ACCOUNTABLE ANONYMOUS E-MAIL

Vincent Naessens*, Bart De Decker, Liesje Demuynck[1]
K.U.Leuven, Department of Computer Science, Celestijnenlaan 200A, 3001 Leuven, Belgium,
**K.U.Leuven, Campus Kortrijk (KULAK), E. Sabbelaan 53, 8500, Kortrijk, Belgium*

Abstract: Current anonymous e-mail systems offer unconditional anonymity to their users which can provoke abusive behaviour. Dissatisfied users will drop out and liability issues may even force the system to suspend or cease its services. Therefore, controlling abuse is as important as protecting the anonymity of legitimate users when designing anonymous applications.

This paper describes the design and implementation AAEM, an accountable anonymous e-mail system. An existing anonymous e-mail system is enhanced with a control mechanism that allows for accountability. The system creates a trusted environment for senders, recipients and system operators. It provides a reasonable trade-off between anonymity, accountability, usability and flexibility.

Key words: privacy, anonymity, accountability, control

1. INTRODUCTION

Anonymous e-mail systems serve many purposes ranging from making public political statements under oppressive governments to discussing topics that might otherwise lead to embarrassment, harassment or even loss of jobs in more tolerant political environments [1]. However, unconditional anonymous e-mail systems (such as remailers) can also be used for sending offensive mail, spam, black mail, copyrighted material, child pornography...

[1] Research assistant of the Research Foundation – Flanders (FWO – Vlaanderen)

Legal actions against this service may even force it to shut down. Therefor, controlling abuse is as important as protecting the anonymity of legitimate users. Both considerations play a central role in the design of the accountable anonymous e-mail system (AAEM). Anonymity should always be guaranteed to legitimate users but extra controls are necessary to prevent or at least discourage abuse.

The paper is organized as follows: section 2 describes a general anonymous credential system; these credentials will be used in the e-mail system that is designed in section 3. Section 4 describes and evaluates a prototype of the system. An overview of related work is given in section 5. We conclude in section 6 with a summary of the major achievements.

2. ANONYMOUS CREDENTIALS

This section describes Idemix [11,12], a general anonymous credential system. Idemix helps to realize anonymous yet accountable transactions between users and service providers. The credential system is used to introduce anonymity control in the AAEM system. A simplified version of the Idemix protocols is presented here. Not all inputs of the protocols are described. The outputs of interactive protocols are not always visible to all parties involved in the protocol.

Nym registration. A nym is the pseudonym by which the user wants to be known by an organization. Idemix has two kinds of nyms: ordinary nyms and rootnyms. The user establishes a nym with an organization based on his master secret[2].

Nym registration (for registering ordinary nyms). $Nym_{UO}=RegNym()$. Note that the rootNym (see below) is hidden in every nym of the user.

RootNym registration (for registering rootnyms). $RootNym_{UR} = RegRootNym(Sig_{UR}, Cert_{UC})$. The user signs the established nym with his signature key, which is certified by an external certificate (which links the user's public key with his identity). Hence, the organization holds a provable link between the rootnym and the identity certified by the certificate.

Credential issuing. $Cred_{UI} = IssueCred(Nym_{UI}, CredInfo_{UI})$. After establishing a nym with an organization, the user may request a credential on

[2] Note that all the user's nyms and credentials are linked to the user's master secret. Hence, sharing one credential means sharing all other credentials as well.

that nym. Credentials can contain additional information: show options (e.g. one/limited/multi-show) and attributes (e.g. age, citizenship, expiration date, ...).

Credential showing. $(Transcript_{UV}, Msg_{UV}) = CredShow(Nym_{UV}, Cred_{UI}, CredShowFeatures_{UV\text{-}I})$. The user proves to the verifying organization V that he owns a credential issued by organization I. A credential show can have several features:

- The credential is shown relative to another nym. The (anonymous) user proves that the nym on which the credential is issued and a nym by which he is known by the verifier, belong to the same user (they are based on the same user secret).
- Local and/or global deanonymization is allowed (cfr. below).

