS agents (mainly content filters) to use these public keys, we place such (signed) lists in public `repository` servers. In particular, SICS implementations may publish keys of trusted rating services by placing them (signed) in a file on the Web, with the address of the file provided to correspondents.

Each content filter may run a process as in Figure 3, following [HM04], to determine whether it can trust the SICS labels from SICS rating service, on the first time it receives a SICS label from it. This process begins with the public key and the rest of the certificate sent by the SICS rating service. The content filter uses this to look up additional certificates in one or multiple collections (repositories), e.g. in files or directories containing lists of certificates of trusted organizations and individuals. The user can specify some mandatory collections which must always be searched, and the SICS label may specify specific collections of certificates for the sender and for the certificate authorities of the sender. If the certificates contain attributes (labels) which are not of the types used in defining the trust policy/rules of the content filter, then the content filter runs an ` Attribute extractor`, mapping all of the labels to a common `language` of attributes used by the policy/rules of the user. The output of this is a list of triplets *<IPK, SPK, Attrs>*, where *IPK* is the issuer's public key, *SPK* is the subject's public key (the subject being the rating service, or one of the certification authorities), and *Attrs* is the list of attributes (from the vocabulary used by the user). Finally, the content filter runs a policy engine, using policy or rules defined by the user, to decide whether to trust the SICS label or not (and possibly the level of trust).

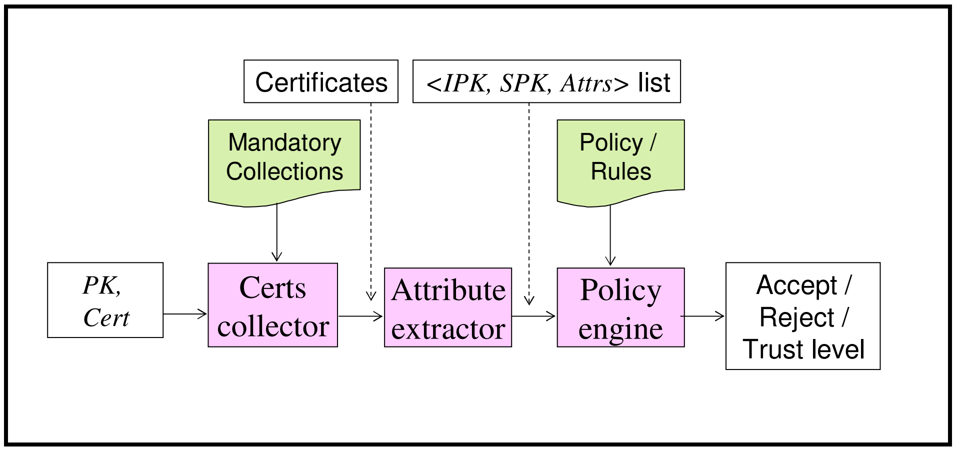

Figure 3: SICS Content Filter Assessment of Trust in a Rating Service

SICS Negotiation. We now discuss the scenario, where the content filter *CF* receives a message which passed via a SICS rating service *RS,* but does not have sufficient or appropriate authentication information. This could be due to incompatible choice of cryptographic algorithms or policy, or due to the usage of keys that the *CF* does not trust. These may include shared secret keys which *CF* does not have (e.g. due to cache expiration), and ratings or certificates issued by entities not trusted by *CF*. In all of these cases, *CF* simply returns the e-mail with an error message, indicating its requirements from a valid authenticator. Alice's rating service *RS* should simply resend the message together with an appropriate SICS authenticator.

If *RS* cannot meet *CF's* requirements, then it may pass a textual message to Alice, suggesting a list of acceptable rating services that Alice can use to obtain necessary credentials. Such rating services may charge Alice a deposit and fee for their services, e.g. using a credit card transaction.

4.4 Multiple Rating and Filtering Services from Sender to Recipient

Our description so far focused on a single rating service *RS* and a single content filtering service *CF*. However, we expect that often several mail servers along the route may want to perform content filtering (i.e. block spam). SICS supports this scenario easily.

We first note that multiple independent *RS-CF* relationships may exist along the route from Alice to Bob. For example, Bob may run a local content filter CF_B, and trust endorsements by RS_A, a rating service run locally by Alice; at the same time, Bob's ISP may run its own content filter CF_{BISP} and trust endorsements by RS_{AISP}, a rating service run by Alice's ISP. The two relationships are independent, and unaware of each other. This only requires the protocol to provide appropriate format, which will make CF_{BISP} ignore the labels and authenticators attached by RS_A. Also, note we take advantage of the fact that the content filters send complaints all the way back to the sender (Alice), therefore they reach every content filter and rating service along the path.

