LENS: LEveraging social Networking and trust to prevent Spam transmission

Sufian Hameed
University of Göttingen
first.last@cs.uni-goettingen.de

Pan Hui
Deutsche Telekom Labs
pan.hui@telekom.de

Xiaoming Fu
University of Göttingen
fu@cs.uni-goettingen.de

Nishanth Sastry
University of Cambridge
nishanth.sastry@cl.cam.ac.uk

Abstract

We introduce *LENS*, a novel spam protection system which leverages the social network of the recipient. In LENS, previous correspondents with a recipient are allowed to send emails directly. To enable new senders to send emails, each recipient independently picks a small number of Gatekeeper (GK) nodes dispersed in the social network. Each GK is authorized to generate voucher, and new senders are required to obtain a voucher (to communicate with the recipient) from a GK in their social neighbourhood. Recipients recover from compromised GKs simply by selecting replacements and revoking vouchers. Using extensive evaluations, we show that *LENS* provides each recipient reliable email delivery from a large fraction (upto 55% of entire userbase) of the social network. *LENS* also proved to be effective in accepting all the legitimate inbound emails from the real email traces. *LENS* imposes zero overhead for the common case of frequent and familiar senders, and remains lightweight for the general case: Our prototype implementation of *LENS* in Postfix/MailAvenger show that *LENS* consumes up to 75% less CPU and 9% less memory as traditional solutions like SpamAssassin.

Technical Report No. IFI-TB-2011-02, ISSN 1611-1044 – February 16, 2011

Contents

1 INTRODUCTION

It is just as easy to send an email to a stranger as it is to send to someone directly related. This has led to vast amounts of abuse. Currently, spam emails have largely outnumbered legitimate ones, increasing from 65% [9] in 2005 to 89.1% (262 billion spam messages daily) in 2010 [3], with a projected cost of $338 billion by 2013 [8].

There have been innumerable attempts to solve the problem of spam, including, recently, solutions that exploit trust embedded in social networks to create solutions without false positives [18, 26]. RE: [18] introduced the idea that recipients can trust senders in their immediate social neighbourhood, and gave a zero false-positive mechanism for vetting emails sent by their immediate social circle (direct friends and friends of friends). However, email coming outside this circle still had to be tested by noisy and unreliable spam filters. In contrast, Ostra [26] introduced a careful system of credits that allowed anyone to send email to anyone else, as long as their balance of credits allowed them to get a token. Unfortunately, for Ostra to be successful, the entire network would need to adopt the system.

In this work, we aim to create a system that, like RE:, can be deployed individually by small groups of users, but allows for a reach greater than friends of friends (FoF). Our threat model assumes that links on the social network can be secured using Sybil defenses like SybilGuard [36] and SybilLimit [35]. However, such defenses are still vulnerable to individual, well-connected nodes on the network going rogue or becoming compromised. Thus, the goal of our system is to make it difficult for a single attacker to gain the ability to spam a large fraction of the network, working in conjunction with standard Sybil defenses.

In order to accomplish this, we create a per-recipient ego-centric view of the entire social network of email users. Anyone who is a friend or FoF can be vouched for using mechanisms that are in principle similar to RE:, and can email the recipient directly. To enable legitimate senders who are farther away, the recipient enlists a set of Gate Keepers (GKs) at various hop counts away from himself. Each GK is allowed to vouch for new senders in his immediate social circle by issuing them unforgeable vouchers. Recipients can also optionally rotate their set of chosen GKs periodically.

Although the above solution is simple and straightforward, there are several advantages to this approach. First, social networks are known to have small dominating sets [11, 16]. Thus, with only a modest number of GKs, each recipient is able to cover a large part of the legitimate social network, allowing most people to reach it. Second, a compromised or rogue node is able to help spam only those recipients who have chosen it as GK. Since the process of GK selection is performed independently by each recipient, the breakdown of individual GKs does not compromise the security of most recipients. The affected recipients can curtail their damage by rotating their GKs. Finally, the protocol requires a first-time sender's mail server (MS) to present a valid token *before* sending the payload. This enables a recipient to quickly terminate an invalid connection, thereby saving valuable resources both on the recipient's mail server and also in the network. This feature becomes especially important when dealing with spams with large payloads such as viruses or other attachments.

We evaluate *LENS* extensively using three large social networks (Flickr with 1.7 million users, Facebook with 3.1 million users and Google Buzz with 2.1 million users) and two traces of email transactions (Uni-Kiel, 57.1 K users and Enron, 52.7 K users). *LENS* is able to receive *all* incoming emails in the email traces with an average of just 31 (0.06% of users) GKs per recipient in the Enron

Technical Report No. IFI-TB-2011-02, ISSN 1611-1044 – February 16, 2011

dataset and 10 GKs (0.017% of users) in the Uni-Kiel dataset. On the social network traces, the worst performance is on Google Buzz where a recipient has to choose 478 GKs on average in order to enable 12.5% of the social network to send emails to him. Where as for Facebook with an average of 871 GKs a recipient can be reach by 55% of the social network. In Flikr, with an average of just 400 GKs a recipient is able to receive reliable email from 54% of the total userbase. For the social network (Buzz) and email traces around 80% to 92% of the selected GKs have less than 300 recipients. Thus, If an attacker compromise a GK, the affected number of recipients remains limited.

We have implemented the solution in Postfix/MailAvenger. Our stress tests and micro benchmarks show that the overheads imposed by the additional processing are tolerably small. *LENS* is significantly less compute intensive (Up to 75% less CPU and 9% less memory) than current solutions like SpamAssassin.

The rest of the paper is organized as follows. § 3 describes the design of *LENS*. In § 4 we discuss how *LENS* is realized and incorporated with existing email processing system. In § 5 we demonstrate scalability, effectiveness and complexity of *LENS* using online social network (OSN)/email datasets and system evaluations. In § 6 we discuss some design decisions and questions related to usage of *LENS*. § 2 describes the state of the art, and finally we conclude the paper in § 7.

2 RELATED WORK

In recent years, several techniques [12, 14, 18–20, 23, 25, 26, 31] have been proposed using social networks and trust and reputation systems to fight against spam. Boykin *et al.* create a social network of friends in the cyberspace based on the emails exchanged between them [12]. With the usage of local clustering properties of social network the emails they are able to classify 53% of all the emails as spam or non-spam with 100% accuracy. However, the method is limited to offline analysis, and even the remaining 47% emails are left for other filtering techniques.

Ostra [26] tries to explore the use of trust relationship to thwart unwanted communication. They used the number of trust relationships the user has to limit the amount of unwanted communication a user can produce. Their system relies on existing trust networks to connect senders and receivers via chains of pair-wise trust relationship and use a pair-wise, link-based credit scheme to impose a cost on originator of unwanted communication. Scalability of this system is still not certain if it maintain a per link credit scheme. Although it can be decentralized by introducing a central tracker component, it is not clear how scalable the system will be - they have not evaluated this part.

Re: Reliable Email [18] talks about use of white list of friends and automatic white list of FoF to increase the communication chance of only white list friends. By using this protocol, RE can accept almost 85% of received emails and prevent up to 88% false positive by the existing spam filters. However, emails other then friends and FoF still had to be filtered by noisy and unreliable spam filters. *LENS*, on the other hand extends the reliable delivery of emails beyond FoF with the help of legitimate GKs.

SOAP [24] presents a social networked based personalized spam filter that integrates social closeness, user (dis)interest and adaptive trust management into the Bayesian filter. SOAP exhibit some problems which will limit its usage. These problems mainly includes intrinsic cost of initialization and continuous adaptation of social closeness (between sender and recipient) and social

interests (of an individual) in Bayesian filter.

SocialFilter [29] propose a collaborative spam mitigation system that uses social trust embedded in OSN to asses the trustworthiness of Spam reporter. The spammer reports from the SocialFilter nodes are stored at a centralized repository that computes the trust values of the reports and identify the spammers based on the IPs. However, effectiveness of SocialFilter is doubtful with spammers using dynamic IPs.

