Arbeitsbericht (Technical Report) 07/01

Preventing E-mail Spam: The Conceptualization and the Analysis of an Infrastructure Framework

Dr. Guido Schryen

schryen@winfor.rwth-aachen.de

Abstract

Spamming remains a form of Internet abuse, which burdens the Internet infrastructure, is generally regarded as an annoyance, and is said to cause a huge economic harm. Many technological, organizational, and legislative anti-spam measures have already been proposed and implemented, but have not led to any substantial decrease in the number of spam e-mails. We propose a scalable and flexible infrastructure framework that integrates several anti-spam measures and that features both a technological and an organizational facet. The key element of our infrastructure is a new organizational unit that reliably and transparently limits the number of e-mails that can be sent per day and per account. We also analyze the proposed framework in terms of its theoretical effectiveness, the required resources, and its limitations.

Key words: Email, Infrastructure, Internet, Spam

Contents

List of Figures

List of Tables

1 Introduction

1.1 The problem

Those of us using the Internet e-mail service are faced almost daily with unwanted messages in our mailboxes. We have never asked for these e-mails, and often we do not know the sender, and we puzzle over where the sender got our e-mail address from. The types of message vary: some contain advertisements, others provide winning notifications, and sometimes we even get messages with executable files, which finally transpires to be malicious codes, such as viruses and Trojan horses. It is sobering to note that, for many years now, statistics have shown that the number of spam e-mails exceeds the number of "regular" e-mails (ham e-mails) (MessageLabs 2006, Symantec 2006). Senders of bulk e-mail benefit from the anonymity that is inherent to the e-mail infrastructure (Klensin 2001): sender data can easily be spoofed, and remotely controlled PCs can be used for sending e-mails (OECD 2004b). The design principles of the e-mail infrastructure, which were originally intended to provide simplicity and flexibility, have since proved to be ambivalent characteristics.

Today, spam has even crossed the borderline between simply being annoying for private users and causing economic harm. For example, companies invest money in anti-spam software and IT staff, and they lose the productivity of employees when these spend time opening, reading, classifying e-mails as spam, and then deleting them (NOIE 2002, Brightmail 2003, Nucleus Research 2003). Private users, on the other hand, lose money due to fraud e-mails, including phishing attacks (OECD 2004b). The worldwide economic harm caused by spam is estimated at several billion USD per year (Ferris Research 2005, OECD 2004a, Gauthronet and Etienne 2001).

There are a number of approaches in use for managing unsolicited bulk e-mail, which is termed "spam"[1]. The anti-spam approaches comprise technological, legislative, organizational, and behavioral measures. With regard to technological measures, organizations mostly employ filtering technology, use authentication methods, or/and construct elaborate rules that determine which senders are allowed to connect or deliver e-mail to their networks and which are to be blocked (whitelists, blacklists). Filters, which are the most frequently deployed type of technological anti-spam measures, are usually rule-based (Cohen 1996, Crawford et al. 2001), signature-based (Damiani et al. 2004, Zhou et al. 2003), or Bayesian-based (Graham 2002, Graham 2003, Pantel and Lin. 1998, Sahami et al. 1998). When using these, we have merely heuristics on hand, that sometimes misclassify e-mails: whereas a spam e-mail in our mailbox might not seem so bad, an e-mail that has been erroneously classified as spam and goes, therefore, unnoticed, does. In a case like this, an anti-spam measure is actually counterproductive. Authentication methods include SMTP extensions (Myers 1997, Myers 1999), cryptographic exten-

[1]The etymology of the word "spam" is usually explained by reference to an old skit from Monty Python's Flying Circus comedy program (for example, see Merriam-Webster's Collegiate Dictionary).

sions (Leibzon 2005, Tompkins and Handley 2003), and path authentication (DeKok 2004, Leibzon 2005). They all reasonably aim at identification and accountability, but they still suffer from conceptual, practical or/and pragmatic drawbacks (Leibzon 2005, Schryen 2006a). Like filtering methods, the usage of blacklists and whitelists are heuristics, too. While these are appropriate weapons against repeatedly used nodes, they do display a shortcoming, as spammers tend to change their IP addresses continually, either by switching to other ISPs or by taking advantage of exploits on unsuspected third party nodes.

The huge economic relevance of spam has motivated the national authorities of many countries and of federal states to address spam also by legislation (ITU 2005, OECD 2004c). However, despite some spammers being prosecuted, the effectiveness is limited today, because (1) world-wide legislative coverage of spam is heterogeneous (Moustakas et al. 2005) and (2) e-mail messages do not contain enough reliable information for them to be traced back to their true senders. A general problem of legislative measures against spam e-mails is that an international phenomenon is being addressed by national legislation. Beside technological and legislative anti-spam measures, organizational and behavioral measures have been proposed, such as abuse systems and international cooperation (OECD 2004b)) and methods for the protection of e-mail addresses from being harvested respectively (e.g. by embedding an address into a picture). Like other approaches, these have not led to any substantial decrease in spam yet. They can probably be regarded as "flanking" measures only.

Although policies and technological measures can be effective under certain conditions and can help to maintain Internet e-mail as a usable service, over time, their effectiveness will degrade due to increasingly innovative spammer tactics. Moreover, today's deployment of anti-spam measures resembles a (still open-ended) arms race between the anti-spam community and the spammers. Even worse, we generally allocate resources of the recipients of e-mails towards fighting spam, instead of increasing the senders' need for resources. This far, no single measure has proved to be the "silver bullet" against spam, and it is doubtful whether any single, simple solution will ever be able to reduce or stop it. Rather, it seems appropriate to look for solutions that provide a complementary application of several anti-spam measures.

1.2 The contribution of this paper

The infrastructure framework presented in this paper features such a complementary application. It is intended to have the following characteristics, which we assume to be preconditions for effectiveness in the long run and for a widespread adoption by the Internet e-mail community, which includes ESPs, other sending/receiving organizations, e-mail users, and Internet authorities:

- Both technological and organizational modifications must be minor. The OECD (2004b, 14.) sums it up: *"A solution to spam should not make the Internet so cumbersome to use that people stop using it. The cure should not be worse than the disease."*

- An openness must be present, insofar as the framework provides for principles, and not for concrete algorithms or data formats.
- Spam should be stopped as close to its true source as possible. The prevention of spam has a higher priority than does its detection.
- Means to support the sending of solicited bulk e-mails have to be provided.
- The deployment of the key elements can be done smoothly and flexibly, i.e. the adoption of the infrastructure can occur evolutionarily.
- The infrastructure must not undertake an “arms’ race” with spammers (for example, filters do engage in such a race).

This paper is structured as follows: Section 2 provides an overview of the framework and of the interaction of its key components. The framework includes both organizational and technological elements, which are discussed in detail in Sections 3 and 4 respectively. The theoretical effectiveness of the framework is then assessed in Section 5, whereas in Section 6, the framework’s resources are analyzed quantitatively. Deployment issues are covered in Section 7, before this paper closes with a consideration of the limitations and drawbacks in Section 8.

2 Overview of the framework

The core ideas of the framework are (1) to limit the number of e-mails that can be sent during a specific time-window and per account[2], (2) to restrict the automatic set-up of e-mail accounts and (3) to provide means for controlling this limitation of e-mail traffic by introducing an element of centralism.[3] In order to support these ideas, a new organizational role is introduced: the Counter Managing & Abuse Authority (CMAA). The framework is intended to include several organizations, each of them taking on the full CMAA role. These organizations are either new and designated ones or established ones, such as trustworthy ESPs. In our framework, in principle, an SO, for example an ESP, either directly transmits an e-mail to the RO or sends the e-mails to a CMAA organization, which then relays the message to the SO. The former option is today’s default option for sending e-mails, but is intended to be used in our framework only if the receiving organization (RO) trusts the sending organization (SO) with regard to the implementation of effective anti-spam measures. Otherwise, the latter option applies, which means that the CMAA first checks whether the sender would exceed the number of e-mails he or she is allowed

[2]The implementation of rate limits on outbound e-mail traffic is part of “Best practices for e-mail service providers”, which are proposed by many organizations, such as the Anti-Spam Technical Alliance (Anti-Spam Technical Alliance (ASTA) 2004).

[3]In principle, the framework follows the idea that a credit of, for example, 100 messages per day is a very large number for an individual, but an inconsequential number for a spammer. It also aims to prevent a compromised account or infected host from sending spam to millions of recipients in a short time frame.

to send on one day. Depending on the result, the CMAA would then either bounce the e-mail or relay it to the RO, whereby any CMAA organization offers a relaying service.

This replacement of the direct SMTP connection between the SO and the RO by a relaying procedure represents an element of centralism, which allows for controlling and accounting the (volume of) e-mail traffic. This control is intended to enormously reduce the sending of unsolicited bulk e-mail. Solicited bulk e-mail may still be sent if a person or organization accepts (legal) responsibility for a proper usage. The (anti-spam) control is also intended to make additional anti-spam measures undertaken by ROs obsolete. As the control mechanism is unlikely to prevent all spamming, it seems reasonable to complementarily provide a forum for e-mail users' complaints about unsolicited e-mails. Therefore, every CMAA organization is intended to also operate a central anti-spam abuse system. The abuse system and the relaying system are connected to each other in that numerous complaints about the spamming activities on behalf of a specific sender may lead to the blocking of that sender's CMAA account and, thus, to the bouncing of further e-mails from this sender. For the rest of this paper, we use the shorter term CMAA for "CMAA organization", unless we explicitly provide the term CMAA to designate the organizational role.

An important feature of the framework is the option of the SO to send an e-mail directly to the RO in order to reduce a CMAA's workload. However, whether an e-mail that has not been relayed and counted by a CMAA is accepted by an RO depends on the RO's policy, which could include a dynamic white list of trustworthy SOs. This alternative procedure, which is today's standard in e-mail delivery, makes the framework flexible and scalable in both its operation and deployment.

In order to implement the accountability, on which the framework is based, the SO sets up a record for each sender's e-mail account prior to the first relaying. The records are stored in a database, herein denoted as Counter Database (CDB). As a CMAA is also responsible for the locking of accounts due to abuse complaints, these complaints are stored in another database, herein denoted as Abuse Database (ADB). A third database, the Organization Database (ODB), serves for the storage of information about those SOs that are registered on the CMAA for the usage of its services. Figure 1 illustrates the infrastructure framework. For the purpose of simplification, those infrastructure elements that are responsible for the administration of the databases are omitted. They are presented in Section 4.

In order to successfully tackle spammers' needs to send a huge number of e-mails, possibly millions of them, some obvious requirements have to be fulfilled, which are addressed in the following two sections in detail:

- The records have to be protected from (illegal) manipulation. This imposes stringent requirements on the database and system security.
- The set-up and removal of records is restricted to trustworthy parties only, such as trustworthy ISPs and SOs. Furthermore, due to the expected high

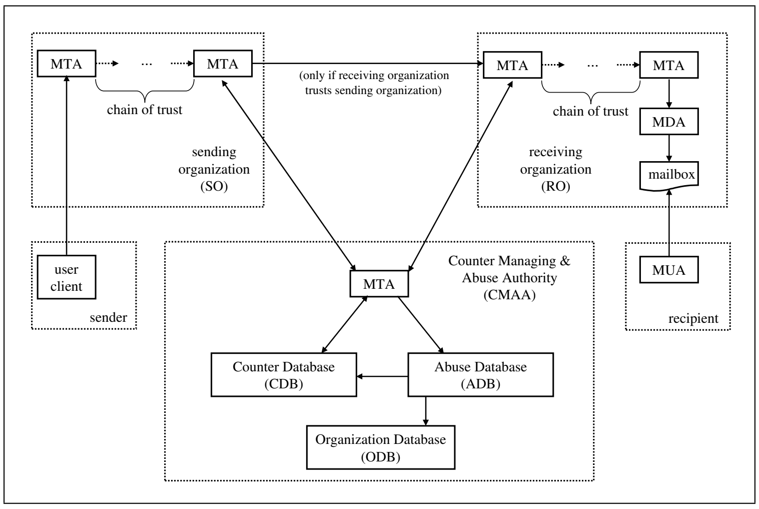

MTA
...
MTA
chain of trust
sending organization (SO)
(only if receiving organization trusts sending organization)
MTA
...
MTA
chain of trust
receiving organization (RO)
MDA
mailbox
user client
sender
MTA
Counter Managing & Abuse Authority (CMAA)
MUA
recipient
Counter Database (CDB)
Abuse Database (ADB)
Organization Database (ODB)

Figure 1: Overview of the infrastructure framework

number of "adding" or "deleting" transactions, these operations must be supported by automatization.