In a more complex case, multiple content filter services use the same content label and endorsement from the same rating service. For example, both the recipient (Bob) and his mail server, require incoming mail messages to contain a (signed) SICS label. Suppose Alice's ISP generates the SICS label. Since the content filtering by Bob's mail server forwards the message together with its label, Bob can also run a content filter on his machine. Furthermore, the guaranteed payment mechanism described in the previous sub-section easily extends to such cases (the label simply needs to include both agreements and identifiers).

5 Supporting Untrusted Senders and Rating Services with Guaranteed Payments

In the overview of SICS which we gave in Section 2, we mentioned that the content filter *CF* might send back to the rating service *RS* any false negatives, i.e. messages received with incorrect labels from the *RS*. One reason for that is that the agreement between the *CF* and the *RS* may require the *RS* to pay some predefined monetary compensation for any false negatives (undetected spam). However, this requires *CF* to make such an agreement with *RS*, establish appropriate payment mechanism, and trust that the *RS* would in fact pay. There are cases, where there is a natural relationship between the *RS* and the *CF*, which fulfills these requirements; for example, when the *RS* is a `spam-filter` service of the *recipient's ISP*, or when the *CF* is a `spam-filter` of the *sender's ISP*.

However, often there is no natural trust and/or payment relationship between the *CF* and the *RS*; for example, when the recipient's ISP runs *CF*, and the sender's ISP provides the rating services, where the two ISPs are remote and not very large. An even more extreme case is when *RS* is an agent on Alice's machine, which is the likely case when Alice is not a customer of the outgoing mail server (e.g. in Internet café), or during the early adoption phases of SICS when Alice's ISP may not yet provide rating service. Finally, especially at early adoption phases, Bob's incoming mail server may also not (yet) provide SICS content filtering services.

Therefore, in such cases, requiring the *CF* to establish trust and payment relationships with the *RS* is undesirable, and may limit interoperability and cause substantial overhead (e.g. to process such small payments and establish trust relationships, between two remote ISPs). In particular, while it may be possible, in theory, to use the rating service's certificate to prove its liability to a third party, such as a judge, it is not practical to actually use the existing legal system to extract compensation from the rating service. In this subsection, we explain how to *automate* the compensation for spam to the content filter and recipient, by using *guaranteed-payment certificates*.

A *guaranteed-payment certificate authority (GPCA)* is a special kind of *CA,* which *CF* trusts, to make payments per an agreement between them. This implies that *CF* could trust ratings signed by *GPCA*, i.e. *GPCA* could be a good Rating Service for *CF*. However, even if *GPCA* could establish appropriate trust relationships with *RS*, this would turn *GPCA* into an additional rating service, and it would have to handle every message, introducing substantial processing overhead and delay. We prefer a solution where *GPCA* is involved only in exceptional situations, mainly when *RS* endorses falsely labeled e-mail (spam).

Therefore, we propose that *GPCA* is contacted only once per period by the *RS,* as a special kind of certificate/attribute authority. However, the *CF* and the *GPCA* must first establish an *agreement*, allowing the *CF* to `honor` certificates signed by the *GPCA* (knowing that the *CPCA* will compensate it for spam). The agreement identifies the following fields:

1. An identifier (`account`) for *CF*, denoted *CF.id,* which rating services must include explicitly in the (signed) label; this ensures that *CF* cannot ask for compensation for messages not endorsed specifically for it.

2. A public signature-validation key *GPCA.v* of the *GPCA*. This allows validation of signatures by the *GPCA* (on certificates to rating services *RS*).

3. An `compensation amount function` identifier *caf*, which identifies the compensation amount for different combinations of a (signed) message *m,* with a (signed) label *l,* as a function *caf(l, l')* returning a non-negative monetary value, where *l* is the SICS label sent and *l'=Label(m)* is the correct label for *m,* where *caf(l,l)=0.* The agreement between *GPCA* and *CF* will list out the meaning of each compensation function *caf*. Some functions could be very simple, e.g. a fixed amount for each (incorrectly labeled) spam message, or a definition of an amount for regular messages and a higher for messages marked as `high priority`. Other functions may depend on specific fields in the label *l*; for example, a rating service could allow labels to specify a monetary amount of compensation for spam (e-mail with incorrect label), and the compensation amount function could specify modifications to this amount, e.g. handling fee for *GPCA*.