In *Trust and Reputation Systems*, network users try to calculate the reliability and trustworthiness of other users based on their own experiences and that of others. Boykin et al. [12] proposed an automatic email ranking system based on trust and reputation algorithms. *MailRank* [14] is a spam detection system based on trust and reputation scheme to classify email addresses (apart from ranking emails as done in [12]) into spammer addresses and non-spammer addresses. It additionally determines the relative rank of an email address with respect to other email addresses. SNARE [19] infer the reputation of an email sender based solely on network-level features, without looking at the contents of a message. Using an automated reputation engine, SNARE classifies email senders as spammers or legitimate with about a 70% detection rate for less than a 0.3% false positive rate. Lack of authentication and non-repudiation in standard trust and reputation solution make them subject of identity spoofing, false accusation and collusion attacks. Further, these solutions consume extra valuable resources of email servers on email reception and filtering. *LENS* on the other hand can reject unwanted email traffic during the SMTP time.

3 *LENS* ARCHITECTURE

Malicious users (bad actors) are responsible for sending spams. Existing spam prevention solutions mainly focus on detection of spam and the bad actors. Every now and then new solutions are proposed to detect spam and the spammer make improvisations. This results in a continuous battle and we are stuck with a permanent arms race. *LENS* mainly focus on selection of legitimate users (community member in close social circle of a user or socially distant trusted users called GKs) in legitimate MS (good actor) .

LENS (Figure 1) comprise of four main components; 1) community formation 2) trust management 3) GK selection and 4) spam report handler. All the components of *LENS* run on a Mail Server (MS) along with the Mail Transfer Agent (MTA) and SMTP server.

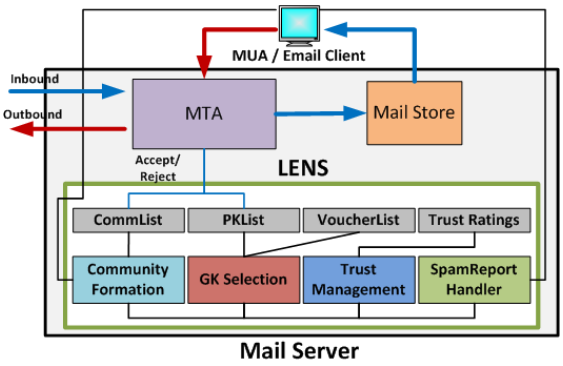


Figure 1: *LENS* Architecture

Technical Report No. IFI-TB-2011-02, ISSN 1611-1044 – February 16, 2011

3.1 MS's Responsibilities

The MS is responsible for executing the *LENS* protocol on behalf of the email users. Each email user can explicitly control his community (friends and FoF) and can give feedback by reporting spam emails. All the remaining functionality of *LENS* is handled transparently by the MS.

All the *LENS* enabled MSs are assumed to be legitimate with a valid certificate issued from a *Trusted Authority*. These certificates are used during server authentication (§ 3.4.4) to prove that the MS is legitimate. All the authentication requests associated with invalid certificates are ignored.

When a MS is certified (just like any web-server) by a *trusted third party* it shows that the MS belongs to a legitimate owner. The owner who controls the MS becomes visible (its easy to take legal actions against the owner or blacklist the MS). Investing in infrastructure or becoming visible has never been a choice of the spammer.

Addition of email users is strictly moderated in companies, private institutes and universities. Furthermore, all the major web-mail providers run bot detectors against non-human automatic account creation and impose an email sending limit (100 to 1000 recipients per day) [1]. However, illegitimate users can create accounts on web-mail providers like gmail, yahoo, hotmail and gmx etc. *LENS* address this problem by maintaining a trust rating (TR) for each user to ensure their legitimacy and differentiate between legitimate and illegitimate ones.

3.2 Community Formation

In *LENS*, the social components of a community consist of two levels, namely friends of the users and his friends-of-friends (FoF). Adding a friend roughly corresponds to the notion that "User A trusts his friend B not to send him spam and vice versa". Triadic closure is supported by adding FoFs into the community.

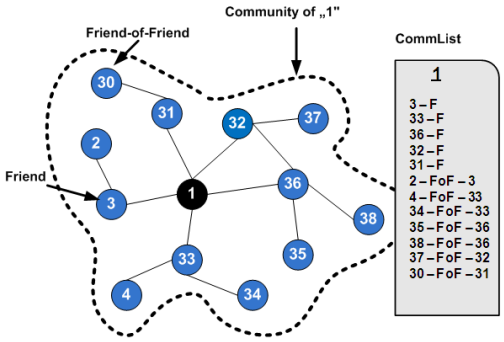


Figure 2: Community structure (friends, FoFs) of user "1"

Figure 2 depicts the community structure for a user. Users can receive *all the messages from his community directly into his inbox*. The formation of a social community is a simple two step process

1. **Adding Friends:** A user can request anyone in the email network for friendship. Once two

Technical Report No. IFI-TB-2011-02, ISSN 1611-1044 – February 16, 2011

users add each other as friends, an entry is made in CommList [1] with the user ID and label "F". CommList can only be accessed by the user himself or by his MS.

2. **Adding FoF:** The idea of FoF addition is that there will be no exchange of friend lists among the friends. Instead a user can suggest two of his friends to add each other into their communities as FoF. Once two nodes add each other as FoF, an entry is made in their CommList with the ID of the added node, label "FoF" and the ID of the referring mutual friend.

By design, community formation is a selective process and involves certain human involvement to prevent any unnecessary addition in communities and preserve high level of privacy. We have developed a community formation client in PyQT to handle the process of community formation.

Figure 3 show the cross platform and email client independent community formation client we have developed in PyQT. The community formation client provides the functionality to add friends, view community, view pending requests, accept/reject the request received and send suggestions to mutual friends.

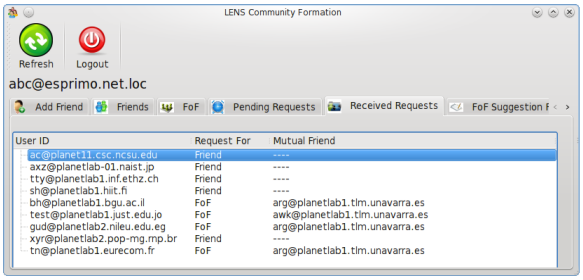


Figure 3: Cross platform community formation client

3.3 Trust Management

In order to receive spam-free emails beyond a recipient's community, *LENS* select legitimate users (GKs), outside the recipient's community within pre-defined social distances. Once selected, the inherited trust of the GKs is used to vouch for legitimate users outside the community of the recipient for communication. However, illegitimate users can create arbitrary number of accounts on web-mail providers (like gmail, yahoo, hotmail or gmx etc) and exploit them in becoming GKs for spamming . In order to ensure that illegitimate users are not selected as GKs, *LENS* maintains trust rating (*TR*) for each email user on the MS. The main goal of *TR* is to assign a rating to each user known to the MS and to use these ratings to decide whether the user is legitimate or not.

Based on the MS's *TR* there are four different user types (UT) i.e. *legitimate (LU), trusted (TU), new (NU)* and *illegitimate (IU)*. *LU* are the non-spammers with clean track of spam-free communication and $TR >= L$(trust rating threshold of legitimate user). *TU* as the name suggests are the users trusted by the MS to be legitimate/non-spammers. They have $TR >= L$ and the MS have verified their Identity uniqueness. *NU* are new addition of users to the MS with $0 <= TR < L$.

[1]CommList is maintained for every user and it contains entries of community nodes, either as friend or FoF (see Figure 2)

IU are the identified spammers with negative *TR*. Assignment of *TR* falls in two main categories as follows.