- The third parties that are allowed to use the services of an CMAA, including the adding and the removal of specific records, have to fulfill a bundle of requirements:
  - They must ensure that the accounts they are responsible for cannot be set-up automatically, thus preventing spammers from using an arbitrary number of accounts.
  - They have to take care that their systems are secured against misuse, i.e. an attacker must not be able to manipulate the CDB on behalf of the respective organization.
  - They have to ensure that their e-mail users need to use mechanisms that protect their accounts from being misused as well as possible, e.g. by using a password for sending an e-mail.
- The account-specific credits must be low enough to prevent unsolicited bulk e-mail, but high enough to not disturb solicited e-mail communication. This requirement includes the provision of means for regular bulk e-mailers.

The introduction of additional (logical or physical) organizational units (CMAAs) that are responsible for the implementation or the control of the described tasks requires organizational, technological, and financial support and, therefore, seems to be unnecessary and even counterproductive. However, some reasons support its appropriateness:

1. The operation of CMAA (role) services is critical for the success of the framework and requires both the willingness and the technological ability to operate a CMAA properly. Organizations that reside in developing countries or in countries with a non-restrictive anti-spam environment may only improperly fulfill these requirements; organizations that are notorious for addressing spam only lackadaisically are likewise unqualified. A CMAA that is operated and controlled by a trustworthy organization seems to be much more appropriate for providing the required services.

2. The list of trustworthy organizations is CMAA-specific and is maintained by each CMAA. The administration of decentralized white lists and blacklists by ROs would become obsolete. Each organization receiving an e-mail that has been relayed (and counted) by a CMAA can assume that the SO is a trustworthy one. Therefore, ROs would only be required to maintain data for all CMAAs, such as IP lists of trustworthy Mail Transfer Agents (MTAs).

3. The infrastructure will not eradicate spam, but should support an abuse system. Currently decentralized abuse systems could be consolidated by integrating this service into the portfolio of the CMAAs.

Although the framework makes demands on SOs and, therefore, seems to resemble reputation-based approaches, such as LUMOS or the sTDL approach of Spamhaus (Email Service Provider Coalition 2003, ICANN 2004), it differs from them in two main issues:

- The presented reputation-based approaches keep the e-mail communication direct, i.e. the SO directly SMTP-connects with the RO. It is the RO that has to prove the reputation or accreditation of a particular SO. In contrast, our framework provides an additional organizational unit, which relays e-mails, and, thus, makes the communication indirect. Therefore, with our approach, it is not up to each RO to prove the reputation of a particular SO; this is a CMAA's task.

- With our approach, the SO's fulfillment of requirements is not sufficient for the successful delivering of a message. In addition, a restriction on the remaining account-specific credit applies.

However, as with reputation-based approaches, it remains an important task to formulate a set of requirements for SOs, which are effective regarding the misuse of a CMAA's services and the fulfillment of which can be verified. Because of this importance, these issues are addresses in Subsection 3.4 in detail.

3 Organizational solution

The framework involves technological as well as organizational modifications to the Internet e-mail infrastructure and the e-mail processes. The organizational modifications, which are addressed in this section, result from the introduction of a new organizational role: the CMAA. As mentioned above, the framework is intended to involve several organizations each of them taking on the full CMAA role. However, a few outstanding key questions must be addressed prior to implementation and deployment:

1. Who will operate a CMAA?
2. How is a CMAA certified and by whom?
3. Which CMAA is responsible for which organization?
4. How does an organization register for the usage of CMAA services?

These issues are addressed in the following subsections. However, the organizational structure of the framework is simple and illustrated in Figure 2.

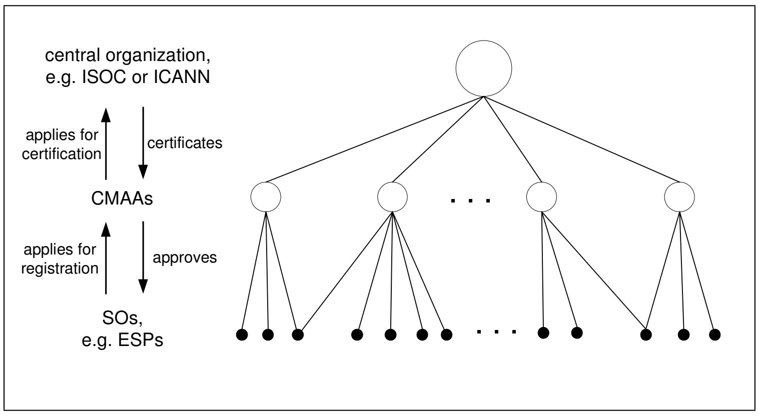


Figure 2: Organizational structure of the infrastructure framework

3.1 Integrating CMAAs into the Internet

The introduction and the maintenance of a new organizational role that is as important and central as the CMAA demands a control and a policy that is independent of technological, economic, social, political, and cultural players. Therefore, we propose to entrust an established and well-accepted Internet organization, such as the Internet Society (ISOC) or the Internet Corporation for Assigned Names and Numbers (ICANN), with the ruling of CMAA issues. In the following, we denote the trustworthy organization as "central organization" (CO).

It is the task of the CO to specify precise requirements for a CMAA, receive submissions, inspect the applications, officially certificate applying organizations as CMAAs, and, if necessary, withdraw CMAA certificates. It is also desirable that the CO provides standardized software for CMAAs and their customer organizations. More information about the intended software is given in Subsections 3.2 and 3.4.

In principle, designated CMAA organizations may be set up. However, it seems reasonable to assume that, at least in the beginning, mainly already established network organizations, such as trustworthy ESPs, anti-spam organizations, and universities, will serve as CMAAs, because they already dispose of the technological experience, tools, and staff, all of which is helpful, if not crucial, to running a CMAA. The motivation to gain certification and serve as a CMAA can result from two goals: (1) If the CMAA services offered have to be paid for by the organizations that make use of them, then there may be an economic incentive. Furthermore, it saves the costs of registering for an external CMAA. (2) It may increase the organization's reputation.

3.2 Certificating an organization as a CMAA

The effectiveness of the framework regarding the reduction of spam e-mails heavily relies on the trustworthiness of the CMAA organizations. Therefore, the requirements on organizations that apply for certification as a CMAA should be stringent. We propose that the CO considers the following evaluation criteria for CMAA applicants:

- An applying organization should have either a good reputation in the Internet community or at least references from such organizations. The reputation could include a high integrity in network-based services, an active involvement in anti-spam activities, and a good reputation with regard to anti-spam blacklists maintained by well-accepted organizations.
- The applicant should be under legislation that allows for prosecution in the case of any tolerating or supporting of spam activities. Any spam-promotive behavior, be it intentional or negligent, must be triable. Additionally, an applicant may be obliged to pay a deposit, that is forfeited in the case of a strong violation of the requirements on a CMAA. These requirements and any case of strong violation would have to be precisely specified in the contract signed by the CO and a particular CMAA.
- The organization's data in the "whois" database must have been successfully validated.
- The implementation of technological requirements that are mandatory for the operation of a CMAA must be accomplished. These include
  - the protection of services and databases against security vulnerabilities,

  - a system redundancy in order to guarantee the operational availability of CMAA services in the case of system crashes and heavy traffic, and
  - an appropriate load balancing system for a time-efficient use of the redundant systems in order to guarantee an appropriate throughput.

  The last issue addresses a performance requirement which is necessary to keep the Internet e-mail service a "real-time" system. We propose that the CO supports applicants with standardized and certified software for the operation of tasks that each CMAA has to perform. The usage of such software could even be regulated by the CO.

The certification process is intended to involve personal contacts between the applying organization and the CO, and the agreement is formally defined by a contract.

The list of certified organizations, their contact information, the CMAA policy that has to be signed by each certified organization, and organization-specific information, such as service fees, should be provided by the CO. Complaints about a violation against the CMAA policy should be directed to the CO, which can withdraw CMAA certificates if this is deemed necessary.

3.3 Mapping organizations onto CMAAs

It is the decision of each organization that sends e-mails on behalf of its users whether it should use the services of one or more CMAAs, so that we have an $(m : n)$ relationship between SOs and CMAAs. Usually, an SO would use only one CMAA. However, there is no limitation to one CMAA intended, as an SO may choose to use more than one for the reason of increased reliability of its own e-mailing service. The framework is scalable in that it allows SOs to bypass any CMAA and to omit the registration on any CMAA. The pressure on these organizations to register is determined by the extent to which the Internet e-mail community accepts the importance of CMAAs, i.e. to which extent the community of ROs makes the decision of whether an e-mail is accepted or rejected dependent on the participation of a CMAA (or trustworthy SO). If the CMAAs' role is widely adopted by the Internet e-mail community, an SO's omission of a registration at a CMAA results in the rejection of messages sent to users of many or even most organizations.

If an organization has decided to register for CMAA services, then, the question arises as to which CMAA to choose. The mapping of organizations onto CMAAs can follow the market or the regulation paradigm:

Market paradigm One option would be to leave the decision to the particular SO. Then, a market emerges with CMAAs as sellers and SOs as buyers. However, in order to support the wide diffusion and adoption of the CMAAs' integration, the CO should regulate some issues that may otherwise hamper the diffusion of the usage of CMAAs, such as an unlimited fee range.

Regulation paradigm Another option would be to regulate the mapping and to assign a CMAA to an SO. Examples of regulatory approaches can be found at ICANN, which is responsible for IP address space allocation, protocol identifier assignment, TLD name system management, and root server system management functions.

3.4 Registering for the usage of CMAA services

One of the most critical requirements of the proposed infrastructure is the integrity of registered organizations. Although it seems impossible to exclude all those organizations that tolerate or even support spamming in advance, a set of requirements that applicants have to fulfill may be helpful for the (increasing) reduction of fraudulent or careless organizations. The fulfillment of these requirements has to be controlled by the CMAAs. Similar requirements can be found in (ICANN 2004).

- The organization's data in the "whois" database must have been successfully validated. This includes that the administrative contact has signed the application form and proved his/her identity by attaching a copy of a valid identity card. In the case of a repeated misuse of CMAA services and of the toleration or even support of spammers, this contact can be prosecuted.
- Each organization being registered has to sign the anti-spam policy to which it must adhere. In the case of a violation, the organization or its administrative contact can be prosecuted.
- The following technological requirements apply:
  - The administration client (see Figure 3) has to be installed. Like the administration server, this software should be provided by the CO.
  - A public key pair must be generated, and the public key must be added to the DNS. The private key has to be stored securely, i.e. it either has to be stored encryptedly on a server or, even better, on a secure external device, such as a smart card.
  - For the purpose of authentication and authorization (when sending an e-mail to a CMAA), LMAP records have to be added to the DNS.
  - The component — be it a software or a hardware unit — that signs messages on behalf of the organization must be protected against attacks and any misuse. The CO should provide such software and specify the requirements on the hardware to be used.
  - It has to be ensured that a reverse DNS query, with any name server of the applying organization as an argument, results in a FQDN whose "SLD.TLD" part is the name under which the organization is registered at its CMAA (see Equation 1 on page 17).

– One of the most important requirements on applying organizations is the demand for a manual set-up of accounts. The automatic set-up must be prohibited because, otherwise, the limitation of the number of e-mails per account and day would be pointless. One option would be to initiate an offline registration procedure, which demands a letter-based application, that includes both user identification by signature and the provision of a valid mail address. Another option would be to implement a CAPTCHA procedure (von Ahn et al. 2003, von Ahn et al. 2004). However, CAPTCHA procedures suffer from several drawbacks. We propose that the underlying algorithm has been evaluated by the CO and that the CO provides CAPTCHA software, which can be used.
– In order to protect e-mail accounts from easy misuse, an authentication mechanism, for example a password-based one, has to be applied. If SMTP-based connection is used, then SMTP-AUTH (Myers 1999) can be used. Web-based e-mailing services are usually implemented with password-based protection.