4. A period during which this agreement holds and a maximal amount of compensation by *GPCA* to *CF* for false negative ratings, over the entire period. This allows *GPCA* to control its risks due to an agreement, similarly to the use of the *ttl* field with messages (to control the risks of the rating service).

5. An agreement identifier *agreement.ID*, which *RS* includes in the (signed) labels for which this agreement applies. This allows *GPCA* to use multiple agreements concurrently with *CF*, e.g. to handle the case where one agreement expires or when the maximal amount of compensation in it is reached.

The process of sending e-mail with guaranteed-payment certificates is quite similar to the process described in Section 2, when *RS* attaches a certificate from a `regular` certificate authority *CA*. Namely, *RS* attaches to the message a certificate $Cert_{RS}= Sign_{GPCA.s}(RS.v,attr)$. However, we have some specific requirements from the attributes *attr* in the certificate. In particular, *attr* must indicate the identity and/or public key of *CF*, i.e. *CF.id* and/or *CF.v*, a maximal liability amount per validity period and a validity period. The rating service *RS* should also include in the authenticator $A_{RS}=Sign_{RS.s}(m_A)$, or simply in the label *l* (which is part of m_A), the values of *CF.id* and *CF.v*. We also assume that labels are unique (if necessary, add a serial number), to prevent `double deposits`.

In this solution, the *GPCA* needs the ability to transfer money from the *RS* to the *CF*. This may be simple, e.g. by requiring *RS* to pre-deposit an amount with *GPCA*. However, to support interoperability on a global scale, we need to consider the case where there is no long-term, direct trust relationship between the *GPCA* and the *RS*. We can solve this if the *GPCA* has long-term trust relationship with some other guaranteed payment authority *GPCA'*, which has trust relationship with *RS*. Namely, the *GPCA'* makes an agreement *a'* with the *GPCA,* who uses this to make appropriate agreement with the *CF*. This extends efficiently, allowing an arbitrarily long `chain of trust` between guaranteed-payment service providers, linking the rating service *RS* and the content filtering service *CF*. For details, see the `payment routing protocol` of [H03].

The guaranteed payment services allow SICS to provide another useful service, namely monetary compensation to recipients and processors of properly marked (commercial) e-mail messages. Some legitimate, respectable corporations may be interested in such a mechanism, whereby they pay a small, predefined amount to the recipients of their advertisements and/or to the mail servers involved in processing them. Moreover, guaranteed-payment services may be useful for other payment applications, unrelated to e-mail, e.g. person to person payments and micropayments; see [H03].

6 Using Shared Keys for Efficiency and Privacy

We now describe how SICS can establish and use shared keys between two mutually trusting parties, for improved efficiency and for ensuring privacy. When using shared keys, the recipient of spam, i.e. a message that violates the specified content label, cannot `prove` the violation to a third party such as a *GPCA*, *BL*, or *CA*. However, this is acceptable, when there are long-term trust relationships between the recipient and the rating service that provided the false label. Such long-term trust relationships often exist between two users (Alice and Bob), between users (Alice/Bob) and their ISP, and between many ISPs.

The idea is to use a shared key authenticator, based on Message Authentication Code (MAC) function, instead of using a digital signature authenticator as described so far. Namely, instead of using $A_{RS}=Sign_{RS.s}(m_A)$, we use as authenticator $A_{RS}=MAC_k(m_A)$ where *k* is a key shared between the rating service *RS* and the content filter *CF*. The key *k* can be sent by the rating service *RS* or manually by the recipient, as described in subsection 4.3; later on, we also show how the parties can establish the key *k* using known public keys. However, we first point out that this idea may require both the rating service and the content filter to preserve state for each labeled message, which is clearly a substantial, undesirable overhead. We therefore begin by exploring the need for state (if using the above solution) and showing a simple fix (that allows stateless rating service and content filter).