3.3.1 Direct *TR*

Direct *TR* is the manual assignment of *TR* to a user by the admin(s) of the MS. Direct *TR* have priority over other methods of *TR* and it overrides the existing values of *TR*. Direct *TR* is more practical for small-medium size, strictly moderated MSs of companies, private institutes and universities, where the admins have direct trust on the users (for example users like Uni professor or company's GM are trusted not to be spammer). For big web-mail providers and ISPs the *TR* of each user should be computed automatically.

3.3.2 Automated *TR*

Automated *TR* is based on user voting. Analysis of spamming nodes show that they exhibit one way communication i.e they are always on the sending end [12, 14]. If A sends email to B, then it can be regarded as trusting B, or voting for B [14]. Trust Management component of *LENS* listens to the FROM field on the received email and computes *TR* according to the algorithm summarized in Algorithm 1. *TR* is maintained by the MS for each of his user.

Algorithm 1: Automated TR
Initialization

```
# Domain = D, UserType = UT, TrustRating = TR
1: # infer votes from existing data
2: For Each inbound email from sender i to receiver j
3:   if (email == Legitimate)
4:      if (Di != Dj)
5:         if (j has no votes from Di)
6:             TRj +=1
7:
8:   if (TRj == L) # reach threshold of Legitimate User
9:       Tj = Legitimate
10:# indentity uniqueness test independent of voting
11: For Each  j with (Tj == Legitimate)
12:     status = verify_identity_uniqueness(j)
13:     if (status == True)
14:         Tj = Trusted
```

Expansion

```
# Processing new email after processing existing data
1: For Each inbound email from sender i to receiver j
2:    run step 3 to 14 of Initialization block
```

After becoming *LU*, a user has to pass the verification of identity uniqueness inorder to become *TU*. During the verification the user is verified and bind with his unique identity, like mobile number, using challenge response authentication (Facebook also verifies the uniqueness of user account by binding him with a unique mobile number). We have discuss a potential attack on trust ranking in § 4.3 along with the means to neutralize it.

3.4 GK Selection

GK serve as a means to vouch for legitimate users outside the community of the recipient for communication. Any emails outside the recipient's community can reach the inbox only if his

selected GKs vouch for them. To maintain a reliable trust structure, a GK is only authorized to vouch for the nodes in his own community.

Selection of GK is performed transparently from the users, by the *GK Selection* component of the MS running *LENS*. On successful completion of the selection procedure, both the recipient and his GK receive their respective keys. GK's key is used (by GK's MS) to issue vouchers, to the GK's community members, to communicate with the recipient. On the receiving end, the recipient's key is used to verify the vouchers. In order to keep *LENS* effective and scalable, the goal of GK selection process is to select **minimum GK** for **maximum coverage**

We do not use the classical dominating set or distributed dominating set approximation to select the GKs for two main reasons. First, we do not want a common set of nodes to serve as GKs for the whole population. The reason is that these common GKs will have too much information about everyone in the network and would become privacy and security weak points. The second reason is that by further considering the communication patterns, we do not expect everyone on the planet to communicate randomly with each other. The probability actually decreases with an increase in the social distance. Instead of working on a global provisioning of optimal GK for the entire email network, we will discuss and present a scalable approximation.

One of the design constraints of *LENS* is that *a user cannot obtain information about the global properties of the social network*. The best approach here would be to restrict a user with only its personal community information. The GK selection procedure of *LENS* consists of the following three stages.

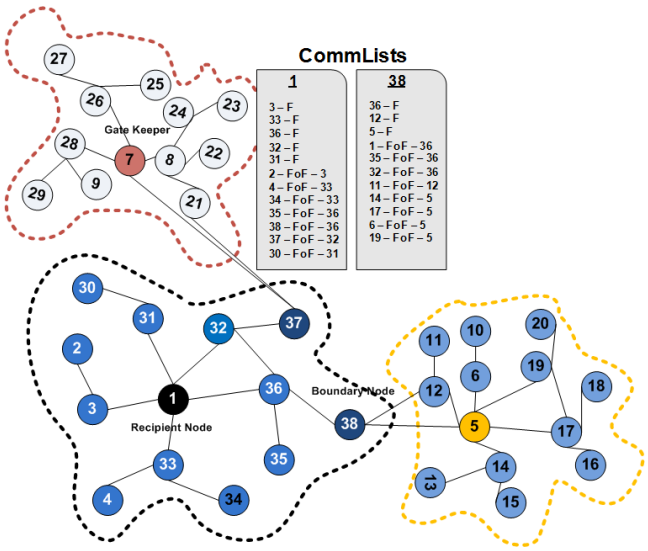


Figure 4: Community structure and GK coverage

3.4.1 Stage 1: GK selection in adjacent communities

For any given recipient node (RN), the GK selection process starts from the adjacent communities of the recipient. The process is described below.

1. **Request:** The MS of the RN will use the RN's FoF (boundary nodes(BNs)) to find the

Technical Report No. IFI-TB-2011-02, ISSN 1611-1044 – February 16, 2011

locally optimal GKs (Figure 4). RN's MS will simply request all BNs of the community to send their suggestion for good GKs.

2. **Suggestion:** The MS of the BN will suggest a user from BN's friends, having largest number of friends (outside the RN's community), to the RN as a GK. The MS of the BN will also inform the BN's friends about the recipient. Exchange of friend list is not required for making suggestions, the MS can use the information in the BN's CommList [2]. For instance node "38" will suggest "5" as GK to "1" instead of "12", since "5" is the friend, outside the community of "1", with maximum nodes referred as FoF. Figure 4 depicts the selection of GKs by two BNs of the recipient.

3. **Verification of Legitimacy:** This is the last and most important step of GK selection process. This step ensures that the GK is legitimate (details of this is covered in § 3.4.4). As a result of this step, a RSA based public key (PK) and secret key (SK) is generated for the GK. PK is shared with the RN and the SK is use to issue vouchers to entire community members of the GK. These members will use the issued vouchers if they need to communicate with the RN (see Figure 5). All the users within a social radius (level or hops) of 5 would be able to send emails to the recipient with an assurance of being free from spam. According to the small world property of social network, any two users can be connected with a small number of hops (six-degree of separations [30]). This suggests that if the email network exhibits a social network behavior, the recipient node would be highly reachable throughout. Distant users having a social distance greater than 5 are covered in stage 2 of the GK selection process. PKList [3] and VoucherList [4] are maintained at each node to store the PKs and vouchers. To communicate with each recipient outside of its social circle, a node would have to maintain a voucher for that recipient. Similarly, a GK issues a single voucher for all the recipient outside of his social circle for whom he is vouching.

3.4.2 Stage 2: GK selection beyond adjacent communities

In order to provide reachability to other distant users in the email network, the GK selection procedure of *LENS* can be easily extended to select GK in distant (beyond adjacent) communities. The process is quite simple (see Figure 6). After the selection of GKs in the adjacent communities (stage1), RN's MS will ***send a request to the selected GK's MSs*** to help them look for GKs from their adjacent communities. As a result of this request, the GKs will use their BN to find new locally optimal GKs and ***send their suggestions back to the RN***. Finally, ***the RN's MS will verify legitimacy of the new set of GKs from social level 6*** and extend reachability of the RN to level 8. Using the same procedure, the extension of GK selection is possible at any further levels. Of

[2]CommList is maintained for every user and it contains entries of community nodes, either as friend or FoF (see Figure 4)

[3]PKList is maintained for GKs selected by the recipients. Any single entry in PKList contains PK, GKID and RecipientID.

[4]Once the GK has the SK, a voucher $(((User_{ID})_{hash})_{Sign-SK})$ is issued to all the users of the GK's community. These vouchers are added to the VoucherList of the users, along with the recipient's and GK's IDs $(< RN_{ID1}, RN_{ID2}, ... >, GK_{ID}, Voucher[((User_{ID})_{hash})_{Sign-SK}])$,to use later for communicating with the recipient. A single voucher issued by the GK, to each of his community user, will work for all the recipients

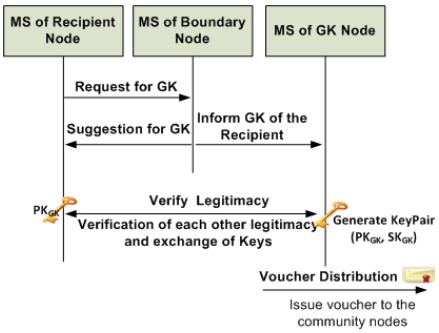


Figure 5: GK verification and voucher distribution

course, all these extensions do not come for free, it is solely dependent on the design choice of the *LENS* users. The GK selection for higher levels must also consider the small world property of social networks [30], in order to *avoid random walks* on the social graph.