In contrast to the CMAA certification process, the registration process is not intended to involve a personal contact. The reason for this is that it would be too cumbersome, as the number of registering organizations is much higher than the number of CMAA applicants.

4 Technological solution

This section describes the technological specification of the framework. This specification consists of the description of the three central data stores, the CDB, the ADB, and the ODB, and of the processes that are related to database administration, to e-mail relaying and bouncing, and to the usage of the abuse system. Regarding the following process descriptions, it is not relevant whether the SO is identical with the CMAA or not. In the former case, the roles "SO" and "CMAA" are both realized by the same organization, and although some process simplifications may then be possible, in principle, the processes are even then intended to run as described.

Further, it should be noted here that all of the technologies required to implement this proposal currently exist. The framework leverages existing technologies and services to reduce spam.

The overall infrastructure framework and its key components are illustrated in Figure 3.

4.1 Databases

Most services offered by a CMAA need to access its CDB. For example, the decision of whether an e-mail is relayed or bounced relies on the data of the

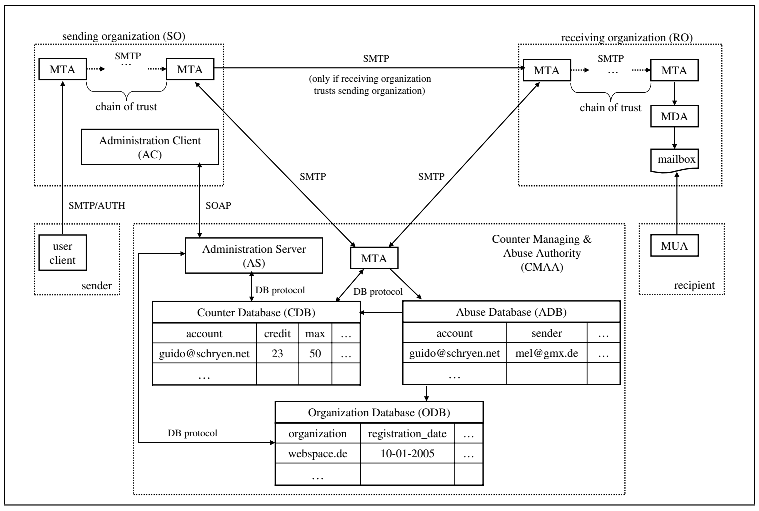


Figure 3: Infrastructure framework

particular CDB. We propose any CDB to maintain for every single CMAA-registered e-mail account the following data:

- *account* is the e-mail address of the database record. E-mails can be sent on its behalf. An example would be any e-mail address, e.g. *guido@schryen.net*
- *credit* contains the current number of e-mails that can be sent on the current day. Its initial value is set to *max*.
- *max* is the number of e-mails that can be sent per day on behalf of the particular account.
- *bounce_status* indicates whether a bounce e-mail has already been sent to the *account*. This would happen when the e-mail limit is first exceeded. Then, *bounce_status* would have to be changed to indicate that no further bounce e-mail is necessary.
- *setup_org* contains the SLD and the TLD, for example *freenet.de*, of the organization that set up the record and which offers the e-mail account to the particular user. Only the organization that set up a record is authorized to relay e-mails on behalf of the e-mail address stored in that particular record.
- *setup_date* gives the date of the set-up procedure and allows for statistical evaluations.
- *holder* provides the name and the mail address of the holder of the account. This information is mandatory, if the credit of the account exceeds the default credit, thus offering the option of sending solicited bulk e-mail on

behalf of that particular account. If this account is misused for the sending of unsolicited bulk e-mail, then the holder information may be used for prosecution.

- *idle_days* is the number of days the account has not been used. When certain thresholds are exceeded, the responsible organization — stored in *setup_org* — is informed about the possibly upcoming removal of the account and, finally, about its removal.

- *blocks* gives the number of times the account has been blocked so far.

- *status* allows the provision of information about the status of the account. Possible values are "open" and "blocked". The status "blocked" may be reached, when a specific number of complaints have been received.

Regarding the misuse of the abuse system, we propose that each complainant can only submit one abuse complaint about an account per day. The ADB would contain the following data:

- *account* is the e-mail address of the database record.

- *setup_org* contains the same type of information as the corresponding entry in the CDB. This redundancy serves the purpose of efficiency, when organization-related abuse information is being composed.

- *sender* provides the e-mail address of the complainant. This information is necessary to ensure compliance with the restriction mentioned above.

- *date* gives the date of the abuse complaint.

The ODB contains information about the organizations that have successfully registered for the usage of the CMAA services. We propose storing the following information:

- *organization* contains the same type of information as the corresponding entry in the CDB.

- *registration_date* gives the date of the registration process.

- *complaints1*, ..., *complaints30* provide the number of abuse complaints for the last 30 days, whereby 30 is an arbitrary number.

- *admonishments1*, ... *admonishments6* allows the storage of the number of admonishments for the last six months. Again, six is an arbitrary number.

- *status* provides information about whether the organization has been excluded or whether it may still use the CMAA services.

It should be noted that the protection of e-mail addresses that are stored in the databases is very important, because the databases would otherwise provide valuable resources for spammers. The proposed extension of the infrastructure would then be even counterproductive. Although the usage of hash values or encrypted addresses would seem to be solutions to this problem, they suffer from the following drawbacks: If only hash values of e-mail addresses are stored, then the e-mail addresses cannot be recovered efficiently. However, the e-mail addresses are needed for some CMAA messages, such as an *REMOVAL_ABUSE* message. If the addresses are stored encryptedly, they can be recovered by applying the decrypt function. However, most CDB administration processes and the e-mail delivery process include the sending of an e-mail address that would have to be encrypted or decrypted. Because of the high number of expected queries, the use of cryptographic functions would probably consume too much time. Therefore, the usage of other mechanisms, such as authorization-based ones, should be explored.

This discussion reflects the challenge to many infrastructures and systems in finding an appropriate balance between security, functionality, and (time-related) efficiency.

4.2 Database administration processes

Access to the CDB is granted to SOs that have been approved for the usage of the CMAA-specific CDB and to the CMAA itself. SOs are allowed to set up, modify, and remove records, herein denoted as processes P1, P2, and P3. The CMAA is responsible for the periodical maintenance of the CDB records in many regards. It has

- to reset values of each record, for example, the credit, by a fixed time of the day (P4),
- to trace accounts that have not been used for a specific time in order to remove those particular accounts or to inform the responsible SO about the possible upcoming removal (P5), and
- to block accounts due to spam complaints (P6).

The administration of the ADB and of the ODB is reserved for the CMAA. It is responsible for both the detection of accounts, for which many abuse complaints have arrived, and the detection of SOs that are responsible for such "suspicious" accounts. As a consequence, accounts have to be blocked and SOs have to be admonished or even excluded from all CMAA services (P7). All complaint and admonishment information stored in the ODB has to be updated periodically, because complaints only refer to the last 30 days and admonishments only to the last six months (P8).

As the data that are exchanged between an SO and an CMAA are completely structured, the usage of e-mails seems to be improper. Rather, the Simple Object Access Protocol (SOAP), which is an XML based W3C standard for a platform-independent communication between applications (W3C 2003), provides means for this communication. Figure 8 (see Appendix A) shows an XML schema for messages that are directed from an CMAA to an SO. The attribute *message_type* specifies the type of message. The mandatory element *object* either contains the name of an account or the name of an organization. A remark may be (optionally) added. Any CMAA message is intended to be cryptographically signed by using the "SOAP Security Extensions", which are described later in this subsection. The message types provided are presented in Table 1. As XML schemas for messages that are directed to an CMAA are process-specific, they are presented in the descriptions of the respective processes.

Table 1: Types of CMAA messages

message_type	**semantics**	**affected process**	**object**	**remark**
SETUP_SUCCESS	setup of user account succeeded	P1	account	–
SETUP_FAILURE	setup of user account failed	P1	account	reason for failure
UPDATE_SUCCESS	modification of user account succeeded	P2	account	–
UPDATE_FAILURE	modification of user account failed	P2	account	reason for failure
REMOVAL_SUCCESS	removal of user account succeeded	P3	account	–
REMOVAL_FAILURE	removal of user account failed	P3	account	reason for failure
REMOVAL_IDLE	removal of user account due to idle time	P5	account	–
UPCOMING_ REMOVAL_IDLE	upcoming removal of user account due to idle time	P5	account	removal date
BLOCKING_ABUSE	blocking user account for one day due to abuse complaints	P6	account	number of remaining blocks
REMOVAL_ABUSE	remove user account due to the exceeding of blocking limit	P6	account	–
ADMONISHMENT	admonish organization due to medium-level abuse of accounts	P7	organization (sld.tld)	number of remaining admonish-ments
EXCLUSION	disclose organization due to high-level abuse of accounts	P7	organization (sld.tld)	–

4.2.1 Process P1: setting up a CMAA record

P1 is illustrated in Figure 4, which models the process with an UML 2.0 activity diagram.

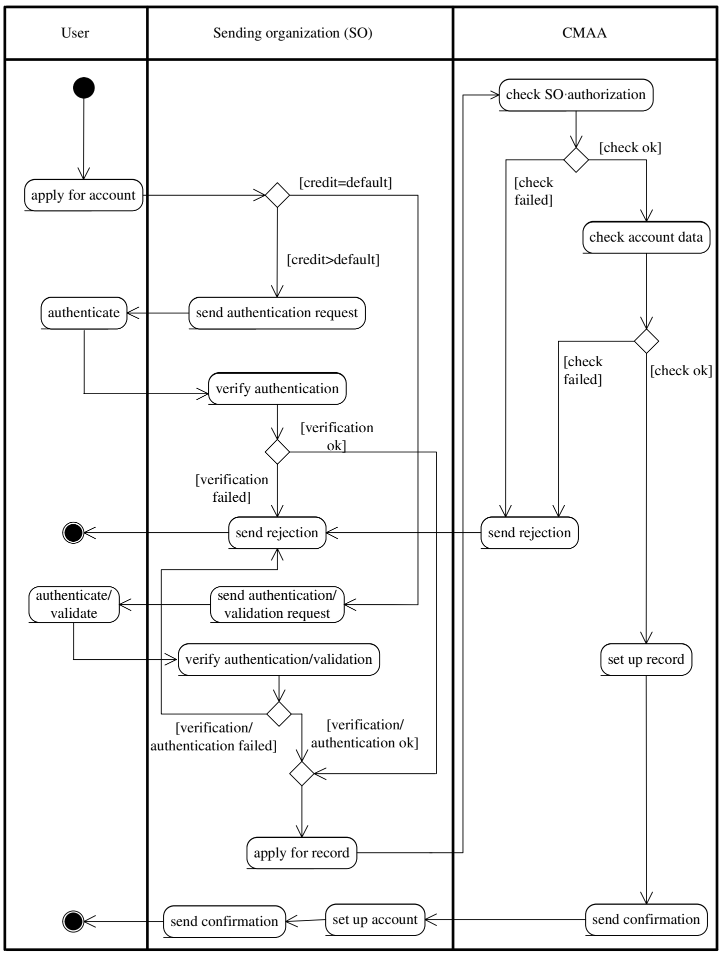


Figure 4: Activity diagram modeling the set-up of a CDB record

The process, by which a CMAA record is set up, is initiated by a user when he or she wants to set up an e-mail account at an SO, for example an ESP. The user usually applies online by using a web form, and he or she is intended to have two options regarding credit: if the user needs more than the default credit, for example 1000 e-mails per day instead of the default value 50, then he or she has to authenticate. This authentication is intended to be submitted offline by mail or fax and must disclose the user's identity and address. For a possible prosecution due to spamming, we propose ensuring that the user underlies an opt-in legislation. If the user applies for an account with default credit, then, either the same authentication procedure applies or an effective CAPTCHA procedure has to validate that a human user is applying. If the authentication/validation succeeds, then the SO applies for a CDB record at its CMAA. The CMAA first checks the authenticity. We propose appling a (cryptographic) signature-based procedure for this, because this approach makes it rather difficult, if not practically impossible, to spoof sender data, which would easily lead to the setting up of an arbitrary number of accounts. The SOs' public keys could be stored in the DNS. If the authentication fails for any reason, a rejection message is sent to the SO. Otherwise, the CMAA has to proceed with the authorization of the SO to set up a record for the particular e-mail account. The SO is granted this permission if it is responsible for the e-mail account. This responsibility is defined as follows: either the *SLD.TLD* domain of the e-mail address is a domain of the SO, for example *schryen@winfor.rwth-aachen.de*, where *rwth-aachen.de* is a domain of RWTH Aachen University, or the domain is hosted by the SO, for example, the domain of the e-mail address *guido@schryen.de*, *schryen.de* is hosted by the SO. In both cases, each authoritative name server for the given domain belongs to the SO. This verification can be undertaken by using the DNS: let *DNSNS(domain)* be the operation that requests the DNS for a name server of *domain*, let *RDNS(IP)* be the operation that requests the DNS for the host that matches *IP*, let *SLDTLD(address)* be the operation that returns the *SLD.TLD* part of a host or an e-mail address, let *setup_org* be the SLD.TLD part of the organization that requests the record set-up, and let *address* be the e-mail address for which a record is requested. Then, the requirement can be verified by using two, possibly cascading, accesses to the DNS:

$$SLDTLD(RDNS(DNSNS(SLDTLD(address)))) = setup\_org^4 \qquad (1)$$

If the verification of responsibility succeeds, the CMAA sets up the record, as requested, by adding default values for the remaining fields (see Table 2) and sends a confirmation (message type=*SETUP_SUCCESS*) to the SO, which then sets up the particular e-mail account and sends a confirmation message to the user. If the verification fails, the CMAA sends a rejection to the SO (message type=*SETUP_FAILURE*), which then sends a rejection message to the user.

Table 2: Initialization data for a CMAA record

Field	**Value**
account	<element “account“ of SOAP message>
credit	<element “max“ of SOAP message>
max	<element “max“ of SOAP message>
bounce_status	unbounced
setup_org	<element “setup_org“ of SOAP message>
setup_date	<current date>
holder	<element “holder“ of SOAP message>
idle_days	0
status	open

[4]Note that for a successful authorization, each requesting organization is responsible for the provision of adequate DNS entries (see Subsection 3.4).

Figure 9 (see Appendix A) displays a SOAP-based message for the set-up of a record, whereas Figure 10 (see Appendix A) shows the underlying XML schema. According to the XML schema, the element “holder” is optional. However, the CMAA SOAP server application has to consider that holder data must be provided if the value of *max* is higher than the default value, which still has to be defined. The SOAP message does not only rely on the XML schema, depicted in Figure 10, but also on a schema that provides “SOAP Security Extensions” (W3C 2001). In this extension, the SOAP body is signed and the signature is added to the header. In our example, the body is hashed by using the algorithm SHA-1. The hash value is then signed by using the algorithm DSA.

A SOAP-based rejection/confirmation message of the CMAA is proposed in Figure 11 (see Appendix 11). For the purpose of lucidity, the digital signature (of the CMAA) is omitted. This SOAP message can also be used for answering modification and removal requests (processes P2 and P3).

4.2.2 Process P2: modifying a CMAA record

SO is allowed to modify the *max* value and/or the *holder* value only. If the *max* value is reset to the default value, no holder data must be given. Otherwise the provision of holder data is mandatory. When an SO sends a modification request to the CMAA, the CMAA proceeds analog to its operations in P1 (see Figure 4). The SO’s SOAP message is similar to that in Figure 9; however, the field *setup_org* can be omitted. The CMAA’s answer message is either of the type *UPDATE_SUCCESS* or *UPDATE_FAILURE*.

4.2.3 Process P3: removing a CMAA record

The deletion of a CMAA record only requires that the SO provide the account name. Regarding the SO’s SOAP message, the notes on P2 apply. The CMAA’s answer message is either of the type *REMOVAL_SUCCESS* or *REMOVAL_FAILURE*.

4.2.4 Processes P4 and P5: resetting the credits of CMAA records and tracing idle CMAA accounts

By a fixed time of the day, the CMAA would have to reset the values of each record. The tracing idle CMAA accounts can be shared with this procedure, as shown in Figure 12 (see Appendix B). The CMAA would use messages of the type *REMOVAL_IDLE* and *UPCOMING_REMOVAL_IDLE* to inform an SO about the (upcoming) removal of a user record. Although a UML activity diagram can be used for this description, the usage of a Nassi-Shneiderman diagram makes the description clearer.

4.2.5 Process P6 and P7: blocking CMAA accounts or/and SOs

The CMAA should daily consolidate abuse complaints. This consolidation may lead to the blocking or removal of user accounts. Furthermore, if too many complaints refer to different accounts of one specific SO, then, the SO has to be admonished or even excluded from all CMAA services. Figure 13 (see Appendix B) shows the CMAA's activities that are related to the management of abuse complaints.

The following issues of processes P6 and P7 are worth a mention:

- When the number of abuse complaints on a specific account exceeds the daily limit (*LIMIT_BLOCKING*), the account is blocked for one day (message_type= *BLOCKING_ABUSE*). Each account may be blocked a number of times (*LIMIT_REMOVAL*), which are still to be specified. If the total number of blocking exceeds this value, then the account is removed and the responsible SO is informed about this deletion (message_type=*REMOVAL_ABUSE*).
- It may happen that SOs ignore, tolerate or even support the abuse of e-mail accounts. Therefore, for each organization, all complaints about those accounts that the organization is responsible for are counted and stored in the ODB — this procedure is denoted as *update* in Figure 13 —, which contains for each SO the number of abuse complaints for each of the last six months. We differentiate between three abuse states that an organization can be assigned: low, medium, and strong. The status results from the application of the CMAA's policy on the SO's six-month complaint history and the SO's number of past admonishments. The following actions have to be taken by the CMAA, depending on the SO's status:
  - If the history is assessed as "low", nothing has to be done.
  - If the value is "medium", then the CMAA sends an admonishment to the SO (message_type=*ADMONISHMENT*) and records this.
  - In the case of a "strong" violation, that particular SO would have to be excluded from all CMAA services. The status would be set to "excluded", all accounts that had been set up by the SO would be removed, and the SO and all other CMAAs would be informed about this exclusion (message_type=*EXCLUSION*).

4.2.6 Process P8: removing complaint and admonishment information

Complaints older than 30 days and admonishments older than 6 months are intended to be removed from the ODB. The removal of complaints has to be executed once a day; the deletion of admonishment information once a month.

4.3 E-mail delivery process

The process of sending an e-mail has to be extended by the integration of a CMAA. Although a CMAA's involvement makes the delivery process more complicated, the modifications are intended to be hidden from the user, who may continue using his/her e-mail client software without any changes. Figure 5 shows the process by using an activity diagram. Figure 3 (see page 12) provides an infrastructure view of this process.

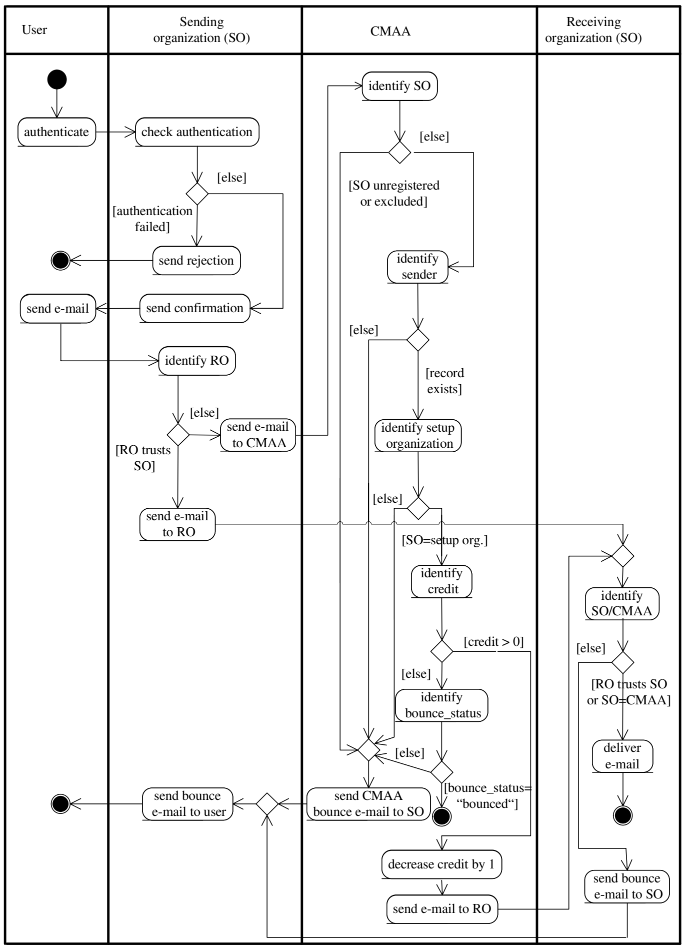


Figure 5: Activity diagram modeling the e-mailing process

The process can be divided into the following components:

1. User authentication

   First, the user has to authenticate, so that his/her account is protected from misuse by an unauthorized person. We propose using the IETF standard SMTP-AUTH (Myers 1999) with a (user,password) SASL authentication mechanism (Myers 1997). However, for effective protection from misuse, the password must be strong, i.e. not guessable and not too short, and protected from being read by malicious software. If the authentication fails, the process is terminated, otherwise the user can send the e-mail to his/her SO.

2. SO's relaying decision

   For each recipient of the e-mail, the SO looks for the RO in the internal database that stores the names of those organizations that accept direct e-mail communication with the own organization. If the RO is listed, then the e-mail is sent directly to the RO, otherwise it is sent to the SO's CMAA. The case where the SO is identical to the RO is covered implicitly. In such a case, the involvement of an CMAA is not intended. However, it should still be an option for a SO to let a CMAA count those e-mails that are not directed to another SO, in order to protect their users' accounts from being spammed. For the sake of simplicity, this option is omitted in Figure 5.

3. CMAA's relaying decision

   The CMAA checks whether the SO is registered and not excluded — the SO data can be obtained from the e-mail's FQDN, which has to be successfully validated against the IP of the sending host by using an LMAP-based procedure (DeKok 2004, Leibzon 2005)[5] —, if the CMAA maintains a record for the sender and if the SO is allowed to send e-mails on behalf of the sender account. If one of these conditions is not fulfilled, the CMAA refuses the relaying and sends a bounce e-mail to the SO, which, then sends a bounce e-mail to the sender. If all tests succeed, the CMAA checks the sender's credit. If no credit is available, the relaying is refused and, provided that no bounce e-mail due to the unavailable credit has been sent, a bounce e-mail is sent to the SO. It is important to send, at most, one bounce e-mail per day and account due to credit unavailability, because it would be otherwise possible to maliciously initiate the sending of an arbitrary number of bounce e-mails to a compromised account: once a password is read or guessed, an attacker could easily send e-mails on behalf of this account thereby causing the CMAA to send a bounce e-mail for each e-mail that exceeds the account's e-mail limit. If the credit is larger than 0, then the credit is decreased by 1 and the e-mail is relayed to the SO.

4. RO's acceptance decision

   When an organization receives an e-mail, it first operates an LMAP-based validation as described above. If the validation fails, the process terminates. Otherwise, the RO checks whether the SO is whitelisted regarding a direct e-mail communication or if the delivering host belongs to a CMAA. If this check is successful, the e-mail is accepted and delivered to the e-mail's recipient. Otherwise, the e-mail acceptance is refused and a bounce e-mail is sent to the SO.

[5]If the validation fails, the process terminates. In order to keep the activity diagram in Figure 5 as simple as possible, this issue is not modeled in it.

If a CMAA participates in e-mail delivery, its MTAs add *Received* entries to the header as described in RFC 2821. No further modification is necessary.