In addition, a user may choose to prove any attribute (or a property of these attributes). Showing a credential results in a transcript for V which can be used later in double spending checking and deanonymization protocols. During a credential show, a message can be signed, which provably links the message to the transcript of the credential show. The following anonymity properties are valid:

- Two or more credential shows of the same credential can not be linked together (unless the credential is a one-show credential).
- A credential show does not reveal the nym on which it was issued.

Deanonymization. Transcripts of anonymous credential shows can be de-anonymized by including a verifiable encryption of the user's nym (local deanonymization) or rootNym (global deanonymization). Thus, only parties that have the deanonymization keys can deanonymize credential shows

Local deanonymization.
$(Nym_{UI}, DeAnonTranscript_{D\text{-}UV}) = DoLocalDeanon(Transcript_{UV})$. If a credential show is locally deanonymizable, the nym for which the credential was issued can be revealed by a deanonymizer D. A deanonymization transcript contains a provable linking between the transcript and the nym.

Global deanonymization.
$(RootNym_{UR}, DeAnonTranscript_{D\text{-}UV}) = DoGlobalDeanon(Transcript_{UV})$. If the credential show was globally deanonymizable, the user's rootnym can be revealed.

Credential Revocation. *RevokeCred(*Nym_{UI}*,* $Cred_{UI}$*).* The issuer can also revoke credentials issued on a nym.

3. ACCOUNTABLE ANONYMOUS E-MAIL

First, the requirements of the players in the system are described. The requirements analysis results in the design of an anonymous mail system enhanced with a control mechanism that allows for accountability. The roles in the system are described in the second paragraph. Third, we describe the protocols used in the different phases. Finally, we evaluate the trust properties in the system.

3.1 Requirements

User requirements. These requirements depend on the role the users play in the the mail system. Senders want their privacy to be protected whereas recipients mainly have control requirements. Whereas current remailer systems mainly focus on the former, our design considers the concerns of both parties.

The control requirements of *recipients* are twofold:

Spam control measures. Recipients don't want spam in their mailbox. The mail system should take measures to discourage this kind of abuse.

Accountability for criminal mail. It should be possible to prosecute the sender of a criminal mail. The mail system should guarantee that the identity of the sender can be revealed (i.e. the user is accountable).

The anonymity requirements of mail *senders* will only be met as long as the sender abides by the rules (no spam or criminal mail).

Unlinkability of a mail to the initiator.

Unlinkability of different mails from the same initiator.

System requirements. The system requirements are twofold:

Offering good service to users. The mail system wants to offer a good service in order to attract users. Therefor, the mail system contracts its users to meet the requirements of their users.

Limited use of resources. To be able to meet the accountability requirements, evidence will be stored in the system. The amount of evidence stored by the AAEM-system itself should be minimal.

Law Enforcement requirements. The government may require accountability of misbehaving users. Such users have sent illegal mails (such as illegal music files, child porn, ...) or criminal mails (such as death threats, bribery, blackmail ...).

3.2 Roles

Registrar (for registration). The registrar knows a provable link between a user's true identity and his rootNym. To increase trust in the system, the registrar is independent of the AAEM-system. Its cooperation is required to identify the sender of an anonymous e-mail.

AAEM system infrastructure (core of the mail system).

- **Activation Manager** (for activation). The Activation Manager handles the (anonymous) user registrations, and if the AAEM-system is not for free, also deals with payments. Payment can be anonymous; however, it is not a pre-requisite to fulfil the requirements.
- **Mail guard** (for mailing). The Mail Guard imposes strict access control to the AAEM-system (only registered users are authorized to use the services of the system) and adds verifiable proofs to the message guaranteeing that the sender of the message can be identified under certain conditions.
- **Complaint handler**. The Complaint Handler handles suspected (unacceptable/criminal) mail that is sent through the mail system. Complaints are sent by recipients of such mail. The Mail Guard can also pass suspected mail to the Complaint Handler.

Deanonymization granting/handling infrastructure.

- **Law Enforcement entity (or Justice).** The role of the Law Enforcement Entity (LE) is to verify whether the identity of a user behind a nym may be revealed.
- **Arbiter.** The arbiter's role is to verify whether an e-mail fulfills the local or global de-anonymization condition.
- **Deanonymizer.** This authority can retrieve the pseudonym of the sender of an anonymous e-mail message.

Communication infrastructure.