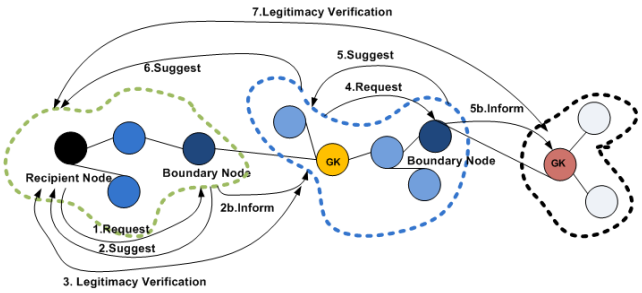


Figure 6: GK selection beyond adjacent communities

3.4.3 Stage 3: GK selection for new communication

One of the reasons that email is so great is because anyone can contact anyone else. People get legitimate emails from new people everyday. *LENS* provides spam free email communication to distant and new users as follows. Instead of extending GK selection to the entire network, *LENS* restricts it only to the social levels covered in stage 1 and 2. If a user wants to send an email to a recipient (for the first time), who is not only outside its community but there is also no GK for the recipient within its community, *LENS* will perform the following two steps.

1. **Announcement:** announce the sender to the RN that wants to communicate and start the legitimacy verification process.

2. **ii) Verification of Legitimacy:** start the legitimacy verification process (see § 3.4.4 for details) to prove that the sender is and not a spammer. As a result of this process, the RN will add the sender as his GK. The sender will further issue vouchers to his entire community nodes and they will be able to use these vouchers as well to communicate with the RN.

This process is only performed once and for all at the start of a new communication. After the sender is verified as a GK, not only the sender but his entire community can send email to the

recipient. Therefore, instead of having a GK for the entire network in *LENS* we can select GK on the fly after stage 2, if communication is desirable. The results in § 5.2 show that the need for stage 3 GK selection is quite infrequent. Further, the results in § 5.3 shows that stage 3 only adds a negligible delay of 0.5 to 1.5 secs in email transmission.

RE: received 85% of the email correctly utilizing just its social network (friends and FoF) [18]. With the GKs, *LENS* enhances its reliable and spam free delivery of emails beyond social network. Success of *LENS* depends on the successful formation of social communities and continuous extension of GK selection between the email users. Users having a larger social community would be benefiting more from *LENS* than the isolated and less socially connected users.

3.4.4 Legitimacy verification in GK Selection

Legitimacy verification is one of the most significant part of GK selection process. This protocol ensures that the recipient can trust the GK to be legitimate. With this protocol a RSA based PK and SK is generated for the GK. PK is shared with the recipient and the GK uses the SK to issue vouchers to his community members as a vouching mechanism to send emails to the recipient.

Based on the trust rating for details on trust management) there are four different user types (UT) i.e. *legitimate (LU), trusted (TU), new (NU)* and *illegitimate (IU)*. Here we would like to emphasize that the main focus of *LENS* is spam protection, so only verifying the legitimacy of a user is enough to counter spam, rather than running costly protocols to authenticate the identity of each user. We propose two variations of the protocol based on the difference of the locations of the recipient and his GK.

i) Recipient and GK on different MS

If the recipient and GK belong to different MSs, legitimacy is verfied in two steps. First, we perform ***server authentication*** to verify that both recipient's and GK's servers are legitimate and untempered. Second, the TR of the user at his MS is verified to ensure that the user is not illegitimate/spammer. Following is the description of our legitmacy verification protocol. In *LENS*, by design, GK's MS will always initiate the protocol

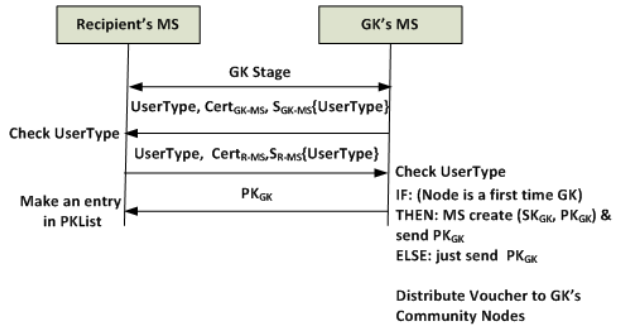


Figure 7: Legitimacy verification in GK Selection

- Recipient's MS and his GK's MS agrees on the stage of GK selection.
- GK's MS picks the UserType (UT) of the GK (based on the TR), signs the UT using his private signature key (S_{R-MS}) and sends the UT [5], signature and his certificate ($Cert_{GK-MS}$), issued by

[5]By design GK has no control in manipulating the MS to send false UT. There exists two possibilities where an

the trusted authority, to the recipient's MS.
• Recipient's MS verifies the signature of the GK's MS. For GK selection of stage 1 & 2 the UT of the GK must be only *legitimate or trusted*, else the verification will fail. In stage 3 the UT of the GK can also be *new* (relaxation of UT *new* is only given in stage 3, with email sending limit).
• Recipient's MS picks the UserType (UT) of the recipient (based on the TR), signs the UT using his private signature key (S_{R-MS}) and sends the UT, signature and his certificate ($Cert_{R-MS}$), issued by the trusted authority, to the GK's MS.
• GK's MS verifies the signature of the recipient's MS. The UT of the recipient can be *new, legitimate or trusted* . Only limitation is that the UT of the recipient must not be *illegitimate*.
• If the GK is selected for the first time the GK's MS will generate RSA based key-pair (PK, SK) for the GK and send the PK to the recipient's MS. If the GK was already selected before for some other recipient, key-pair (PK, SK) will already exist for the GK and the MS will only send the PK to the recipient's MS.
• Once the GK has the SK, GK's MS will use the SK to issue a voucher ($((User_{ID})_{hash})_{Sign-SK}$) to all the users of the GK's community. These vouchers are added to the VoucherList of the users, along with the recipient's and GK's IDs ($< Recipient_{ID}, ... >, GK_{ID}, Voucher[((User_{ID})_{hash})_{Sign-SK}]$), to use later for communicating with the recipient. If a user is selected as GK for multiple recipients, his community users will not need multiple vouchers for each recipient. One voucher (from the GK) is enough to communicate with all the recipients. Only the recipient's ID is appended to the existing voucher entry in the VoucherList ($< Recipient_{ID1}, Recipient_{ID2}, ... > , GK_{ID}, Voucher[((User_{ID})_{hash})_{Sign-SK}]$).
• On receiving the PK of the GK, the Recipient's MS will add new entry (PK, GKID) to the PKList of the recipient. In the GK selection stage 3, if the UT of the GK is *new*, the recipient's MS will enforce a rate limit on the number of emails per day from the GK and his community user. Later, if UT of the GK will upgrade to *legitimate or trusted*, the recipient's MS will remove email sending limit with the verification of updated UT. This email limit is applied to limit the number of spams sent by a user before being identified as malicious. On the other hand, it will allow new honest users to send email outside their community.