4.4 Abuse complaint process

The success of the abuse system depends on the user participation in the sending of abuse e-mails to the CMAAs. In order to make a user send a spam complaint to a CMAA, he/she has to know to which CMAA the complaint has to be directed. We envisage at least two options for providing this information: either the CMAA that relays a message adds a new header entry to the e-mail, for example: *X-Compliant: <abuse-e-mail-address>*, or adds this information to the e-mail's body as part of a CMAA signature. The first option would be preferable for keeping an e-mail text free from any CMAA (meta) information and for easing the implementation. The reason is that the header entry could be added at the beginning of the message without seeking the appropriate position in the body, which could contain several MIME parts thereby complicating the e-mail's structure. The second option allows the recipient to easily identify the abuse address without having to make the header entries visible. Furthermore, many users are likely to know little or nothing about the (existence of an e-mail) header.

When a user wants to complain about a received e-mail, then the user would have to send an abuse e-mail to the responsible CMAA via his or her organization. The CMAA that receives the complaint e-mail would have to perform three checks: (1) Is the e-mail a complaint message? (2) Does the CMAA maintain a record for the account being complained about? (3) Does the ADB already contain a complaint tuple (account,sender,date)? The purpose of the third check is to prevent the abuse system from being misused by users sending multiple complaints about the same account in order to discredit it. Only if all checks are positive, is a new complaint record added to the ADB. The *setup_org* data can be obtained by requesting the CDB. As this process is very simple, a graphical representation is omitted here.

The content and format of a complaint e-mail is not specified here, in order to avoid an overstandardization; however, an abuse e-mail must contain the account and the date of the e-mail being complained about. Furthermore, each complaint message has to labeled as such a message because the CMAAs have to handle it different from a "regular" message. The content and format may vary between different CMAAs, although for the purpose of consistency, it is useful to standardize these issues.

5 Theoretical effectiveness

An empirical evaluation of the proposed infrastructure is not possible, unless it has been already implemented. However, using an appropriate model of the (proposed) Internet e-mail infrastructure, we are able to investigate which delivery routes that a spam e-mail may take are covered. This investigation is the assessment of what we denote as the theoretical effectiveness of the infrastructure. Schryen (2006b) proposed a graph G that models the current e-mail infrastructure and that is displayed in Figure 6. The nodes of G represent types of e-mail nodes that operate on TCP/IP application layer, such as MTAs and gateways. Directed edges correspond to e-mail connections between two (types of) e-mail nodes, with the edges' direction indicating the orientation "client to server". The edges are assigned a specific value, which is a set of labels representing those protocols which are feasible for the particular edge or connection respectively. Therefore, G can be denoted as a directed, labeled graph.

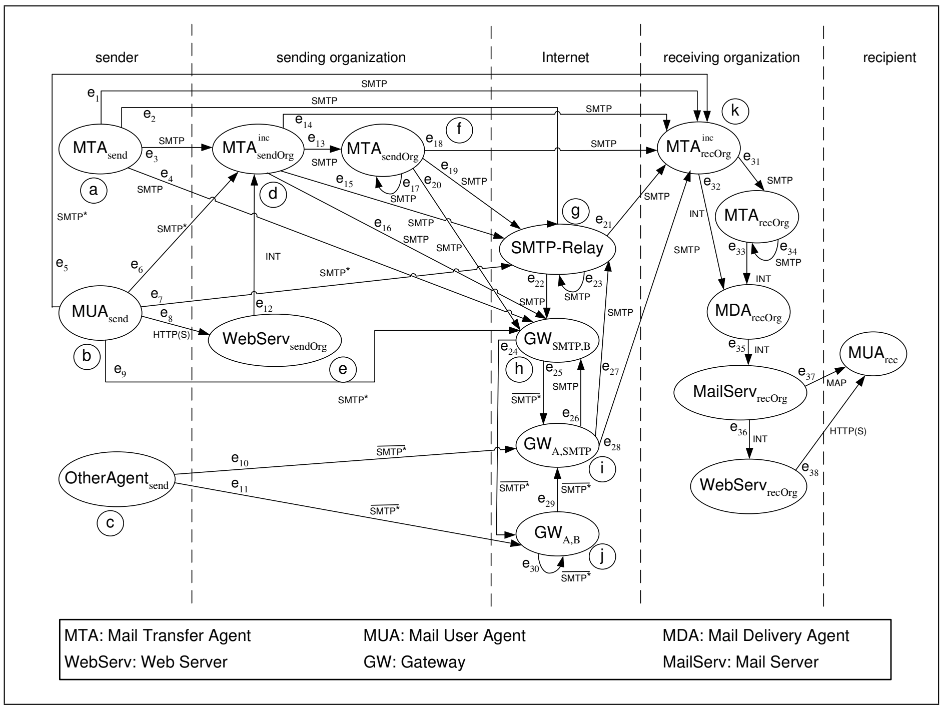


Figure 6: Internet e-mail network infrastructure as a directed graph (Schryen 2006b)

Our framework does not add a new element to the modeled infrastructure: From a technological point of view, the CMAA simply represents an SMTP relay. Therefore, the graph model also represents the proposed e-mail infrastructure.

According to the construction of G, e-mail delivery routes are represented by paths in G. As it is essential for today's e-mail delivery process that the way in which an e-mail node received an e-mail does not restrict the way it passes the e-mail forward, each path p corresponds to a feasible e-mail delivery route. We are only interested in complete e-mail deliveries, which means that the e-mail has reached the recipient's e-mail box on his or her e-mail server or the MTA of the RO, which applies a forwarding rule or rejects the message. That is, forwarding an e-mail and sending a bounce e-mail starts a new sequence. Furthermore, only those e-mail deliveries are regarded which are either initiated by a sender's client or, for example, in case of bouncing or forwarding e-mails, by an ESP's MTA. More precisely, we are interested in all paths $p = (v_{\text{start}}, \ldots, \text{MailServ}_{\text{recOrg}})$ with $v_{\text{start}} \in \{\text{MTA}_{\text{send}}, \text{MUA}_{\text{send}}, \text{OtherAgent}_{\text{send}}, \text{MTA}^{inc}_{\text{sendOrg}}, \text{MTA}_{\text{sendOrg}}, \text{WebServ}_{\text{sendOrg}}\}$, which represent (spam) e-mail delivery routes.

Schryen (2006b) applies some basic ideas from automata theory to derive the set of paths. The paths are represented by regular expressions. For simplicity, the nodes of G are labeled with letters, which are assigned to the states of the corresponding Deterministic Finite Automaton (DFA), which is used to derive the regular expressions.

Table 3 shows the paths/routes and explains to which extent our framework addresses them. The framework addresses all scenarios completely by adhering to three principles:

1. Some delivery routes, that are difficult to control, are classified as insecure. These routes comprise those that do not include a CMAA relay (scenario III and large parts of scenarios II , IV, and VI). ROs are recommended for refusing e-mails that have taken such delivery routes.

2. Each RO can maintain a whitelist and decide which organizations it trusts. If one of these organizations sends or relays spam e-mails, each RO can exclude that organization from any direct e-mail communication at any time (scenarios I and V).

3. In most cases, an SO would send an e-mail to an RO that does not include that SO on its whitelist. Then, the only option for a successful delivery is to use a CMAA's relaying service. However, in order to be allowed to use such a service, the SO must have successfully registered for the CMAA services. This registration requires the implementation of organizational and technological anti-spam measures, which, for example, avoid the automatic setup of e-mail accounts and prevent the hampering of existing e-mail accounts. The relaying scenarios cover the complementary parts of the scenarios II, IV, and VI.

Today's most challenging issues of technological anti-spam activities include third party exploits and the fact that it is all too easy for spammers to set up e-mail accounts automatically. Both problems are addressed in our framework with a combination of prevention and limitation of the possible harm: the framework provides means to ensure that (1) only a manual set-up of accounts is allowed, (2) the number of (spam) e-mails per account and day is restricted, (3) users are informed about the misuse of their accounts so that they can take countermeasures.

Table 3: Effectiveness of the proposed framework (Schryen 2006b)

Scenario	Regular expression	Nodes involved	Effect of the proposed framework
I	d (k ∨ ff*k) ∨ ff*k	MTAs of provider	❑ Organizations can control e-mails via local white lists, that contain trustworthy sending organizations.
II	(d ∨ Λ)(ff*gg*k) ∨ dgg*k	MTA of provider, then relay(s)	❑ Organizations accept incoming SMTP connections only from relaying hosts that belong to a (trustworthy) CMAA. ❑ The relaying service is only offered to (registered) organizations that have implemented both technological and organizational measures against the misuse of e-mail accounts. ❑ Misused accounts can be identified and blocked by a CMAA.
	(d ∨ Λ) (ff*gg*hH) ∨ dgg*hH	MTA of provider, then relay(s) and gateway(s)	
	(d ∨ Λ) (ff*hH) ∨ dhH	MTA of provider, then at least gateway(s)	
III	ak	Local MTA	❑ These e-mails are sent from a host which neither belongs to a CMAA nor to a whitelisted organization. ❑ These e-mails **do not have to be accepted** by a receiving organization.
IV	(a ∨ b) gg*k	Local MTA or MUA, then relay(s)	❑ These e-mails come from hosts that are not whitelisted and do not belong to a CMAA (no CMAA would accept e-mails from hosts of unregistered parties). ❑ These e-mails **do not have to be accepted** by a receiving organization.
	(a ∨ b) (gg*hH ∨ hH)	Local MTA or MUA, then relay(s) and gateway(s)	
	cj*i (hH ∨ gg*H ∨ gg*k ∨ k)	Local agent other than MTA or MUA, then at least gateway(s)	
V	(ad ∨ bd ∨ bed) (k ∨ ff*k)	Local MTA or MUA, then MTA(s) of provider	(see remarks on scenario I)
VI	(ad ∨ bd ∨ bed) (ff*gg*k ∨ gg*k)	Local MTA or MUA, then MTA(s) of provider, then relay(s)	(see remarks on scenario II)
	(ad ∨ bd ∨ bed) (ff*hH ∨ hH)	Local MTA or MUA, then MTA(s) of provider, then at least gateway(s)	
	(ad ∨ bd ∨ bed) (ff*gg*hH ∨ gg*hH)	Local MTA or MUA,then MTA(s) of provider, then relay(s) and gateway(s)	

6 Resource analysis

The proposed framework involves the need for resources. This need comprises the demand for storage for CDB, ADB, and ODB databases, and for bandwidth due to the increased network traffic, which results from the relaying of e-mail messages, from the sending of abuse e-mails, and from the traffic caused by the administration processes. This section performs a quantitative analysis of these factors and allows for assessing under which assumptions (of a reduced number of spam e-mails) the total demand on storage and network traffic decreases.

6.1 Storage analysis

The framework includes the maintenance of several data pools, which require additional storage. We determine the storage requirements and compare them with the storage savings that result from the intended reduction of spam e-mails. We determine the storage requirements for CMAA databases only, because all other data pools, such as those storing white lists, can be neglected with regard to these databases. However, we only focus on the CDB and the ODB, but not on the ADB for two reasons:

1. No reliable data about the expected numbers of abuse complaints is available.

2. The storage of abuse complaints is not a designated feature of our framework. Abuse complaints are already stored today, although they are stored decentrally at different organizations, such as ESPs and anti-spam organizations. Therefore, the storage of abuse complaints has to be assessed independently from the assessment of our framework.