- **Anonymous connection system (AC).** The connection between the sender and some of the other participants need to be anonymous.
- **Re**
mailers (REM). The remailers constitute the existing anonymous e-mail system. The AAEM system forwards mail towards the recipient through a remailer system.

Users of the AAEM system.

- **(Anonymous) Sender.** The sender is entitled to send e-mail anonymously (when he is registered to the system). As long as he abides by the rules, his anonymity will be respected.
- **Recipient.** The owner of a mailbox and e-mail address. The recipient may refuse to accept anonymous e-mails unless they are "stamped" by a trusted Mail Guard. If the email is spam or contains illegitimate or criminal content, the recipient may file a complaint with the AAEM-system.

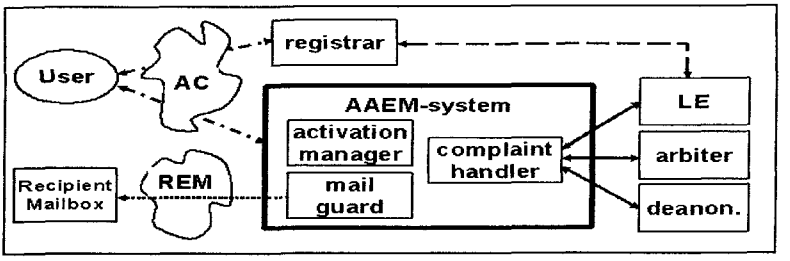

Figure 1. Overview of the AAEM system.

3.3 Protocols

This section describes the protocols used in different phases.

Registration. In this phase, the user contacts the registrar to obtain a root credential. The user first establishes a rootNym and signs that rootNym with an external certificate (issued by a trusted CA). The registrar stores the identity proof and issues a root credential on the rootNym. The root credential can be shown multiple times to the Activation Manager.

U: $Cert_{UC}$ from C (an external Certification Authority)

$U \leftrightarrow R$: $RootNym_{UR} = RootNym(Sig_{UR}, Cert_{UC})$
R: stores $[Sig_{UR}, Cert_{UC}, RootNym_{UR}]$
$U \leftrightarrow R$: $Cred_{UR} = IssueCred(RootNym_{UR}, ['ACT'])$

Activation of Anon-Email Service. Each user has to activate the anonymous mail service with the Activation Manager before he can send mail. The Activation Manager issues a *send credential*, required to send mail. Before the user receives a send credential, he must prove that he has previously registered (possesses a valid root credential) and solve an activation puzzle.

To achieve these goals, the user first establishes an anonymous communication channel and registers a nym with the Activation Manager. The user then shows his root credential relative to that nym. This prevents unregistered users to activate the mail service. The credential show is undeanonymizable. The Activation Manager verifies the credential show. The user then solves the activation puzzle. The puzzle discourages users to activate the service several times[3].

$U \leftrightarrow M$: $Nym_{UM} = RegNym()$
$U \leftrightarrow M$: $(Transcript_{UM\text{-}R}, Msg_{UM}) =$
$CredShow(Nym_{UM}, Cred_{UR}, CredShowFeatures_{UM\text{-}R})$
with $CredShowFeatures_{UM\text{-}R}$=[LDeDanon=null, GDeAnon=null]
and Msg_{UM} = contract between U and AAEM, may contain explanation of acceptable use policy.
$U \leftrightarrow M$: activation procedure (solving puzzle)
$U \leftrightarrow M$: $Cred_{UM} = IssueCred(Nym_{UM}, ['SEND'])$
M: stores $[Transcript_{UM\text{-}R}, Nym_{UM}, Cred_{UM}]$

Sending anonymous mail. Sending mail is conditionally anonymous. A user is anonymous as long as he abides by the rules. If the system is used to send spam mail, the user's send credential will be revoked; if criminal mail is sent through the system, the sender can be identified and prosecuted.

The sender is responsible for removing identifying headers before contacting the mail guard, who will verify the sender's send credential. During the credential show, the mail is signed, which provably links the mail to the transcript of the credential show. The transcript is locally and globally

[3] The activation puzzle can be omitted if the user has to go through a prior payment phase, in which he receives a one-show payment credential. In that case, the user must pay in order to activate the mail service.

deanonymizable. The Mail Guard verifies the credential show and attaches the transcript to the mail. The Mail Guard then forwards the mail to the recipient mailbox over a remailer network.