In *LENS*, usage of RSA based key-pair (PK, SK) have many advantages over using using symmetric keys between the GKs and their recipients. A GK requires a single key-pair, irrespective to the number of recipient he has. Based on the evaluation is § 5.2, a recipient on average requires only 10 to 31 GKs i.e. a recipient has to maintain only 10 to 31 PKs on average for his selected GKs. Moreover, the GK does not have to re-issue voucher to his community members for every single recipient. A single voucher will work for all the recipients the GKs is authorized to vouch for. In *LENS*, the PKs are distributed directly by the MS of the GK. Hence, *LENS* does not requires a Public Key Infrastructure (PKI) for key management, distribution and verification and by doing so it avoids considerable amount of overhead.

ii) Recipient and GK on same MS

illegitimate GK can send false UT to the recipient. First, a malicious GK host a private MS certified by a trusted authority and configure it to always send *legitimate* for his UT. However, by doing so the malicious GK also becomes visible and this has never been an option for spammers as they become vulnerable to legal actions. Second, the MS is compromised by the malicious GK. This is synonymous to hacking the MS and the corresponding defense mechanisms are beyond the scope of this paper.

If the recipient and his GK are hosted by the same MS, the protocol works exactly the same without any need for *server authentication* (since all the messages are exchanged internally).

3.5 Spam Reports

Spam reports are handled by *Spam report handler*. In *LENS*, by design, only reports from *trusted users* are weighted. This design decision is to prevent spammers from falsely reporting non-spammers as spammers. When a user will receive a spam, he will report to his MS that the sender is a spammer. Upon receiving a report of spamming, the *spam report handler* will register the report against the *TR* of the reporter. Once the handler will receive RT (report threshold) reports from distinct trusted users, it will assign a negative *TR* to the sender of spam. Furthermore, if the spammer is not a local user on the MS, the handler will add the offending node to the revocation list, preventing further spamming. Handling of spam report is summarized in Algorithm 2.

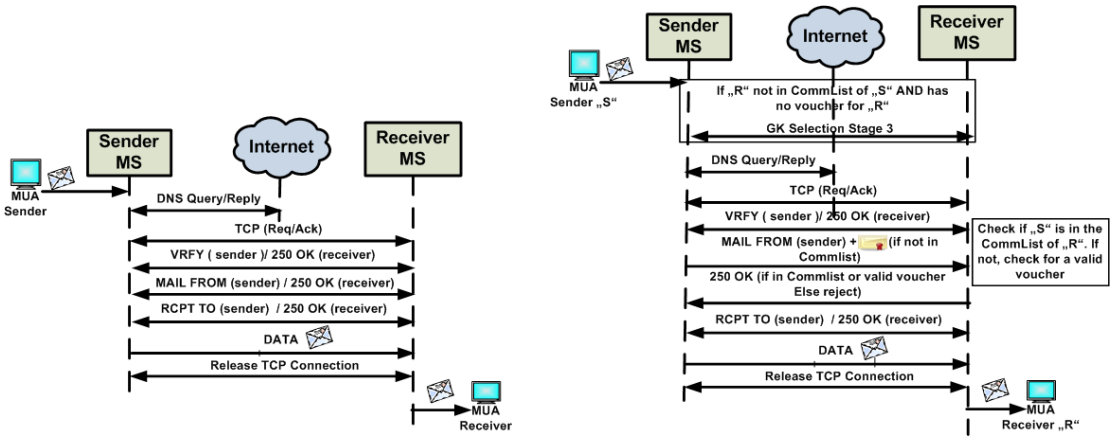
Algorithm 2: Spam Report Handling

```
# Domain = D, UserType = UT, TrustRating = TR
# SRij = # of SpamReports by user i of user j
# Ij = # unique reports against j
# RT = threshold
ForEach SpamReport of Spammer s by Reporter r

   if(UTr == trusted && SRrj ==0)
      Is++
   if (Is >= RT)
      if (s = localuser)
         TRs = negative
         UTs = illegitimate

      else:
         Add s to Revocation list as IU
```

After the malicious users (local or remote) are identified and marked as *illegitimate*, their associated entries in PKList and VoucherList are also terminated from the MS. This means that if a GK is malicious, his associated entries from the PKList of the recipients and VoucherList of the community members will also be removed with him.

4 EMAIL PROCESSING with *LENS*

In this section we discuss the processing of emails with *LENS* and how spam can be prevented from transmission. We conclude this section with the discussion on possible attacks and prototype implementation of *LENS*. Figure 8(a) depicts the flow of an email i.e. from message creation, transport to delivery. Mail user agent (MUA), the sender's email client, submits the email to its Mail Server (MS) using SMTP. The sender's MS will look up the destination's mail exchanger record (MX) in the DNS server. The DNS server finds the highest preference MS for the recipient and reports the name of the MS by returning a MX resource record. After this point, a TCP connection is established between the sender's and the receiver's MSs and the sender's MS sends the VRFY,MAIL FROM and RCPT TO commands (one by one) to the receiver. With successful acknowledgment from the receiver side, the complete email (header and the body) is sent, and the TCP connection is released. The mail delivery agent (MDA) delivers the accepted email to a server for local mail delivery. Once delivered to the local MS, the mail is stored for batch retrieval by authenticated mail clients (MUAs) using IMAP or POP.



(a) without *LENS* (b) with *LENS*

Figure 8: Email processing model with *LENS*

4.1 *LENS* Based Email Processing

The *LENS* based email processing works transparently from the user, like most spam filters. Now let us assume that we have the social communities defined and the GKs selected. For each user the Commlist, PKList and VoucherList are kept at the local MS. There are two types of messages sent and received i.e. messages within the social community and those outside. The processing of both the messages is explained as follows (also see Figure 8(b)).

1. **Message within the community:** When a message is sent to any recipient within the community, the recipient's MS will verify the sender against the recipient's Commlist and place the message in the mailbox.

2. **Message outside the community with GK:** If a message is sent to a recipient outside the sender's community, the sender's MS will bind a voucher, issued by a authorized GK along with the message. On reception, the MS will verify the voucher using the PK stored in PKList against the GKID and place the message in the recipient's mailbox.

3. **New Message outside the community without GK:** If a new message is intended for a recipient outside the sender's community and with no voucher issued by any GK. The sender's MS will hold the message and start a GK selection procedure (stage 3). The sender will be announced as a potential GK for his community and the GK verification procedure will be carried out with the recipient as discussed in § 3.4.4. On successful completion, the sender will be selected as a GK for the recipient. The sender's MS will now bind a voucher with the withheld message and send it out. When the message arrives at the recipient's MS, the MS will verify the voucher using the PK stored in PKList against the GKID and place the message in the recipient's mailbox.

4.2 Prevention of Spam Transmission

One of the main contributions of *LENS* is that it prevents the transmission of spam across the network. Let us consider that the sender's and receiver's MS have already established a TCP

connection. Now, when the sender's MS sends the RCPT TO command (RCPT TO:<forward-path>[SP <rcpt-parameters>] <CRLF>), it also appends the voucher and issuing GK's ID (for e.g RCPT TO:<example@abc.com>Voucher = 1f2a91aa236d0012 GK = gk@example.com) as additional rcpt-parameters, to communicate with the recipient, if the recipient is not in the sender's community.

According to the current draft standard of SMTP [22] using additional rcpt-parameters is optional and contemporary SMTP implementations MUST support it as basic extension mechanisms. The SMTP server not obliged to understand the additional rcpt-parameters simply ignores them. At the recipient's end, the MS verifies if the sender is a community member or has a valid voucher from a authorized GK. Failure of the verification results in the termination of the TCP connection by the recipient and the transmission of email (header and body) will not take place, thereby preventing the spam message from being transmitted.

4.3 *LENS* Under Attack

Compromised user: If a user (who may also be a GK) is compromised (someone stole the identity), it will only have a local effect within the community. The effect is temporary and only lasts until the victimized user broadcasts the incident using his other ids (may be through friends or word of mouth) or claims back his ownership from the email service provider. Let us suppose that the victimized user is unable to reclaim the ownership of its Id. In that case, the user can always request its community to abandon his compromised Id and the MS will remove all the data associated with the compromised Id from the PKList and VoucherList. Hence, the attacker would not be able to harm the system on a large scale as the SKs and vouchers are handled internally by the MS.