Table 4 shows how the storage requirement for a single entry is determined. The maximum space required for an entry is 433 Byte and 2 Bit. However, it should be noted that most entries need much less space for two reasons: (1) account names are usually shorter than 320 characters; we suppose that the average number of characters is under 50. (2) Holder data may easily take almost 100 characters, because the full name and the address have to be stored. However, not every entry includes holder data, as they have to be provided only for accounts on whose behalf more than the default number of e-mails can be sent per day. We provide values for different assumptions (50 Byte, 75 Byte, 100 Byte) on the average number of characters required to store the holder. We further need to make assumptions on the total number of entries, i.e. accounts, stored in the CDBs: according to (Internet Systems Consortium 2006), in January 2006, the total number of hosts registered in the DNS was about 395 million. Unfortunately, updated information about the average number of e-mail accounts per host is unavailable. However, in 2001, Telcordia Technologies Inc. found that the ratio of users to hosts in the United States is 2.4 to 1, whereas some countries, including China and India, have as many as 100 Internet users per host (Telcordia Technologies, Inc. 2001). We use this range for assumptions on the average number of e-mail accounts per host and perform computations for different values (10, 20, 50, and 100). Table 5 shows the required storage for different assumptions. The storage requirements are calculated by multiplying the total storage per CDB record — 2 Bit are rounded up to 1 Byte — with the number of hosts (395 million) and with the average number of accounts per hosts. This calculation is based on the assumption that, for each e-mail account, one CDB entry is stored.

Table 4: Storage requirement for a CDB entry

Data field	**max. size**	**Remark**
account	320 Byte	64 Byte for local part [RFC 2821], 255 Byte for domain [RFC 2821], 1 Byte for @
credit	2 Byte	
max	2 Byte	no account is allowed to send more than per day
bounce_status	1 Bit	
setup_org	4 Byte	only a pointer to an ODB entry is stored
setup_date	3 Byte	
holder	100 Byte	arbitrary maximum length of name and address 100 characters
idle_days	1 Byte	idle time must not exceed 255 days
blocks	1 Byte	number of blocks must not exceed 255
status	1 Bit	
total	433 Byte+2 Bit	

Table 5: Total storage requirement for CDBs

Storage per CDB record		**Storage requirements (data/index)**			
		Average number of accounts per host			
Holder	Total	10	20	50	100
50 Byte	114 Byte	420 GB 200 GB	840 GB 400 GB	2100 GB 1000 GB	4200 GB 2000 GB
75 Byte	139 Byte	511 GB 200 GB	1022 GB 400 GB	2555 GB 1000 GB	5110 GB 2000 GB
100 Byte	164 Byte	603 GB 200 GB	1206 GB 400 GB	3015 GB 1000 GB	6030 GB 2000 GB

In addition to the storage of the CDB entries, we have to take into account the storage of an index for the CDB. For ensuring an appropriate performance, it seems reasonable to store an index for the *account* field, as most transactions need to access the CDB with a given account. If we follow the assumption that the average number of characters of an account is 50, then the average storage requirement of an index entry is 55 Byte (50 Byte for the account and 5 Byte for the index). Finally, we obtain the storage requirements displayed in Figure 5.

Table 6 shows the storage requirements for a single ODB record. If we multiply the total number of hosts registered in the DNS (395 million) by the storage requirement of a single entry (193 Byte), we obtain (after rounding up) about 71 GB as the total ODB storage requirement. If we assume that an index entry for the organization's domain requires 68 Byte (63 Byte for the account and 5 Byte for the index), then the index would need about 25 GB. This calculation is based on the assumption that, for each registered host, one ODB entry is stored.

Table 6: Storage requirement for an ODB entry

Data field	Max. size	Remark
organization	63 Byte	maximum length of domain name in IDNs format [RFC 3490]
registration date	3 Byte	
complaints$_i$, i=1..30	30*4 Byte = 120 Byte	
admonishments$_j$, j=1..6	6*1Byte=6 Byte	number of monthly admonishments must not exceed 255
status	1 Bit	
total	192 Byte+1Bit	

Summing up the storage requirements for all CDBs and ODBs, we obtain (for our worst scenario) 8,126 GB. We now estimate the storage requirements for spam e-mails today and compare the numbers: According to market research companies, the estimated number of spam e-mails per day is about 12.4 billion (Evett 2006). The average size of a spam e-mail is about 25 KB (Bundesamt für Sicherheit in der Informationstechnik (BSI) 2005). If we further assume that a spam e-mail is stored for only one day on ESPs' servers — this assumption seems very optimistic —, we obtain as a total storage requirement about 295,639 GB. We further have to add the storage requirements for bounce e-mails that result from spam e-mails that could not be delivered due to a non-existing recipient mailbox. According to a study by Ironport (2006), about 13.4% of all spam e-mails are bounced, and about 10% of these e-mails have valid addresses and, therefore, are stored in the users' mailboxes. The size of a bounce e-mail is larger than the size of the e-mail that is bounced, because the bounce e-mail usually contains the bounced e-mail. Following our principle of making conservative assumptions and aiming at a simplification of computations, we assume a bounce e-mail to have the same size as the e-mail that is bounced, i.e. 25 KB on average. If we further assume that a bounce e-mail is stored for only one day, we obtain for bounce e-mails a total storage requirement of about 1.34% * 295,639 GB $\approx$ 3,692 GB. In total, we obtain as the storage requirement 299,331 GB or almost 300 TB.

This means that our framework would have to prevent the storage of $8,126/299,331 \approx 2.7\%$ of all spam e-mails sent in order to achieve the "break-even". This percentage would have to be divided by the expected numbers of "storage days" if this number is different from 1. The Bundesamt für Sicherheit in der Informationstechnik (BSI) (2005, p. 31) even assumes that a spam e-mail is deleted after 8 days. It should also be noticed that the calculations of the frameworks's storage requirements is based on some "worst case assumption" so that the "break-even" value 2.7% can be regarded as an upper bound.

6.2 Traffic analysis

The analysis of the traffic that relies on e-mails is more challenging than the analysis of the storage, because we know nothing about the traffic that would rely on the maintenance of CMAA databases. Therefore, we have to make (very conservative) assumptions regarding this traffic.

We do not consider any traffic related to abuse complaints. The reasons for this omission are those mentioned in Subsection 6.1.

Today's traffic comprises the ham e-mail traffic, the spam e-mail traffic, and the bounce e-mail traffic. These traffics are quantified on the basis of the study of Ironport (2006). According to this study, which claims to observe 25% of the world's e-mail traffic, the aggregate global e-mail is made up of only 20 percent legitimate messages (ham e-mails), whereas spam comprises up to 71% and misdirected bounce e-mails 9%. Relying on the study of Evett (2006), we assume that the daily number of spam e-mails is 12.4 billion. Consequently, we assume about 3.5 billion ham e-mails and about 1.6 billion misdirected bounce e-mails. As no reliable data about the average size of a ham e-mail is available, we introduce a variable, $\emptyset\_ham\_size$, for this size. Unlike the determination of data storage, we do not use "Byte" as a unit, but we have to use "Bit". The reason is that, in e-mail communication, different "content transfer encodings" have to be used for representing binary data or 8bit ASCII text, because Internet e-mail messages are originally restricted to the 7bit US-ASCII format. The IANA proposes the following formats: 7bit, 8bit, binary, quoted-printable, and base64 (Freed and Borenstein 1996). Depending on the encoding mechanism, the data volume that has to be transfered is increased. For example, the "base64" encoding leads to an increase of one third. Unfortunately, we do not have any reliable distribution of encodings regarding different types of e-mails. Therefore, we introduce factors $\text{enc}_i \in [1; \frac{4}{3}], i \in \{\text{ham}, \text{spam}, \text{bounce}\}$, that take this increase into account. The traffic is computed as follows:

$$\begin{aligned}
\text{Ham e-mail traffic} &= 3.5 * 10^9 * \emptyset\_ham\_size * \text{enc}_{\text{ham}} \\
\text{Spam e-mail traffic} &= 12.4 * 10^9 * 25\,\text{KB} * \text{enc}_{\text{spam}} \approx 2,294\,\text{Tbit} * \text{enc}_{\text{spam}} \\
\text{Bounce e-mail traffic} &= 1.6 * 10^9 * 25\,\text{KB} * \text{enc}_{\text{bounce}} \approx 296\,\text{Tbit} * \text{enc}_{\text{bounce}}
\end{aligned}$$

Thus, the total traffic is estimated at

$$\begin{aligned}
\text{total traffic} &= 3.5 * 10^9 * \emptyset\_ham\_size * \text{enc}_{\text{ham}} + \\
&\quad 2,294\ \text{Tbit} * \text{enc}_{\text{spam}} + \\
&\quad 296\ \text{Tbit} * \text{enc}_{\text{bounce}}. \qquad (2)
\end{aligned}$$

We now compute the expected traffic that occurs after the introduction of our framework. We do not consider the traffic that is necessary to set up the CMAA accounts, because this traffic occurs only once and is negligible in the long run. The traffic consists of the ham e-mail traffic, the bounce email traffic, the spam e-mail traffic, and the traffic that results from the administration of the CDB and ODB databases.

We introduce some variables that are used for the quantifying of the traffic. Let

- p be the portion of ham e-mail s that is sent in a direct SO-RO connection,
- t be the daily portion of CMAA accounts whose e-mail limits are exceeded, which results in the sending of a bounce e-mail,
- $\emptyset\_bounce\_size$ be the average size of a CMAA bounce e-mail,
- $\#messages$ be the number of daily administration messages, which are either initiated by an SO or by a CMAA, and
- $\emptyset\_message\_size$ be the average size of an administration message.

Then, the traffic parts can be quantified as follows:

$$\begin{aligned}\text{Ham e-mail traffic} &= (p * 3.5 * 10^9 * \emptyset\_ham\_size + \\ &\quad (1-p) * 3.5 * 10^9 * \emptyset\_ham\_size * 2) * \text{enc}_{\text{ham}}\end{aligned}$$

The first summand describes the traffic that results from a direct communication, the second one describes the traffic that is relayed by a CMAA. The latter comprises e-mails that are first sent to the CMAA, which then sends them on their final destination. Therefore, the e-mail data have to be doubled.

$$\text{Bounce e-mail traffic} = t * 395 * 10^6 * (10/20/50/100) * \emptyset\_bounce\_size * \text{enc}_{\text{bounce}}$$

The term (10/20/50/100) refers to the number of accounts per host (see the explanation above).

$$\text{CMAA administration traffic} = \#messages * \emptyset\_message\_size$$

We now determine the spam e-mail traffic that would lead to “break-even”. Therefore, we have to make (conservative) assumptions about the variable values for the computation of the ham e-mail, bounce e-mail, and administration traffic:

- $\emptyset\_ham\_size$: Many statistics on the Internet mention figures of between 50 and 150 KB. Therefore, we perform sample computations for the values 50, 100, and 150.

- p: The portion of ham e-mails that is sent in a direct SO-RO connection is very difficult to estimate, because this depends on the organizations' white lists. We perform calculations for the values 1%, 5%, 10%, 20%, and 50% respectively.
- t: A bounce e-mail is sent if the account's limit is exceeded. This can happen, if a user accidentally send more e-mails than allowed or if the account has been corrupted and is misused. We assume that 0.1% is an upper bound for t.
- $\emptyset\_bounce\_size$: The average size of a CMAA bounce e-mail should not exceed 5 KB including the CMAA signature.
- $\#messages$: We differ between the following types of messages:
  - messages sent from an SO to a CMAA: We assume that, on average, no more than 1 message per 100 user accounts is sent daily.
  - messages sent from a CMAA to an SO: Each message from an SO has to be answered, which results in 1 message per 100 user accounts. Furthermore, we assume that no more than 1 message per 1000 accounts is sent due to the exceeding of the credit. We further assume that no more than 1 message per 100 accounts is sent due to other administration processes, such as the sending of a removal information.
  - ODB administration messages: We assume that no more than 1 message per 100 ODB entries is sent per day by a CMAA on average. ODB administration messages sent from an SO are not intended.

  We obtain as the total number of messages:

  $0.031*395*10^6*(10/20/50/100)+0.01*395*10^6 = 26.4/52.8/132/264*10^6$.
- $\emptyset\_message\_size$: Taking the message in Figure 9 as a reference message, we assume that the average size of a SOAP message is not larger than 2 KB.
- $\mathrm{enc}_{\mathrm{ham}}, \mathrm{enc}_{\mathrm{spam}}, \mathrm{enc}_{\mathrm{spam}}$: Starting from our e-mail pool, we assume that an upper bound is 1.2 for each of them.