$U \leftrightarrow G$: $(Transcript_{UG\text{-}M}, Msg_U) =$
$CredShow(null, Cred_{UM}, CredShowFeatures_{UG\text{-}M})$
with $CredShowFeatures_{UG\text{-}M} =$
$[LDeAnon=[DCond=Unacceptable|Criminal, Arbiter=A],$
$GDeAnon=[DCond=Criminal, Arbiter=A]]$
and Msg_U = message to be sent anonymously to the recipient
G: forwards $[Msg_U, Sig_G(Transcript_{UG\text{-}M})]$ to recipient r through REM

Receiving anonymous mail. The recipient checks the validity of the $Transcript_{UG\_M}$ with respect to the message Msg_U. If the verification fails, the message is discarded. If the verification is successful, the user reads the message. If the message is abusive (unacceptable or criminal), the recipient forwards the mail to the complaint handler.

Unacceptable behavior (Spam, ...). If a user has sent spam, the send credential of that user should be revoked. Revoking the send credential consists of three steps:

Decision of Arbiter. The recipient sends the suspected mail (mail contents and transcript) to AAEM system. The Complaint Handler signs the mail and forwards the request to the Arbiter. The Arbiter first verifies the validity of the mail w.r.t. the transcript. It then verifies whether the mail is really unacceptable. The Arbiter returns his signed decision. If the mail is unacceptable, the Complaint Handler informs the Deanonymizer.

Disclosing Nym. The Deanonymizer receives a signed message from the Complaint Handler. The message contains the Arbiter's decision (i.e. "Unacceptable"), the mail and the transcript. The Deanonymizer verifies the advice, and if positive, locally deanonymizes the transcript. He then returns the nym and a deanonymization transcript to the Complaint Handler. The Deanonymizer also stores the Arbiter's signed decision.

Credential revocation. When the Complaint Handler receives the nym from the Deanonymizer, the mail system actually revokes the credential issued on the nym. The victim is also kept informed.

$r \rightarrow C$: *[ComplaintSpam,* $Mail_U$*]*
with $Mail_U = [Msg_U, Transcript_{UG\text{-}M}]$
and $Transcript_{UG\text{-}M}$ *contains LDeAnon=[Unacceptable|Criminal, A]*

$C \rightarrow A$: $Sig_C(Mail_U$, *Unacceptable?)*
$C \leftarrow A$: $Sig_A(Mail_U$, *Unacceptable)*

$C \rightarrow D$: $Sig_A(Mail_U$, *Unacceptable)*

D: $(Nym_{UM}, DeAnonTranscript_{D\text{-}UG}) = DoDeAnonLocal(Transcript_{UG\text{-}M})$
D: stores $Sig_A(Mail_U$, *Unacceptable)*

$C \leftarrow D$: $[Nym_{UM}, DeAnonTranscript_{D\text{-}UG}]$

C: $RevokeCred(Nym_{UM}, Cred_{UM})$
C stores: $[Sig_A(Mail_U$, *Unacceptable),* $Nym_{UM}, DeAnonTranscript_{D\text{-}UG}]$

$r \leftarrow C$: Sig_C*(Sender=BANNED,* $Sig_A(Mail_U$, *Unacceptable))*

Criminal behavior. Criminal behavior can be detected by the recipient (e.g. blackmail, stalking, ...) or by a mail system component (e.g. illegal content...). In both cases, the identity of the mail sender should be revealed. In addition, the user's send credential can be revoked. Revocation of a send credential is described above. Revealing the identity of the sender requires the following steps:

Decision of Arbiter(see above).

Disclosing RootNym. If the mail is criminal, the Arbiter convinces the deanonymizer to reveal the rootNym behind the transcript. The deanonymizer globally deanonymizes the transcript and returns the rootNym and the deanonymization transcript to the Complaint Handler.

Revealing identity. The Complaint Handler forwards the evidence to LE. LE then orders the Root Authority to reveal the identity of the user behind the rootNym. LE stores the evidence that proves the link between the sender and the criminal mail.