To see why the $A_{RS}=MAC_k(m_A)$ solution requires state, we examine the trust relationship among the parties more carefully. Clearly, the parties trust each other only to a limited extent. In particular, it seems reasonable that the content filter would trust the rating service as per their agreement for compensation in cases of correctly reported falsely labeled content (spam), since the content filter can detect quickly any violation of this trust. However, it does *not* seem reasonable that the rating service will trust the content filter to report false labels (spam) accurately. In fact, have we used $A_{RS}=MAC_k(m_A)$ without state in the rating service, then the content filter could have modified the message and sent back $A'_{RS}=MAC_k(m'_A)$ where m'_A is spam. As a result, the *CF* would receive (unjustified) compensation from the rating service, and/or caused the rating service to suspect (and possibly black-list) the sender (Alice), without any way for the rating service to detect this fraud. We solve this easily, by adding another authenticator, using another secret key known *only* to the rating service, denoted *RS.selfkey*. Namely:

$$A_{RS}=\{MAC_k(m_A), MAC_{RS.selfkey}(m_A)\}.$$

Note that if there are several content filtering services along the route, using the authenticators from the same rating service *RS,* as discussed in the subsection 4.4, then the authenticator may need to include a MAC_k for each *RS-CF* relationship.

We next show how the parties establish the shared key or multiple keys, either using different algorithms or using the same algorithm (e.g. to limit the exposure of the same key); let *k[i]* denote the secret shared key with identifier *i.* Once *RS* and *CF* share a secret key *k[i], RS* can simply compute the authenticator A_A using a message authentication code (MAC) with this shared key instead of using a digital signature, as described above; this saves resources to both services. *RS* should also send an identifier *i* for the current key (identifying also the algorithms, similarly to the SPI – Security Parameters Index – in IPSec [CG*98]).

To establish the shared key *k[i]* and identifier *i,* the parties must either agree on a shared, secret `master key` k_m, or use their public keys. In both cases, the parties could use a secure connection, e.g. using TLS, or a standard key-setup protocol such as IP-Sec IKE, or implement a simple key-setup protocol. The content filter may invoke this protocol upon detecting that the message passed thru a rating service, by appropriate indication in the message (e.g. special header field).

We can also use the shared secret key to encrypt the content of the messages between the two parties, providing support for confidentiality as well as for secure content labeling and authentication.

7 Conclusions and Discussion

The Secure Internet Content Selection (SICS) protocol, which we presented in this paper, is a relatively simple cryptographic protocol, which may aid in controlling spam, including different forms of messages sent with undesirable content and with misleading representation and labels. The protocol meets the many design goals presented, and in particular it `punishes` spammers, or more precisely rating services that endorse messages sent with false, inaccurate or misleading labels. This is important, as usually spammers have a financial incentive, which they may share with seemingly-disinterested rating service, creating an incentive to endorse spam. Recipients, by defining appropriate policies to their content filter, can define an arbitrarily large compensation for spam. In reality, we expect that eventually most ISPs and e-mail client developers will agree on some reasonable default values, e.g. based on the pricing scheme of sending airmail letters of different priorities (corresponding to e-mail messages marked with different priorities).

The biggest obstacle to the acceptance of SICS, and many other of the more advanced spam controls, may be the fact that it requires cooperation between (honest) senders and recipients, and/or their mail servers. Indeed, there are many aspects in the design, described in Section 4, which are specifically targeted to support the gradual migration and adaptation, and provide value even to early adopters.

Another significant obstacle may be our requirement of `punishing` spammers, mainly by financial penalties. This may require modification to the agreements between Internet and e-mail service providers and their customers, especially to ensure a deposit to cover fines for possible future spamming. We believe, however, that this process is inevitable, especially in light of the current insecurity of most computers connected to the Internet these days. Indeed, from discussions with some Internet connectivity providers, it turns out that many of them already apply financial penalties to spammers, typically by closing spamming accounts (with a financial implication).

We also believe that most users will agree to set a reasonable limit for messages sent daily, which translates into limited spamming value and limited damage to the consumers as well as to spam recipients and e-mail servers carrying the spam. In fact, we hope and believe that many users will appreciate the value of becoming aware of a penetration to their computer, by their ISP informing them of spam originating from their computer. Growing user awareness may also result in development and adoption of better computing security mechanisms. In this way, SICS may help reduce the number of insecure computers connected to the Internet, thereby `breaking` the `vicious cycle of spam` illustrated in Figure 1. Internet security may further benefit from the message authentication and confidentiality (encryption) facilities, based on SICS mechanisms for establishing shared secret keys between senders and recipients. In particular, this may make it more difficult for e-mail viruses to propagate and will help to prevent other e-mail spoofing, phishing and eavesdropping attacks.

Acknowledgement

The author developed some of the basic ideas in this work together with graduate student Jonathan Levi, at 2002-2003 [L03], and Jonathan also provided many helpful comments to this manuscript. Ron Rivest encouraged the author to complete the unfinished work following the discussion in [DGH*04]; thanks for this (necessary!) push.