The three traffic parts mentioned above can now be specified as follows:

$$\text{Ham e-mail traffic} = 3.5 * 10^9 * \emptyset\_ham\_size * (2 - p) * 1.2$$

Table 7 shows the traffic data for different values for p and $\emptyset\_ham\_size$.

Table 7: Ham e-mail traffic

ham_size / p	1%	5%	10%	20%	50%
50	3110.4 Tbit	3052.8 Tbit	2976 Tbit	2812.8 Tbit	2342.4 Tbit
100	6220.8 Tbit	6105.6 Tbit	5952 Tbit	5625.6 Tbit	4684.8 Tbit
150	9331.2 Tbit	9158.4 Tbit	8928 Tbit	8438.4 Tbit	7027.2 Tbit

$$\begin{aligned}\text{Bounce e-mail traffic} &= 180.5/361/902.5/1{,}805\,\text{Gbit}\\ \text{CMAA administration traffic} &= 403.2/806.4/2{,}016/4{,}032\,\text{Gbit}\end{aligned}$$

The bounce e-mail traffic and the CMAA adminstration traffic can be neglected with regard to the ham e-mail traffic, so that we can compare the values in Table 7 with the total traffic values that result from Equation (2) with $\text{enc}_{\text{ham}} = \text{enc}_{\text{spam}} = \text{enc}_{\text{bounce}} = 1.2$ and $\emptyset\_ham\_size = 50$ KB, 100 KB, and 150 KB: 4,672.6 Tbit, 6,237.2 Tbit, and 7,801.8 Tbit. If we compare the values, we obtain the following findings:

- The traffic that would occur in our framework is mainly determined by the average size of ham e-mails. This fact is not surprising, because we assume that more than half of all ham e-mails would have to be relayed by a CMAA, which means that the traffic caused by these e-mails is doubled.
- If the average size of the ham e-mails is 50 KB or 100 KB, then today's total e-mail traffic exceeds the traffic (without any spam e-mail traffic) in our framework.
- If we assume that the average ham e-mail size is only 50 KB, our framework needs 1,562 Tbit (p=1%) and 2,330.2 Tbit (p=50%) less traffic by omitting any spam e-mails. Therefore, these values determine the "break-even". Today's spam e-mail traffic is estimated at about 2,752.8 Tbit (with $\text{enc}_{\text{spam}} = 1.2$, see Equation (2)). This means that our framework would have to eliminate at least about $1 - \frac{1562}{2752.8} \approx 43\%$ and about $1 - \frac{2330.2}{2752.8} \approx 15\%$ respectively of today's spam e-mail traffic in order to reduce the total e-mail related traffic.
- If we assume that the average ham e-mail size is 100 KB, we obtain by analogous computations 99% and respectively 44%. This means that for $p = 1\%$ almost all spam e-mail traffic would have to be eliminated.
- If the average size of the ham e-mails is 150 KB, then the traffic (without any spam e-mail traffic) in our framework exceeds today's total e-mail traffic, unless the portion of direct e-mail communication is about p' or more with

$$3.5 * 10^9 * \emptyset\_ham\_size * (2 - p') * 1.2 = 7801.8Tbit => p' \approx 33.8\%.$$

This means that, if about every third e-mail is sent in a direct communication, our framework shows the same traffic (without any spam e-mails) as occurs today (including spam e-mails).

If even every second e-mail is sent directly, our framework would have to eliminate about 72% of current spam, in oder to achieve the “break even” mentioned above, which refers to total traffic (including spam e-mails) in both current infrastructure and proposed infrastructure.

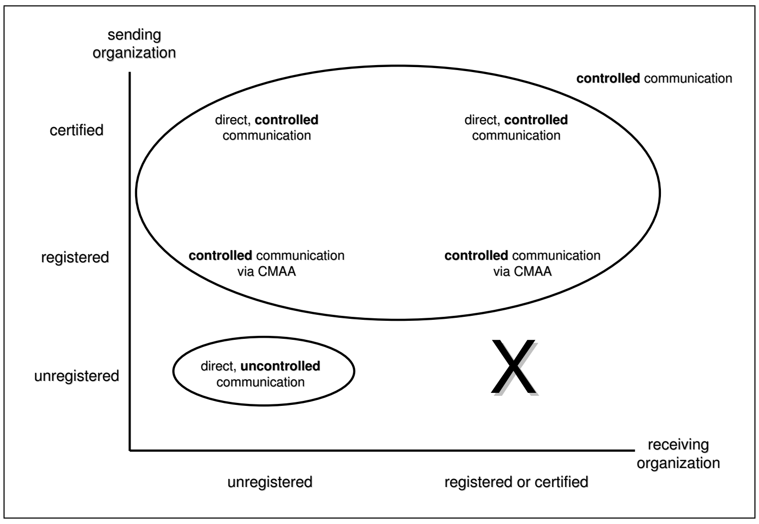
Both an increasing portion of direct e-mail communication, i.e. increasing p, and an increasing number of spam e-mails would lower the “break-even” insofar that that portion of spam e-mail traffic that would have to be eliminated is reduced.

7 Deployment and impact on e-mail communication

A precondition for any deployment of the proposed framework seems to be its adoption by the ISOC, ICANN, and/or large ESPs. This adoption includes both the maintenance of a CO, that has to fulfill the role described in Subsection 3.1, and the propagation of the framework in the Internet e-mail community.

The framework is designed to use both a direct e-mail communication and an indirect one by integrating CMAAs. This flexibility means a scalability of the framework that allows the avoidance of a“big bang” at its introduction, but leaves the (time) schedule for using CMAAs and its grade up to each organization. An ESP, for example, can decide not to use CMAAs at all, to use CMAAs for incoming e-mails, to register for a CMAA’s services, or even to apply for a CMAA certificate. Although no organization is forced to participate in the centralized services, market pressure — assuming that the infrastructure has been widely adopted — will push them to do so, as they are otherwise in danger of being excluded from large parts of the world-wide e-mail communication. This consequence would make the ESP probably unattractive or even unacceptable from the users’ view.

If we categorize communication scenarios according to the SO and RO types, we get those categories illustrated in Figure 7. Organization that are certified or registered are not limited in their e-mail communication. Other organizations would not be allowed to send e-mails to certified or registered organizations, which would usually insist on the registration or certification of the SO. This means that the overall e-mail communication becomes limited. The area of limitation is indicated by the “X” in Figure 7. The grade of limitation will depend on the extent to which the CMAAs will be accepted and used. If the proposed infrastructure is either widely accepted or hardly accepted, then, the


sending organization
controlled communication
certified
direct, controlled communication
direct, controlled communication
registered
controlled communication via CMAA
controlled communication via CMAA
unregistered
direct, uncontrolled communication
X
receiving organization
unregistered
registered or certified
Figure 7: Partitioning of the Internet e-mail communication

limitation is low, because most e-mail communication belongs to one of the categories, which are displayed as ellipses. A high limitation, i.e. "X" indicates a large subset of e-mail communication, would result from a balanced distribution.

8 Drawbacks and limitations

The implementation of the framework requires both organizational and technological modifications of today's Internet e-mail infrastructure. These modifications have to be propagated by Internet organizations and providers in order to become widely accepted. However, even then, the framework has its drawbacks and limitations:

- Spam e-mails are unlikely to be eliminated, because some options for spamming still remain, even if they consume more resources than today: First, e-mail accounts can be set up manually at registered organizations and then used for spamming. If the accounts have a default credit, for example 50 e-mails per day, a spammer would have to set up 20,000 accounts to send 1 million e-mails per day. This takes time and human resources, which would decrease the spammer's profit. Second, if a spammer sets up an account with an increased credit, then, he or she has to provide personal information that has to be validated. In the case of the misuse of such an account, the spammer could be prosecuted. However, in practice, we have to take into account that personal data have been misused and that prosecution is difficult, because some national authorities do not cooperate appropriately. Third, accounts of legitimate users can be compromised by malicious software, such as keylogging programs, which spy out passwords, or software that looks for passwords in the file system. Although a bounce e-mail would be sent to the user's account, it would take some time to fix the user's host. Fourth, organizations that have successfully registered for CMAA services may be corrupt or may tolerate spammers. They would be excluded from CMAA services, and the administrative contact could be prosecuted. However, the same limitations as described above apply. Fifth, an SO that is stored on an RO's whitelist can bypass any CMAA and send an unlimited number of (spam) e-mails. It is up to the RO to cope with this problem by informing or admonishing the SO and/or by even removing it from the whitelist.

- The approaches requires a critical mass of organizations to drive the framework's adoption.
- The DNS becomes an even more critical and important resource than it is today for the following reasons: First,the DNS has to provide entries for public keys of registering organizations. Ideally, the public keys are signed by a trustful organization. Second, LMAP records of SOs and CMAAs have to be added to the DNS. Currently, no single approach has been adopted as a world-wide standard. Third, DNS spoofing would have an impact on the sending of spam e-mails: a CMAA's decision to relay an e-mail depends on the LMAP record. If LMAP data are spoofed, then a third party could send e-mails on behalf of another registered organization. Fourth, the availability of DNS servers is closely related to the availability and functionality of the Internet e-mail infrastructure. This attracts attacks on the availability of these servers, such as Distributed Denial of Service (DDoS) attacks.
- The CMAAs' systems represent a critical resource, too: First, the availability of the relays and administration servers is critical with regard to the operational maintenance of large parts of Internet e-mail traffic. Therefore, the consequences of a successful DDoS attack are tremendous. Second, the servers have to handle a huge amount of traffic and requests (see Subsection 6.2). This requires a careful implementation of load balancing systems, if e-mail communication is not to become (timely) inefficient. Third, the CMAAs' CDBs contain large numbers of valid e-mail addresses and have to be protected from unauthorized accesses. Address harvesters will make ambitious efforts to get access to CDBs. Options for protecting CDB e-mail addresses from being read unauthorized are discussed in Subsection 4.1.

References

Anti-Spam Technical Alliance (ASTA): 2004, Anti-spam technical alliance technology and policy report, *Technical report.*

Brightmail: 2003, The State of Spam Impact & Solutions.

Bundesamt für Sicherheit in der Informationstechnik (BSI): 2005, Antispam - Strategien: Unerwünschte E-mails erkennen und abwehren.

Cohen, W.: 1996, Learning rules that classify e-mail, *Papers from the AAAI Spring Syposium on Machine Learning in Information Access*, pp. 18–25.

Crawford, E., Kay, J. and McCreath, E.: 2001, Automatic induction of rules for e-mail classification, *Proceedings of the Sixth Australasian Document Computing Symposium.*

Damiani, E., De Capitani di Vimercati, S., Paraboschi, S. and Samarati, P.: 2004, P2p-based collaborative spam detection and filtering, *Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, pp. 176–183.

DeKok, A.: 2004, Lightweight MTA Authentication Protocol (LMAP) Discussion and Applicability Statement, *Internet draft*, IETF Network Working Group.

Email Service Provider Coalition: 2003, Project Lumos: A Solutions Blueprint for Solving the Spam Problem by Establishing Volume Email Sender Accountability, *Technical report.*

Evett, D.: 2006, Spam Statistics 2006, http://spam-filter-review.toptenreviews.com/spam-statistics.html.

Ferris Research: 2005, The Global Economic Impact of Spam, 2005. Report #409.

Freed, N. and Borenstein, N.: 1996, Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, *RFC 2045*, IETF Network Working Group.

Gauthronet, S. and Etienne, D.: 2001, Unsolicited Commercial Communications and Data Protection.

Graham, P.: 2002, A Plan For Spam, http://www.paulgraham.com/spam.html.

Graham, P.: 2003, Better Bayesian Filtering, *Proceedings of the 2003 Spam Conference.*

ICANN: 2004, new sTLD RFP Application .mail, Part B. Application Form, *Technical report.*

Internet Systems Consortium: 2006, Internet Domain Survey, Jan 2006, http://www.isc.org.

Ironport: 2006, Internet Email Traffic Emergency: Spam Bounce Messages are Compromising Networks, http://www.ironport.com/bouncereport/.

ITU: 2005, ITU Survey on Anti-spam Legislation Worldwide.

Klensin, J.: 2001, Simple Mail Transfer Protocol, *RFC 2821*, IETF Network Working Group.