$r \rightarrow C$: *[ComplaintCriminal,* $Mail_U$*]*
with $Mail_U = [Msg_U, Transcript_{UG\text{-}M}]$
and $Transcript_{UG\text{-}M}$ *contains GDeAnon=[Criminal, A]*

$C \rightarrow A$: $Sig_C(Mail_U$, *Criminal?)*

$C \leftarrow A$: $Sig_A(Mail_U, Criminal)$

$C \rightarrow D$: $Sig_A(Mail_U, Criminal)$

D: $(RootNym_{UR}, DeAnonTranscript_{D\text{-}UG}) =$
$DoDeAnonGlobal(Transcript_{UG\text{-}M})$
D: stores $Sig_A(Mail_U, Criminal)$

$C \leftarrow D$: $[RootNym_{UR}, DeAnonTranscript_{D\_UG}]$

$C \rightarrow LE$: $[Sig_A(Mail_U, Criminal), RootNym_{UR}, DeAnonTranscript_{D\text{-}UG}]$

$LE \rightarrow R$: $Sig_{LE}(RootNym_{UR}, GETIDENTITY)$
$LE \leftarrow R$: $[RootNym_{UR}, Sig_{UR}, Cert_{UC}, Msg_{UR}]$

LE stores $[RootNym_{UR}, Sig_{UR}, Cert_{UC}, Msg_{UR}]$

3.4 Properties

This section focuses on the trust properties in the system. The mail system creates a trusted environment for senders, recipients and administrators of an AAEM system.

Sender. First, the sender may trust that different mails cannot be linked by the AAEM system. Although send credentials are issued by the AAEM system, credential shows are unlinkable. Therefore, different mails from the same user can not be linked by AAEM. Note that an anonymous communication infrastructure is required. Second, the sender may trust that his send credential will not be revoked as long as he does not send abusive mail. Three parties are involved in revoking send credentials: AAEM, A and D. D will only locally deanonymize the transcript after permission of A. AAEM can only revoke the send credential related to the transcript after local deanonymization. Moreover, AAEM needs A's permission to revoke the credential. Nevertheless, trust is required in a righteous Arbiter. AAEM and user can possibly negotiate which Arbiter to involve before a credential show. Third, revealing the user's identity is only possible with cooperation of external entities: A, LE, D and R. D only globally deanonymizes a transcript after A's approval. R only reveals the link between the rootNym and the identity of the user after approval of LE.

Recipient. A valid transcript guarantees that a mail is locally and globally deanonymizable. Recipients also know the verifier of valid

transcripts (i.e. AAEM) and the deanonymization conditions. Moreover, the recipient can block mail containing no or invalid transcripts.

AAEM, D and R. AAEM can not be liable as long as it observes the rules (i.e. respects the decision of the Arbiter). AAEM stores evidence about unacceptable/criminal mail and the judgement of A about the mail. D also stores such evidence. In case of criminal mail, R stores LE's judgement.

4. PROTOTYPE

4.1 Description

Client side infrastructure.

The *application layer* consist of three components. The registration/activation module receives registration/ activation requests from the user and passes them to the credential layer. The mail client is configured to forward mail to a local SMTP server (running at the sender's machine). The mail server filters any identifying headers and passes the message to the credential layer.

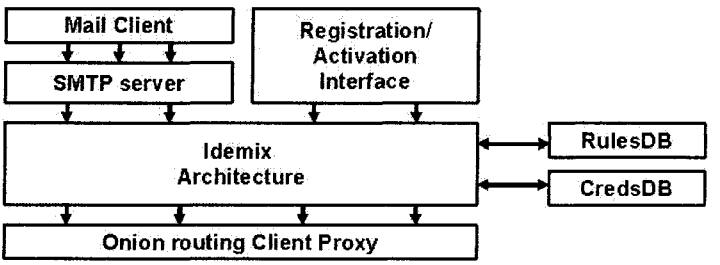

Figure 2. Client side infrastructure

The *credential layer* implements Idemix credential protocols (credential showing, credential issuing, ...). The credential layer requires two databases in order to execute a credential protocol. First, the CredsDB stores credentials (root credential, activation credentials, mail credentials). This database is updated as (new) user credentials are retrieved/revoked. Second, the RulesDB specifies the rules for credential showing/credential issuing during each phase (registration, activation, sending mail). The RulesDB is configured at set up time. Idemix requests are passed to the communication layer.