This work benefited from many fruitful discussions on the cryptography@metzdowd.com mailing list, including different ideas and proposals related and similar to ours, in particular see [DGH*04]. We wish to thank the owner and moderator, Perry Metzger, and the many participants. Special thanks to John Gilmore, whose criticism in [DGH*04] was instrumental in improving the design; I definitely hope that SICS, or whatever spam solution is eventually adopted, will not result in the kind of censorship John experianced. Free, uncensored and spam-free communication is important to all of us.

This work was support in part by National Science Foundation grant NSF CCR 03-14161 and by Israeli Science Foundation grant ISF 298/03-10.5.

References

[CG*98] Pau-Chen Cheng, Juan Garay, Amir Herzberg, and Hugo Krawczyk, "A security architecture for the internet protocol," IBM Systems Journal, Special issue on the Internet, vol. 37, no. 1, pp. 42-60, 1998.

[CSRI04] The Coordinated Spam Reduction Initiative, Microsoft corporation, February 2004.

[D02] Hadmut Danisch, A DNS RR for simple SMTP sender authentication, Internet-draft, online at http://www.danisch.de/work/security/antispam.html, first version December 2002, current version October 2003.

[DC*98] Philip DesAutels (Ed.), Yang-hua Chu, Brian LaMacchia and Peter Lipp, PICS Signed Labels (DSig) 1.0 Specification, W3C Recommendation, http://www.w3.org/TR/REC-DSig-label, May 1998.

[DGH*04] Victor Duchovni, John Gilmore, Amir Herzberg, Ben Laurie, Perry Metzger and others, messages to the cryptography@metzdowd.com list on the topic ` why "penny black" etc. are not very useful (could crypto stop spam??)` , January 2004.

[DGN03] Cynthia Dwork, Andrew Goldberg and Moni Naor, On Memory-Bound Functions for Fighting Spam, Advances in Cryptology - CRYPTO 2003, LNCS 2729, Springer, 2003, pp. 426-444.

[DN92] C. Dwork and M. Naor, Pricing via Processing or Combating Junk Mail, Crypto ’92, pp.139 – 147, 1992.

[GJMM98] Eran Gabber, Markus Jakobsson, Yosi Matias, Alain Mayer, Curbing Junk E-Mail via Secure Classification, proceedings of Financial Cryptography, pp. 198-213, Feb. 1998.

[H98] R.J. Hall, Channels: Avoiding Unwanted Electronic Mail, to appear in Communications of the ACM. Also available at URL: ftp://ftp.research.att.com/dist/hall/papers/agents/channels-long.ps.

[H03] Amir Herzberg, Micropayments, pp. 245-280, chapter 12 in `Payment Technologies for E-Commerce`, Editor Prof. Weidong Kou, Springer-Verlag, ISBN 3-540-44007-0, 2003.

[HG04] Amir Herzberg and Ahmad Gbara, Protecting (even) Naïve Web Users, or: Preventing Spoofing and Establishing Credentials of Web Sites, DIMACS technical report 2004-23, April 2004. A more updated version is available online in http://eprint.iacr.org/2004/155/ (pdf) and in http://www.cs.biu.ac.il/~herzbea/Papers/ecommerce/spoofing.htm.

[L03] Jonathan Levi, MAPS – Mail Authentication and Prevention System, Master thesis proposal, computer science dept., Bar Ilan University, March 2003.

[M98] Kevin McCurley, Deterrence Measures for Spam, presented at the RSA Conference, January, 1998.

[M99] Removing the spam. Email Processing and Filtering. ISBN 0-201-37957-0, Addison-Wesley, 1998.

[MK*96] Jim Miller (Ed.), Tim Krauskopf, Paul Resnick and Win Treese, PICS Label Distribution Label Syntax and Communication Protocols Version 1.1, W3C Recommendation, http://www.w3.org/TR/REC-PICS-labels, October 1996.

[MRS96] Jim Miller (Ed.), Paul Resnick and David Singer, Rating Services and Rating Systems and Their Machine Readable Descriptions Version 1.1, W3C Recommendation, http://www.w3.org/TR/REC-PICS-services, October 1996.

[RFC822] David Crocker, Request For Comments 822, Standard for the Format of ARPA Internet Text Messages, http://www.ietf.org/rfc/rfc822.txt, August 1982.