Leibzon, W.: 2005, Email Security Anti-Spoofing Protection with Path and Cryptographic Authentication Methods, http://www.metasignatures.org/path_and_cryptographic_authentication.htm.

MessageLabs: 2006, MessageLabs Intelligence Report: Spam Intercepts Timeline May 2006, http://www.messagelabs.com/publishedcontent/publish/threat_watch _dotcom_en/threat_statistics/spam_intercepts/DA_114633.chp.html.

Moustakas, E., Ranganathan, C. and Duquenoy, P.: 2005, Combating Spam through Legislation: A Comparative Analysis of US and European Approaches, *Proceedings of the Second Conference on Email and Anti-Spam(CEAS 2005).*

Myers, J.: 1997, Simple Authentication and Security Layer (SASL), *RFC 2222*, IETF Network Working Group.

Myers, J.: 1999, SMTP Service Extension for Authentication, *RFC 2554*, IETF Network Working Group.

NOIE: 2002, The Spam Problem and How it Can be Countered An Interim Report by NOIE, *Technical report.*

Nucleus Research: 2003, Spam: The Silent ROI Killer.

OECD: 2004a, Background Paper for the OECD Workshop on Spam.

OECD: 2004b, Report of the 2nd OECD Workshop on Spam.

OECD: 2004c, Report on non-OECD Countries' Spam Legislation.

Pantel, P. and Lin., D.: 1998, SpamCop - A Spam Classification & Organization Program, *Proceedings of AAAI-98 Workshop on Learning for Text Categorization.*

Sahami, M., Dumais, S., Heckerman, D. and Horvitz, E.: 1998, A Bayesian Approach to Filtering Junk E-Mail, *Proceedings of AAAI-98 Workshop on Learning for Text Categorization.*

Schryen, G.: 2006a, Do anti-spam measures cover the e-mail communication network? A formal approach, *The Journal of Information Systems Security* . accepted for publication.

Schryen, G.: 2006b, A formal approach towards assessing the effectiveness of anti-spam procedures, *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*.

Symantec: 2006, Spam-Statistiken, http://www.symantec.com/region/de/PressCenter/spam.html.

Telcordia Technologies, Inc.: 2001, Internet Hosts Reach 100 Million Worldwide, http://emailwire.com/news/int374.shtml.

Tompkins, T. and Handley, D.: 2003, Giving E-mail Back to the Users: Using Digital Signatures to Solve the Spam Problem, *FirstMonday* **8**(9).

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

von Ahn, L., Blum, M. and Langford, J.: 2004, Telling Humans and Computers Apart Automatically, *Communications of the ACM* **47**(2), 57–60.

W3C: 2001, SOAP Security Extensions: Digital Signature, http://www.w3.org/TR/SOAP-dsig/. W3C Note.

W3C: 2003, SOAP Version 1.2 Part 1: Messaging Framework, http://www.w3.org/TR/soap12-part1/.

Zhou, F., Zhuang, L., Zhao, B., Huang, L., Joseph, A. and Kubiatowicz, J.: 2003, Approximate object location and spam filtering on peer-to-peer systems, *Proceedings of ACM/IFIP/USENIX International Middleware Conference*.

A XML schemas and SOAP messages

```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.cmaa.int"
xmlns="http://http://www.cmaa.int"
elementFormDefault="qualified">

<xs:element name="CMAA_message"">
  <xs:complexType>
    <xs:attribute name="message_type" type="xs:string"/>
    <xs:sequence>
      <xs:element name="object" type="xs:string"/>
      <xs:element name="remark" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

Figure 8: XML schema for messages directed to an SO

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">

 <env:Header>
    <sec:Signature
      xmlns:sec="http://schemas.xmlsoap.org/soap/security/2000-12"
      env:actor="<CMAA URI>"
      env:mustUnderstand="1">
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:SignedInfo>
          <ds:CanonicalizationMethod
            Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026">
          </ds:CanonicalizationMethod>
          <ds:SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
          <ds:Reference URI="#Body">
            <ds:Transforms>
              <ds:Transform
Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026"/>
            </ds:Transforms>
            <ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <ds:DigestValue>...</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>...</ds:SignatureValue>
      </ds:Signature>
    </sec:Signature>
 </env:Header>

 <env:Body>
    xmlns:sec="http://schemas.xmlsoap.org/soap/security/2000-12"
    sec:id="Body">
   <p:setup_account xmlns:p="http://www.cmaa.int/setup_request">
   <p:account>guido@schryen.net</p:account>
   <p:max>200</p:max>
   <p:setup_org>terabyte.com</p:setup_org>
   <p:holder>
     <p:lastname>Schryen</p:lastname>
     <p:firstname>Wolfgang</p:firstname>
     <p:street>Linderner Bahn 21</p:street>
     <p:city>Geilenkirchen</p:street>
     <p:zip>52511</p:zip>
     <p:country>Germany</p:country>
   </holder>
  </p:setup_account>
 </env:Body>

</env:Envelope>
```

Figure 9: SOAP message for setting up a CMAA record

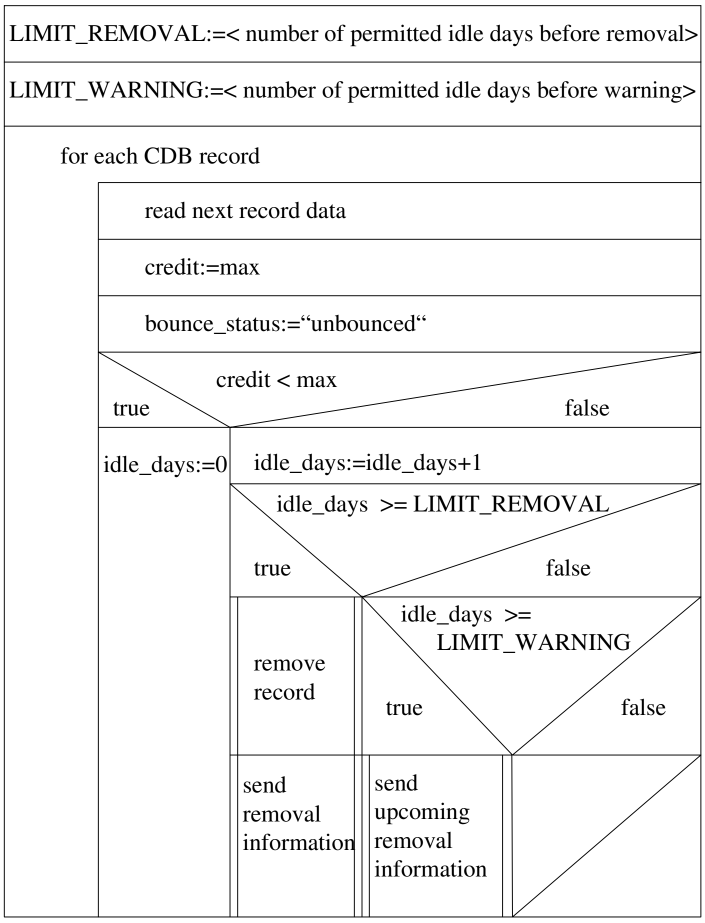
```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.cmaa.int"
xmlns="http://http://www.cmaa.int"
elementFormDefault="qualified">

<xs:element name="setup_account">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="account" type="xs:string"/>
      <xs:element name="max" type="xs:integer"/>
      <xs:element name="setup_org" type="xs:string"/>
      <xs:element name="holder" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="lastname" type="xs:string"/>
            <xs:element name="firstname" type="xs:string"/>
            <xs:element name="street" type="xs:string"/>
            <xs:element name="city" type="xs:string"/>
            <xs:element name="zip" type="xs:string"/>
            <xs:element name="country" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

Figure 10: XML schema of the SOAP message for setting up a CMAA record

```
SOAP confirmation message:

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
 <env:Body>
   <p:CMAA_message message_type="SETUP_SUCCESS"
      xmlns:p="http://www.cmaa.int/answer">
     <p:object>guido@schryen.net</p:object>
  </p:CMAA_message>
 </env:Body>
</env:Envelope>


SOAP rejection message:

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
 <env:Body>
  <p:CMAA_message message_type="SETUP_SUCCESS"
    xmlns:p="http://www.cmaa.int/answer">
     <p:object>guido@schryen.net</p:object>
     <p:remark>Authorization failed</p:remark>
  </p:CMAA_message>
 </env:Body>
</env:Envelope>
```

Figure 11: SOAP confirmation/rejection message on behalf of the CMAA

B Nassi-Shneiderman diagrams



LIMIT_REMOVAL:=< number of permitted idle days before removal>
LIMIT_WARNING:=< number of permitted idle days before warning>
for each CDB record
read next record data
credit:=max
bounce_status:="unbounced"
credit < max
true
false
idle_days:=0
idle_days:=idle_days+1
idle_days >= LIMIT_REMOVAL
true
false
remove record
idle_days >= LIMIT_WARNING
true
false
send removal information
send upcoming removal information

Figure 12: Nassi-Shneiderman diagram for resetting the credits of CMAA records and for tracing for idle CMAA accounts

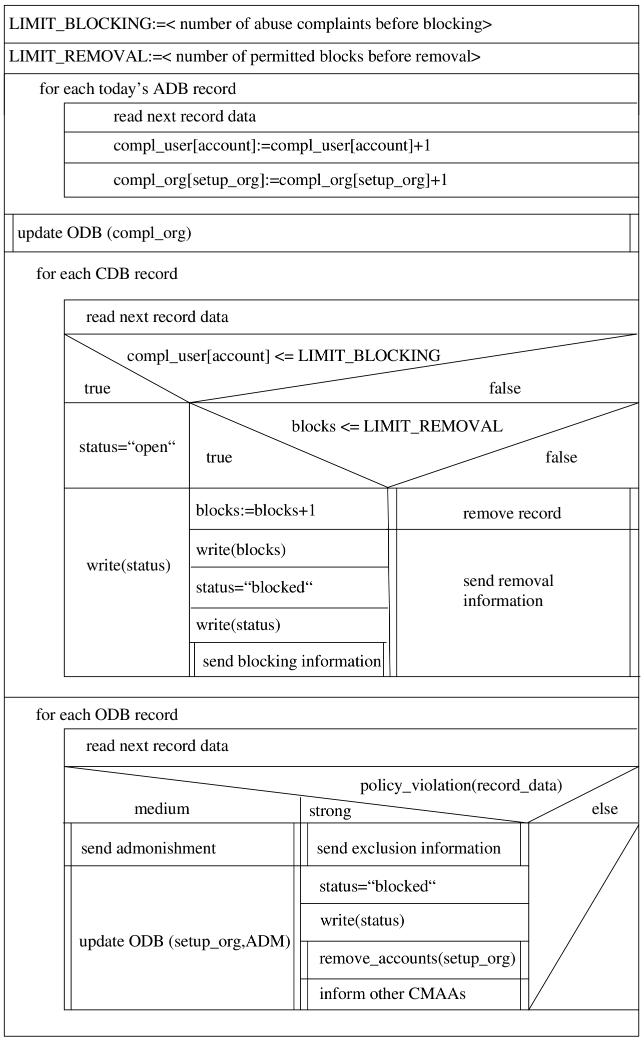

LIMIT_BLOCKING:=< number of abuse complaints before blocking>
LIMIT_REMOVAL:=< number of permitted blocks before removal>
for each today's ADB record
read next record data
compl_user[account]:=compl_user[account]+1
compl_org[setup_org]:=compl_org[setup_org]+1
update ODB (compl_org)
for each CDB record
read next record data
compl_user[account] <= LIMIT_BLOCKING
true
false
status="open"
blocks <= LIMIT_REMOVAL
true
false
write(status)
blocks:=blocks+1
write(blocks)
status="blocked"
write(status)
send blocking information
remove record
send removal information
for each ODB record
read next record data
policy_violation(record_data)
medium
strong
else
send admonishment
update ODB (setup_org,ADM)
send exclusion information
status="blocked"
write(status)
remove_accounts(setup_org)
inform other CMAAs

Figure 13: Nassi-Shneiderman diagram for blocking CMAA accounts and/or SOs