Trust Farming: A Spammer can not increase his own TR (please refer to § 3.3 for details on trust management) by issuing votes towards others i.e. by sending emails. However, Spammer could launch a trust farming attack on *LENS*, where a spammer(s) vote for another spammer(s) to increase his rank. This attack is comparable to link farming on the Internet to attack the PageRank. However, extensive amount of work has already been done to identify and neutralize such attacks on social reputation schemes. Solutions like [34] can be used orthogonally in *LENS* for protection against trust farming.

Human spamming using GK selection stage 3: Stage 3 of GK Selection protocol allows new users to send emails to complete strangers. A new user who is also a spammer can exploit stage 3 to get places as a *new* GK and spam the recipient. However, due to rate limit on the emails from *new (UT)* GK, the scale of the attack will remain low. Further, creation of arbitrary new IDs (if spammer doesnot care about being reported and donot reuse old ID ever) and repetition of stage 3 will require substantial human involvement. This will incur cost which is against the spamming model.

Voucher misuse: The vouchers issued by the GK are bind with the identities (userid+domain for e.g. abc@example.com) of the community members and can only be used by the user issued to. Current version of *LENS* doesnot impose any expiration limit on the vouchers. Vouchers become invalid or get removed/terminated by the MS based on four conditions; 1) If GK removes any user

from his community the MS will also remove the associated voucher (even if the user gets hold of the voucher, he cannot use the voucher on his own after being removed), 2) The user himself gets out of the GK's community, 3) UT of the user becomes negative and 4) If the Recipient revokes the GK, all the associated vouchers will become become invalid.

Key theft: Protection of keys is the responsibility of the MSs. Attacks related to key thefts are synonymous to hacking the MS and the corresponding defense mechanisms are beyond the scope of this paper.

Table 1 lists some additional threats to *LENS* along with their existing solutions which can be used in conjunction.

Attacks	**Solutions**
Sybil attack	SybilGuard [36] and SybilLimit [35]
Trust farming	Anti-link farm based solution [34]
From forgery	SPF [33]

Table 1: Companion solutions to additional threats

4.4 Forgery of *from* Addresses

In *LENS* we have authentication at the MS level to verify the legitimacy of the MS. Further, addresses are not authenticated in SMTP. Therefore, it will be very easy for the spammer to launch a spam attack with forged ***from*** addresses as if they are from the recipient's community. In order to solve this problem *LENS* utilizes SPF [33] as standard sender authentication techniques to robustly verify that the *from* address in the received email is not forged. SPF filters the inbound email at MAIL FROM: command and is already being used effectively in the existing email system.

4.5 Prototype Implementation

We have developed a *LENS* based prototype implementation for email processing during the SMTP transactions using CommList and Voucher issued from trusted GKs. We have integrated this prototype with the mutt mail client, the Mail Avenger SMTP daemon [4], and the Postfix MTA [6]. Note that we made no modification to SMTP protocol, *LENS* email filter (using CommList and vouchers issued by the GK) run as an independent daemon (like spamd for SpamAssassin [7]) to monitor the SMTP transaction and based on the results take different actions (see Figure 8(b)). Community formation, GK selection and trust management run as independent components on the MS. The prototype uses SHA1 for hash and RSA based signature and verification.

Incremental deployment:

It is easy to integrate *LENS* into current SMTP servers. Inevitably, when deployed, some users will adopt *LENS* before others. Until every user is familiar with *LENS*, it will run complementarily with the existing spam filters. See Figure 9, after sender authentication *LENS* is first to examine inbound email. Incremental deployment will have four communication scenarios as follows.

a) Sender and recipient both have *LENS*: email processing will follow the procedure discussed in § 4.1 (also see Figure 8(b))

b) Only sender has *LENS*: sender will send an email as usual and the MS on the recipient side will process the email normally to its email processing mechanism.

c) Only recipient has *LENS*: after performing a check for existence of community or signature, *LENS* will pass the email to the existing spam filters.

d) Both sender and recipient do not have *LENS*: emails will be process according to the existing mechanisms deployed at the sender's and recipient's MSs.

Based on our evaluations around 81%-89% (§ 5.2) of the emails are from the community members. With the increase in adaptation/deployment of *LENS* the users community will grow and their reachability via the GKs will increase. With global deployment *LENS* is envisioned to stop the transmission of spam at the first place. Even with partial deployment, the sender and recipient deploying *LENS* will benefit in two ways. First, all the emails from the user's community (which is 81%-89% in Enron and Uni-Kiel email samples, see § 5.2) and with valid GK signatures will be processed by *LENS*, which is significantly efficient than SpamAssassin (see § 5.4). Second, less number of emails will pass through existing content-based filters, and avoid the chances of false +ves and -ves due to imprecise content signatures. These benefits will also help motivate the deployment of *LENS*.

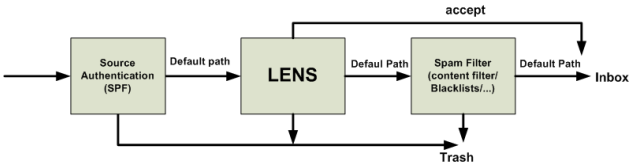


Figure 9: Complementary *LENS* and existing spam filters

5 EVALUATIONS

In this section, we present the experimental evaluation of *LENS*. We are concerned in evaluating four things;

a) Scalability of *LENS*:

In order to verify the **feasibility and scalability** of *LENS*, we use three large scale OSN datasets i.e. Facebook, Google Buzz (Buzz) and Flickr. Data samples of Facebook, Buzz and Flickr are good choices for evaluating *LENS* as they represent real online social connections. For OSN datasets, we are only able to evaluate the GK selection procedure at stage 1 because of the limitation of the dataset size. Although the datasets contain millions of users, the average path lengths are no more than 5 hops. For the results we are mainly interested in following two things.

1. **Number of GKs for receiving messages:** Feasibility and scalability of *LENS* depend on a reasonable number of GKs selected for a particular recipient.

2. **Reachability of recipient via GKs:** The success of *LENS* also depend on how many legitimate users outside the community can possibly reach a recipient with a certain number of GKs. Ideally, we expect a maximum number of legitimate users with a minimum number of GKs.

b) Acceptance all the legitimate inbound emails:

In order to verify the **effectiveness** of *LENS* at accepting all the legitimate inbound emails, we use two real email traces from both a large commercial company and a large academic unit.

c) Performance of GK selection:

To evaluate the network overhead and latency of GK Selection protocol we performed extensive test of GK Selection in stage 1 and 3 on twenty geographically distant nodes in PlanetLab [15].

d) Performance of email processing with *LENS*:

For evaluating the system performance of email processing with *LENS*, we deployed the prototype of *LENS* over the LAN and ran different experiments.

5.1 Scalability Evaluation with OSN Data

Table 2 presents the high-level statistics of Facebook, Flickr and Buzz datasets gathered and used in [21, 27, 32]. Currently, Facebook is the largest social network in the world and the number one photo sharing site on the internet. It is a "pure" social network, in the sense that its primary purpose is finding and connecting users. Our data sample of Facebook consists of 3.1 million users with over 23 million edges and an average of 15.2 friends per user. The Buzz data sample consists of 2.1 million users with over 5.2 edges and 4.5 friends on average for each user. Flickr on the other hand is not a pure social network, intended primarily for publishing, organizing and locating content. Our dataset of Flickr consists of 1.7 million users with over 15 million edges and an average of 18.1 friends per user.