The *communication layer* deals with anonymous connections to the AAEM system. An onion routing proxy[2,3] is inserted at communication level. In the current implementation, the client composes an anonymous path to the AAEM system. In an alternative implementation, the communication layer can contact an external communication proxy that sets up anonymous connections. However, the latter implementation has different anonymity properties. Access configurations to anonymous connection systems are discussed in [2,3]. The client side also consists of a module that verifies transcripts and sends complaints to the AAEM system.

Core of AAEM system.

The *activation manager* and the *mail guard* are implemented as two different Idemix organizations. The Mail Guard verifies mail, stores the transcripts as attachment and passes them to a Mixmaster remailer proxy (running at the same machine). The remailer proxy chooses a chain of remailers, recursively encrypts the message and forwards the message through the remailer system.

The *Complaint Handler* receives complaints from recipients. The Complaint Handler forwards them to an arbiter and/or a law enforcement entity. This depends on the type of complaint.

4.2 Evaluation

Anonymity. Anonymity at application level is achieved by using anonymous credentials as building block. However, anonymity at application level is useless without support at the communication level. A global passive adversary is the most commonly assumed threat when analyzing anonymity at this level. No current practical low-latency, bi-directional system (i.e. anonymous connection system) does protect against such a strong adversary.

The prototype implements anonymous connections between the sender and the AAEM system. The credential protocols require a real-time, bidirectional communication channel. However, sender and recipient are the real endpoints of communication. The AAEM system forwards mails to recipients over a remailer system that resists global attacks. Thus, global attackers cannot link the endpoints of communication.

Usability/Deployability. To be deployed and used in the real world, the system is not expensive to run:

The design does not place a heavy liability burden on AAEM operators (as discussed in section 3.4).

- Decentralized storage of mail transcripts reduces the number of disk space required by AAEM. The number of stored activation transcripts is linear to the number of activations. The amount of evidence stored by AAEM is linear to the number of accepted complaints. However, the system discourages multiple activations and abusive behaviour.
- The system extends an existing infrastructure. AAEM uses a pre-existing network of anonymous remailers and anonymous connections.
- Once a user has installed the client software, he only has to change the location of the outgoing SMTP server in his mail client.

Flexibility/transparency. The components are loosely coupled by a layered design. Transparency is achieved between the mail component and the communication component. The communication component can easily be replaced by another implementation. Second, the system foresees a loose coupling between different entities: the Arbiter and the Law Enforcement entity do not require any knowledge about Idemix and the structure of the mail system to judge complaints. Even the deanonymizer doesn't require knowledge about the structure of the mail system. To simplify the complaint handling procedure, the deanonymizer itself can be the Arbiter. This requires additional trust in the deanonymizer.

5. RELATED WORK

Our work on AAEM was largely motivated by the problems of current anonymous mail systems and tries to be a reliable extension of current remailer systems. The Mail Guard functions as front end to a remailer system. Our implementation uses Mixmaster[5, 10] remailers. However, only the communication proxy at the Mail Guard has to be re-implemented to work well with other types of remailers. If replies should be supported, the SMTP server at the client side also has to be re-implemented. This server must know the available remailers in order to build a reply structure. The current implementation does not support replies: the SMTP server just removes the "return-path" header.

The first anonymous mail system open to the public was *anon.penet.fi* [9]. Unfortunately, penet did not use encryption. Moreover, only one machine needs to be compromised in order to break the anonymity. *Type-1 remailers*, also called Cypherpunk remailers, were developed to address many shortcomings of the penet system. Type-1 remailers have public keys with which incoming messages are encrypted. A message can be sent through a

chain of type-1 remailers, having been successively encrypted for each of them. Each remailer in a chain knows only the identity of the previous remailer and the next remailer in the chain. The system also supports reply functionality.

Type-2 remailers[5,10] offer several improvements in security over type-1 remailers. These improvements make hop-by-hop analysis considerably harder. They include fixed size messages, replay detection and better reordering of messages at remailers. Type-2 remailers do not support replies to unknown destinations.