[SG98] Alan Schwartz and Simson Garfinkel, Stopping Spam, O'Reilly & Associates 1998.-

[UBE-DEF] Paul Hoffman, Internet Mail Consortium Report, Unsolicited Bulk Email: Definitions and Problems, IMCR-004, October 5, 1997. http://www.imc.org/imcr-004.html

[UBE-SOL] Paul Hoffman, Dave Crocker, Internet Mail Consortium Report, Unsolicited Bulk Email - Mechanisms for Control, IMCR-008, May 4, 1998. http://www.imc.org/imcr-008.html.

[Z95] Phil R. Zimmerman. The Official PGP User's Guide. MIT Press, Boston, 1995.

Appendix: Overview of Spam Controls

Spam is a major nuisance to e-mail users and providers. As a result, there are many deployed and proposed measures for preventing spam, or at least controlling its proliferation; we refer to such means as *spam controls*. A thorough review of spam controls is beyond the scope of this work; in this sub-section, we provide a very brief and incomplete discussion.

Many e-mail recipients and servers use different mechanisms to eliminate, or at least reduce, the amount of spam e-mail they receive and handle. Recipients usually choose the mechanism as a function of the specified source of each message. In particular, recipients often separate between the following categories of incoming e-mail messages:

- Messages from predefined and trusted sources (senders or mail-servers), vs. messages from other sources.
- Messages from sources that use standard e-mail clients or servers, vs. messages from sources who use e-mail clients or servers which are enhanced with spam control mechanisms.

For each category, there are different spam-control mechanisms that the receiver can apply.

We show some of the most important spam-controls mechanisms and proposals in Table 1, including the SIPS protocol that we present in this paper. In the table, we categorize spam controls using two criteria. The first criterion is the use of cryptographic mechanisms; most deployed spam-controls do not use cryptography. We believe that cryptography is essential for solid spam control, and present cryptographic spam controls, which may complement non-cryptographic spam controls. The second criterion is the use of some trusted entity; most cryptographic solutions, including the SICS protocol, require or can take advantage of some trusted entities (e.g., rating services and black list services).

There are substantial differences in the level of protection against spam among the different mechanisms. In particular, messages from sources with spam-prevention mechanisms verify as spam-free with much better accuracy than messages from standard mail clients and servers. However, as long as spam controls are not widely deployed, most e-mail recipients have to accept mail also from standard e-mail clients (without spam controls). We anticipate that as spam controls becomes more and more popular, e-mail recipients will give precedence to messages from senders with spam-control mechanisms, and many recipients will not agree to process messages from senders without spam controls.

Currently, well designed use of the available techniques, and in particular of site/domain blacklisting and of content filtering, are quite effective against current spam. For example, Duchovni [DGH*04] claims that while 1000 spam messages pass his blacklist blocking mechanisms, only 75 of them (7.5%) pass his content filter, which leaves an acceptable level of less than three spam messages per day. However, Gilmore [DGH*04] responds that this is partially due to false positive filtering – in particular, Gilmore's messages are categorized as spam by this filter.

We now describe (very briefly!) each of the known spam control mechanisms in Table 1, grouped by the categories used in the table.

7.1.1 Non-cryptographic Spam Controls without Trusted Entities

The most basic defense against spam does not use cryptography, assumes no spam control by the sender, and allows incoming e-mail from unknown, untrusted senders. Under these severe limitations, the ability to prevent spamming is limited, although some of these technologies are able to handle current spam levels quite well. These techniques are very important, since the recipient can invoke them, without assuming adoption of any spam controls by the sender. Techniques in this category include the following:

- *Pattern, Heuristic and/or Human Content filtering:* these are techniques that inspect the actual contents of the e-mail message, including its header fields (e.g. \`from:\`, \`subject:\`), and try to identify spam – advertising, offensive or malicious content. Content filters may look for identifying patterns (\`signature') of known, common spam messages, such as a sequence of bytes that identifies a virus program. Alternatively, content filters may use various heuristics to identity spam, e.g., by identifying the frequency and pattern of use of some words which are common in certain spam messages (\`Super\`, \`win\`, \`Discount\`,…). Unfortunately, spam creators are well aware of the filtering mechanisms, and are often able to modify the spam messages to avoid identification by content filtering (e.g. by intentionally introducing typos and breaks in important words, e.g. \` Su.per Dis.count\`). Content filters are also prone to identify legitimate e-mail as spam (false positive), which could have serious consequences[2]. Of course, some recipients use \`human content filters\`, in the form of assistants who try to filter out spam; this is hard to beat for reliability (but has its own drawbacks).