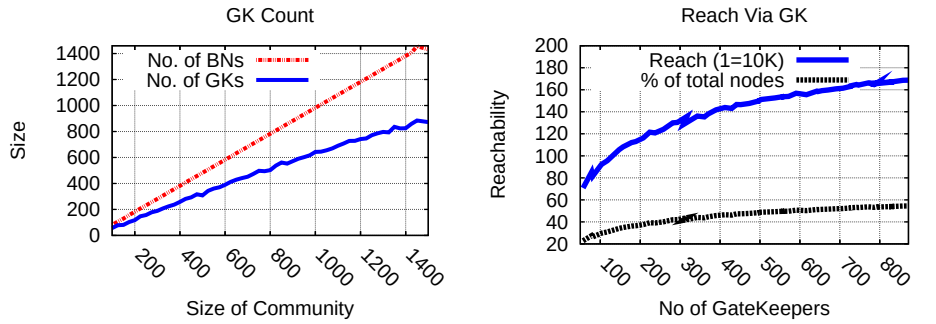
Social network data set	Facebook	Flickr	Buzz
Number of Users	3,097,165	1,715,255	2,123,421
Number of Edges	23,667,394	15,555,041	5,257,684
Average Friends	15.28	18.13	4.59
Clustering Coefficient	0.175	0.313	0.09
Avg Path Length	5.13	5.67	5.39
Average Community Size	1,587.32	4,398.44	189

Table 2: High-level stats of OSN datasets

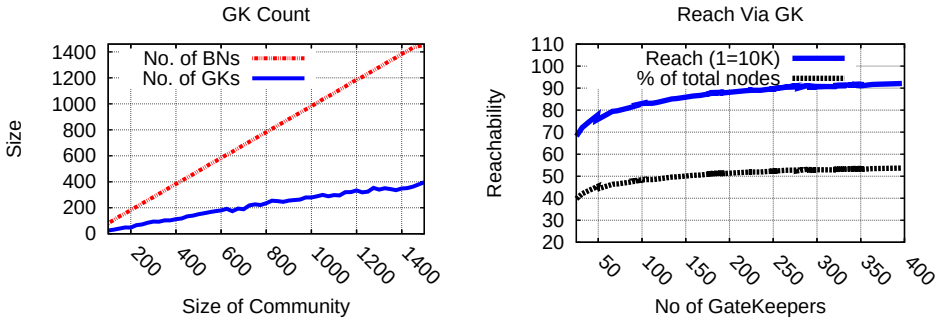
5.1.1 Experiments on Facebook samples

We randomly selected 4000 nodes from the Facebook dataset and tested them for GK selection in *LENS*. The nodes are selected randomly with the constraints that the community size should be between 100 and 1500; and the number of friends for any given node should be greater than 25.

This is to mimic real social network size.



(a) Facebook dataset



(b) Flikr dataset

Figure 10: Number of GK for receiving messages and the reachability of recipient via GKs

Number of GKs for receiving messages

Figure 10(a) (left) presents the number of GKs selected for a recipient to receive messages from outside its community. The number of required GKs is quite reasonable, ranging from 56 to 871 but mostly remaining less than half of the community size. The number of GKs show a nearly linear relationship with the number of boundary nodes. Increase in the number of boundary nodes results in a relative increase in the number of GKs but this is not always the case. It is also observed that a higher number of boundary nodes result in smaller number of GKs. The GK number is lower if the GK is selected from a region where the nodes have high clustering coefficient, which results in the suggestion of the same GK from a number of boundary nodes.

Reachability of recipient via GKs

Figure 10(a) (right) shows the number of users that can reach a particular recipient with the help of GKs. With a minimum number of GKs, the reachability of the recipient is ranging from 700K to 1.7 million i.e. 22% to 55% of the total network, and remains above 40% most of the time. All of these can be achieved by only the execution of stage 1 GK selection process. Based on these results, we can safely assume that in reality there would rarely be a message sent to the recipient by a sender that is not covered by the GKs.The recursive iterations of stage 2 and the use of stage 3 of the GK selection process will handle the rare cases of a legitimate message not being handled.

5.1.2 Experiments on Flickr samples

In the Flickr case, we randomly selected 4000 nodes from the Flickr dataset and evaluated them for GK selection in *LENS*. The nodes have a community size between 100 and 1500 and an average number of friends greater than 25 (same setting as for Facebook).

Number of GKs for receiving messages

Figure 10(b) (left) shows the number of GKs for a recipient to receive messages from outside its community. The resulting number of GKs range between 20 to 400 and most of the time the numbers are less than 30% the community size.

Reachability of recipient via GKs

In Figure 10(b) (right) we present the number of users that can reach a particular recipient with the help of GKs. With the selected GKs above, the reachability of the recipient is ranging between 680K to 920K i.e. 40% to 54 % of the total network and mostly it remains above 45%. Flickr is not a pure social network and is intended primarily for publishing, organizing and locating content. It contains a large number of strongly connected cores of very high degree nodes. Due to this most of the boundary nodes end up suggesting the same node as GK, thus, resulting in smaller number of GKs covering a large number of users. The case with Facebook is different as it is a pure social network with the primary purpose to find and connect to new users.

5.1.3 Experiments on Buzz samples

In Buzz dataset, the community size of approximately 98% of the users is less than 1500. Based on that we randomly selected 1 million nodes from the Buzz dataset and tested them for GK selection in *LENS*.

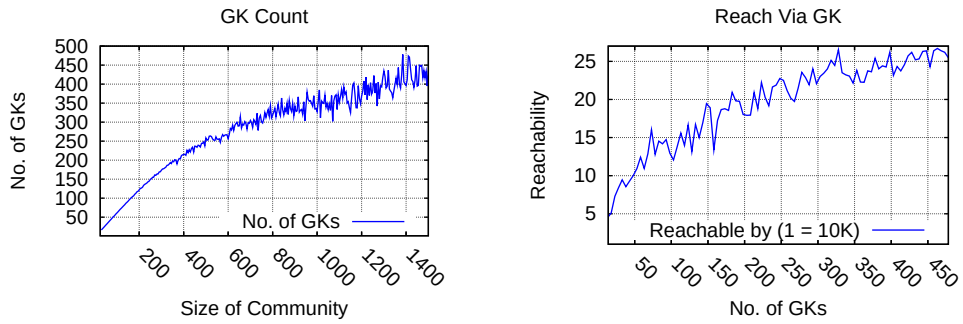
Number of GKs for receiving messages

Figure 11(a) presents the number of GKs selected for a recipient to receive messages from outside its community. The number of required GKs ranges between 14 to 478 and mostly remains less than half of the community size. In some cases it is also observed that even with a bigger community size the resulted number of selected GKs are smaller. The GK number is lower if the GK is selected from a region where the nodes have high clustering coefficient, which results in the suggestion of the same GK from a number of boundary nodes.

Reachability of recipient via GKs

Figure 11(b) shows the number of users that can reach a particular recipient with the help of GKs. With a minimum number of GKs, the reachability of the recipient ranges from 15K to 260K users. Figure 11(c) show that with the selection of 0.02% of the nodes as GKs a recipient can be reached by 12.5% of the total network. All of these can be achieved by only the execution of stage 1 GK selection process.

In Buzz, a total of 358K (i.e. 16%) nodes are selected as GKs for the total userbase. In Figure 11(d) we have distributed the selected GKs according to the number of recipients they are selected for. More than 80% of the nodes selected as GKs have less than 300 recipients. Thus, the additional burden of altruistic processing imposed on GKs is not overwhelming for any GK.



(a) No. of GKs

(b) Reachability (#) via GKs

(c) Reachability (%) via GKs

(d) Recipients per GK

Figure 11: Buzz dataset

Further, if a GK will be compromised by a spammer, the affected number of recipients will remain limited.

Based on the results presented in 5.1, we can conclude that *LENS* is scalable in terms of number of required GKs and the reachability. With the help of only hundreds of GKs, a recipient can be reached by millions of users and the solution can be scalable extended to the users with even further social distance by further GK selection. In contrast, RE: which handles up to FoF, will only be reachable reliably from his community (friends, FoF). The reachability is RE: on average is 0.051%, 0.25% and 0.01% of the network for each recipient in Facebook, Flikr and Buzz respectively. Increase in the size of a recipient's community has a direct impact on his reachability. User having a larger social community would be benefiting more from *LENS* than the isolated and less socially connected users.