Type-3 remailers[8], also called Mixminion remailers, support secure single-use reply blocks. Mix nodes cannot distinguish Mixminion forward messages from reply messages. Directory servers allow users to learn public keys and performance statistics of participating remailers. Mixminion provides a mechanism for each node to specify an exit policy (open exit nodes versus middleman exit nodes) and describes a protocol which allows recipients to opt-out of receiving mail from remailers. However, this requires recipients to send an opt-out request to each open exit node. This is very difficult to realize in practice as new remailers become available. Moreover, if receiving mail is opt-out, non-abusive mail is also retained. Our approach is to discard only anonymous messages without a valid transcript. Senders of abusive mail can be held accountable.

Nymserv[1] is an e-mail pseudonym server: the server keeps a public key and a reply block for every nym. Nymserv also functions as front end and back end to a remailer system. Mail sent from the server to a user leaves through a chain of Cypherpunk remailers; requests to create nyms and to send mail from them arrives through a chain of Cypherpunk remailers. Nyms of abusive users can be revoked. Nymserve also uses a high-latency anonymous communication system. However, different mails from the same user can be linked. Moreover, only a limited amount of control is possible: users can not be accountable for sending abusive mail.

6. CONCLUSION

The presented mail system considers both anonymity requirements of senders and accountability requirements of recipients. A reasonable trade-off between these interests is achieved.

An acceptable level of anonymity at communication level is achieved by reusing existing solutions: anonymous connections and remailers. An anonymous credential system is used as building block for accountability of application specific data/actions. Moreover, the credential system is loosely coupled to the application.

Trust is achieved by splitting responsibilities over different entities and accurate complaint handling procedures. However, a trusted external party is still required in applications where conditional anonymity is a design issue.

REFERENCES

[1] David Mazieres and M. Frans Kaashoek. The design, Implementation and Operation of an Email Pseudonym Server. In Proceedings of the 5th ACM conference on Computer and communications security, p.27-36, November 02-05, 1998, San Francisco, California, United States.

[2] P. Syverson, M.Reed and D. Goldschlag. Onion routing access configurations. In DARPA Information Survivability and Exposition (DISCEX 200), volume 1, p.34-40. IEEE CS Press, 2000.

[3] P. Syverson, G. Tsudik, M. Reed and C. Landwehr. Towards an Analysis of Onion Routing Security. In H. Federrath, editor, Designing Privacy Enhancing Technologies: Workshop on Design Issue in Anonymity and Unobservability, p.96-114. Springer-Verlag, LNCS 2009, July 2000.

[4] M. Reed, P. Syverson and D. Goldschlag. Anonymous connections and onion routing. IEEE Journal on Selected Areas in Communications, 16(4): 482-494, May 1998.

[5] U. Moller, L. Cottrel, P. Palfrader and L. Sassaman. Mixmaster Protocol - Version 2. Draft, July 2003, *http://www.abditum.com/mixmaster-spec.txt.*

[6] B. Levine, M. Reiter, C. Wang and M. Wright. Timing analysis in low-latency mix-based systems. In A. Juels, editor, Financial Cryptography. Springer-Verlag, LNCS, 2004.

[7] C. Gulcu and G. Tsudik. Mixing E-mail with Babel. In Network and Distributed Security Symposium(NDSS 96), P.2-16. IEEE, February 1996.

[8] G.Danezis, R.Dingledine and N. Mathewson. Mixminion: Design of a type-3 anonymous remailer protocol. In 2003 IEEE Symposium on Security and Privacy, p.2-15. IEEE CS, May 2003.

[9] J. Helsingius. anon.penet.fi press release. *http://www.penet.fi/press-english.htm.l*

[10] Cottrel. Mixmaster and remailer attacks.
http://www.obscura.com/~loki/remailer/remailer-essay.html.

[11] Jan Camenisch, Els Van Herreweghen: Design and Implementation of the Idemix Anonymous Credential System. Research Report RZ 3419, IBM Research Division, June 2002. Also appeared in ACM Computer and Communication Security 2002

[12] Els van Herreweghen, Unidentifiability and Accountability in Electronic Transactions. PhD Thesis, KULeuven, 2004.