- *Return address validation:* in this technique, the recipient of e-mail (from unknown senders using standard e-mail without spam controls) *always* returns the message to the sender, with a brief message asking her to re-send the same message (often with minor change, e.g. send to a different e-mail address). The goal of this is simply to validate that the e-mail was sent from a valid e-mail account, which can receive the response; as noted, many spammers use invalid source addresses, so they will normally not be able to respond. Of course, this technique, like blacklists, does not help if the spammer uses a valid e-mail account, which is quite common. Note also that this solution causes substantial annoyance to the sender of the e-mail, who may fail to re-send the message (undesirable) or install anti-spamming e-mail enhancement that will automate the re-transmissions (desirable). Also, note that it introduces additional latency and overhead (retransmissions, synchronization of address books, change of address, etc.).

- *Proof of human work:* this technique replaces or complements the \`special recipient address\` technique, by forcing the sender to perform some \`human\` task, which is hard to perform by a computer program.

[2] This happened to the author, whose messages – using a valid but unusual e-mail address – were blocked repeatedly by a customer's anti-spam filter, with no indication.

The idea is that a spammer will have to spend considerable resources for performing these tasks, making spamming not (or at least considerably less) profitable. There are different proposals for tasks which are `easy for persons but hard for programs`. For example, the task could be simply provision of answer to one of few simple questions (and answers), entered manually during the installation of the software (e.g.: `what are the first three letters of the alphabet? `. A possible drawback of these methods is the burden on senders, some of whom may refuse or fail to complete the task.

7.1.2 *Non-cryptographic Spam Controls Based on Trust and Trusted Entities*

- *Unlisted e-mail address:* here, the recipient gives her e-mail only to known, trusted peers, and never publishes it, hoping that it will not be known – or used – by spammers. This is a very common, low-tech solution, but not very secure. It is often combined with generation of `disposable` alternate e-mail addresses, used in less-private forums (and disposed if abused by spammers).

- *Unique unlisted address per trusted sender:* this technique combines the `return address validation` and `unlisted e-mail address` techniques described above; the goal of this extension is to perform `return address validation` only on the first e-mail received from any particular mail sender (to reduce the annoyance to the sender). To facilitate this, the recipient sends, in his reply to the sender, a special e-mail address, which also reaches the recipient, but contains also a unique identifier allowing the recipient to identify this particular sender. The special e-mail address may be generated automatically, possibly using a cryptographic mechanism as described in [GJMM98, H98].

- *Senders whitelist:* in this technique, the recipient maintains a list of *legitimate* e-mail senders. E-mail from these senders is accepted (without additional filtering), while e-mail from other senders is discarded or filtered as described above. E-mail recipient often use this technique to minimize the risk of false positives for e-mail, when using content filtering. However, this technique fails whenever the sender changes her e-mail account. Also, since viruses are used by spammers to collect legitimate e-mail addresses, often from address books, this technique could fail if the virus found the correct addresses (e.g. in address book of a joint friend).

- *Senders / domains blacklist:* in this technique, the e-mail recipient or her mail server compares the e-mail address of the mail sender, or the domain (DNS) name / IP address of the sender's mail server, against blacklists of suspected e-mail accounts, hosts, domains and network (IP) addresses. Such blacklists, e.g. RBL™ and other lists by mail-abuse.org, often use DNS also to distribute records. However, currently there is no widely-adopted secure and reliable mechanism for managing or trusting the blacklists. Furthermore, an attacker can use spoofed messages to avoid the blacklist or 'frame' a honest server, causing it to be entered to the blacklist. Hence, blacklists work well when the e-mail comes from a trust-worthy mail server. A trustworthy mail server validates the authenticity of the sender's e-mail address for mail originating from it, and relay mail only from other trustworthy e-mail servers, with known IP addresses and domain names. However, attackers are able to open new e-mail accounts in different free services, open new domains with very little cost, and obtain many different IP addresses (e.g. in Internet cafes). Indeed, the `black list` method may make Internet connectivity more expensive and difficult, to (honest) users in areas with substantial spammer activity (e.g. some undeveloped countries). Furthermore, recently spammers often use broken-into machines to send spam; it is not realistic to add all of these machines or their mail servers to the blacklists.

- *Identification of (trusted) source domain (`caller-ID`, domain whitelist):* this approach prevents e-mail spoofing by typical (spoofing) attackers, by establishing that the mail server sending the message is authorized to send mail from the specified source domain. Technically, a mail server receiving e-mail validates the IP address of the sending mail server, against records indicating the authorized IP address of the outgoing mail server of the domain (distributed via DNS). The mail servers of these domains are trusted to take reasonable measures against spoofing, spam and other illegitimate content in e-mail messages. There are several proposals following this approach; one of the earliest appears to be [D02]. In one of the most recent and important proposals, Microsoft [CSRI04, section 10] proposes that the DNS records also point at to `white-lists` containing conformant domains, maintained by different organizations (the dual solution to the blacklists of `bad` domains). Notice that the spoofing defense fails against intercepting/eavesdropping adversaries, who can e.g. spoof the DNS replies; worse, such `strong` adversaries could mount denial-of-service attacks against victim domains, by making it appear that the victim domain is issuing spam. We can prevent such spoofing by using digital signatures, message authentication codes or secure connections such as SSL/TLS or IP-Sec tunnels, to ensure the e-mail really comes from the specified source domain; in particular many mail servers try to communicate with peers using SMTP over SSL/TLS, even without proper certificates/PKI (opportunistically). However, two serious weaknesses remain. First, this approach assumes that the source mail servers will implement such policies properly. Unfortunately, there is a potential financial incentive for mail servers to be permissive and allow (some of) their customers to actually send spam or other illegitimate mail, and it could be impractical for mail recipients to detect this or `punish` such mail source servers. Second, if this approach becomes popular, few entities that maintain the `white-lists` may exclude non-spamming domains for technical or even unrelated (e.g., political) reasons, potentially causing them to be cut off from much of the Net. See [DGH*04].

7.1.3 Cryptographic spam controls

We now discuss proposed cryptographic spam controls (except SICS which we describe in previous subsection and following sections).

- *Source authentication:* in this scenario, the recipient trusts the sender; furthermore, both uses cryptographic-enhanced e-mail services. Specifically, the sender authenticates the e-mail sent, typically using shared-key authentication (Message Authentication Code – MAC) if sending to a single recipient, and public key digital signatures if sending to multiple recipients. This solution provides excellent protection against e-mail spoofing, and therefore also against spamming. The basic cryptographic mechanisms for authenticating and validating e-mail messages are available in most e-mail client agents, and integrating them with anti-spam controls should be straightforward, when there is a long-term relationship between sender and recipient. A similar solution could be deployed for cryptographic authentication between the source and destination mail servers (domains); this is discussed above together with the (more common) non-cryptographic identification of (trusted) source domain.

- *Proof of computational work:* this technique, proposed in [DN92, DGN03, CSRI04], replaces or complements the `return address validation` requirement, by forcing the sender to perform substantial amount of computations (possibly used towards some computational goal). The idea is that a spammer will have to spend considerable resources for performing these computations, making spamming not (or at least considerably less) profitable. One concern with this approach is that spammers, who send their spam from penetrated computers (see Figure 1), may not care much about the computational cost, while innocent users end up wasting their computer resources (on every e-mail sent by their computer, legitimate or spam). Another concern is that this technique may prevent sending e-mail from computationally limited or energy-sensitive client devices, such as some mobile devices; this concern may be partially addressed by the techniques of [DGN03].

- *Pay to send:* these proposals [M98] require payment for each message sent, but they use traditional, monetary payments, instead of payments using computational work as in [DN92, CSRI04, DGN03]. In addition, some propose that if the e-mail is legitimate (not spam), then the sender will be paid back – either by the fact that the recipient will normally respond to the e-mail (and pay for this response), or by the recipient declaring that the mail was not spam. The main objection to this technique is that the processing of payments and `rebates` for non-spam will introduce substantial overhead, inefficiencies, objections by consumers and high costs of customer relationship. Our proposal can be seen as extending the `pay to send` approach, but changing it slightly to `*pay to send spam*` - as senders of non-spam are not charged at all.

- *Message signing with E-mail Policy Indication* [CSRI04, sections 8-9]: this proposal suggests that e-mail message contain an E-mail Policy Indication field (similar to our `content label`), digitally signed by some trustworthy authority. Our SICS protocol also uses signed content labels (using the PICS term for what CSRI calls `e-mail policy indication`), but deals also with inadvertently or maliciously incorrect labels. Gilmore and others [DGH*04] argue that there may be financial incentives for label-signing authorities to be negligent or intentionally to sign incorrect labels. Furthermore, by dealing with incorrect labels explicitly in SICS, we improve openness (support for many independent authorities), and allow reduction in the computational overhead. SICS was developed independently and in parallel to CSRI[3].

[3] Early versions of our work were reported in [DGH*04, L03].