Email data set	Enron	Uni-Kiel
Number of Users	52,747	57,158
Number of Edges	74,248	22,648
Messages Exchanged	1,136,760	447,543
Avg Community	167	22

Table 3: High-level stats email datasets

5.2 Real Email Trace Driven Evaluation

We use two real email traces, from both a large commercial and a large academic unit, to evaluation the effectiveness of *LENS* in accepting the inbound emails. The first email dataset is taken from Enron [2]. It contains data from mostly senior management of Enron. The corpus contains a total of about 0.5 million messages. Messages with multiple recipients are counted as one message per recipient. In total, there are about 1,136,760 messages exchanged between 52,747 users. The second email dataset is taken from the log files of the email server at Kiel University [17], which records the source and destination of every email from or to a student account over a period of 112 days. The dataset consists of 44,7543 messages exchanged between 57,158 users.

Similar to previous studies [13, 26, 28], we extracted a social network from the email data by examining the messages sent between the users. Specifically, we created an edge between users who sent at least three emails to each other. These edges are used to form social graphs for the email users. Table 3 presents the high-level statistics of Enron [2] and Uni-Kiel [17] datasets.

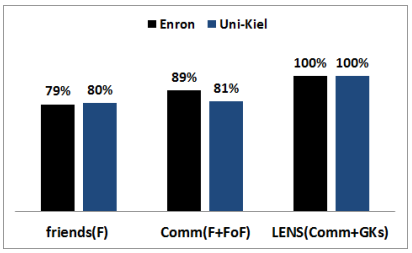


Figure 12: Friend, community and *LENS* based filtering

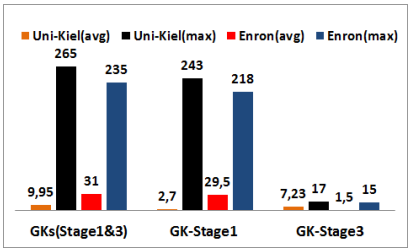
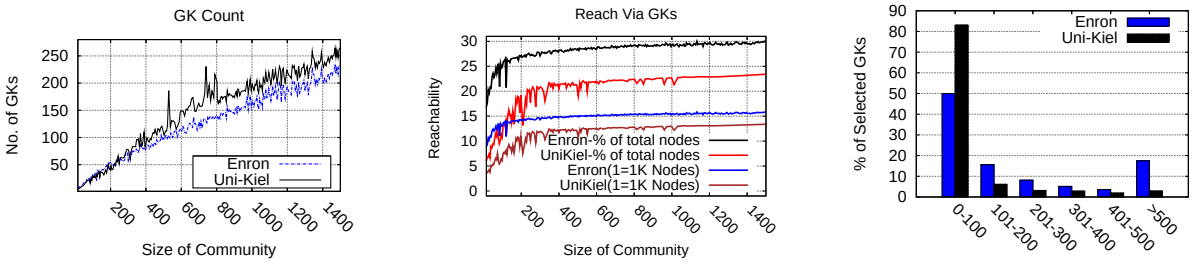


Figure 13: Average GKs for Enron and Uni-Kiel



(a) No. of GKs (b) Reachability via GKs (c) Recipients per GK

Figure 14: Enron and Uni-Kiel email traces

Figure 12 shows the *successful filtration of emails based on friends, community and LENS* on Enron and Uni-Kiel email traces. First of all, we use the edge information to form a graph between the email users and apply a friend filter to accept the inbound emails i.e. accept an email if it is from a friend. Based on the friend filter we are able to accept 79 % of emails in Enron and 80 % in Uni-Kiel datasets. With the application of community filter, the acceptance rate increase to 89 % in Enron and 81 % in Uni-Kiel datasets. Finally, we use *LENS* on the datasets with GK selection procedure at stage 1 and stage 3. The results shows that with the application of *LENS* we can effectively filter and accept all the legitimate inbound emails.

The *number of GKs required*, with the application of *LENS*, in Enron and Uni-Kiel datasets are quite reasonable (see Figure 14(a)). In Enron, the resulting number of GKs range between 6 to 235. The overall average of selected GKs is 31 per user (see Figure 13). Out of these 31 GKs, 29.5 GKs are selected in stage 1 of GK selection process and only 1.5 GKs are selected spontaneously in stage 3. In the worse case we have a GK count of 218 for stage 1 and 15 for stage 3. On the other hand, in Uni-Kiel, the number of selected GKs are between 5 to 265. The overall average of selected GKs is 9.95 per user (see Figure 13). Out of these 9.95 GKs, 7.7 GKs are selected in stage 1 of GK selection process and only 2.2 GKs are selected spontaneously in stage 3. In the worse case, we have a GK count of 243 for stage 1 and 17 for stage 3.

In Figure 14(b) we present the reachability of a particular recipient with the help of GKs[6] only. With the selected GKs above, the reachability of the recipient is ranging between 8.9K to 15.8K i.e. 16.8% to 30% of the total users in Enron. In Uni-Kiel, the reachability of the recipient is between 3.3K to 13.4K i.e. 5.8% to 23.4% of the total users. In contrast, the reachability in RE:(depending on the community size) on average is 0.31% and 0.04% of the network for each recipient in Enron and Uni-Kiel respectively.

In Enron, a total of 3911 nodes are selected as GKs, which constitutes 7.4% of the total number of user. On the other hand, in Uni-Kiel, 2944 nodes are selected as GKs which is approximately 5.7% of the total number of users. In Figure 14(c) we have distributed the selected GKs according to the number of recipients they are selected for. In Enron and Uni-Kiel, the distribution pattern of the number of recipients per GK is similar to that of Buzz. For Enron, 75% of the selected GKs have less than 300 recipients. Whereas in Uni-Kiel, 92% of the selected GKs have less than 300 recipients. Thus, if a GK is compromised by some malicious user very limited proportion of recipients will be effected with spam un till the system recovers back.

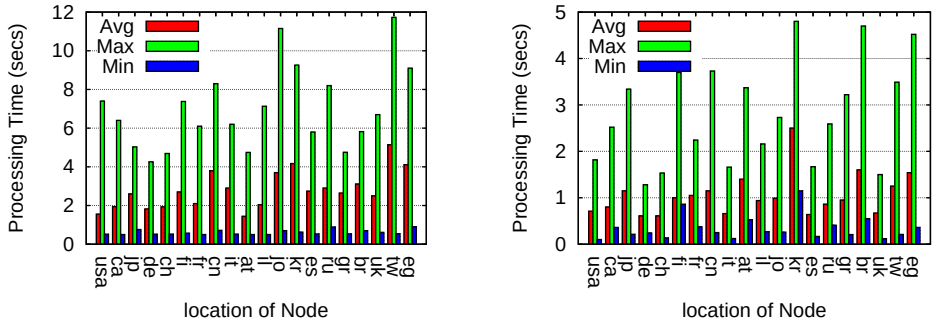
Based on the results presented in this section, we can confidently conclude that *LENS* is effective in efficiently accepting legitimate emails. With the help of only dozens of GKs, a recipient can successfully receive all the legitimate inbound emails.

5.3 Performance of GK Selection Protocol

The performance of GK selection protocol was evaluated on PlanetLab. We wanted to see how the protocol will behave on geographically distant servers i.e how much will be the latency if the MSs are located in different countries. We used 20 nodes across the globe on PlanetLab and evaluated the GK selection protocol in stage 1 and stage 3. In our experiments each node sends out a request to one of the other 19 nodes after every 0.1 second. Every second there were 200 random requests.

In stage 3, the protocol only involves two parties, the MS of the recipient and the GK itself.

[6]We do not count the reachability achieved by Friends and FoF in this case and only purely limited to GKs.



(a) Stage 1 (b) Stage 3

Figure 15: GK Selection: X-axis values represent country codes of the location of the MS

Figure 15(b) shows the result of stage 3. The results are averaged on each node. The average execution time of the protocol remained between 0.5 secs to 1.5 secs with the only exception of kr node where the average execution time remained 2.6 secs. More than 60% of the time was spent in the transmission of the message and the actual processing time remained less than 40%.

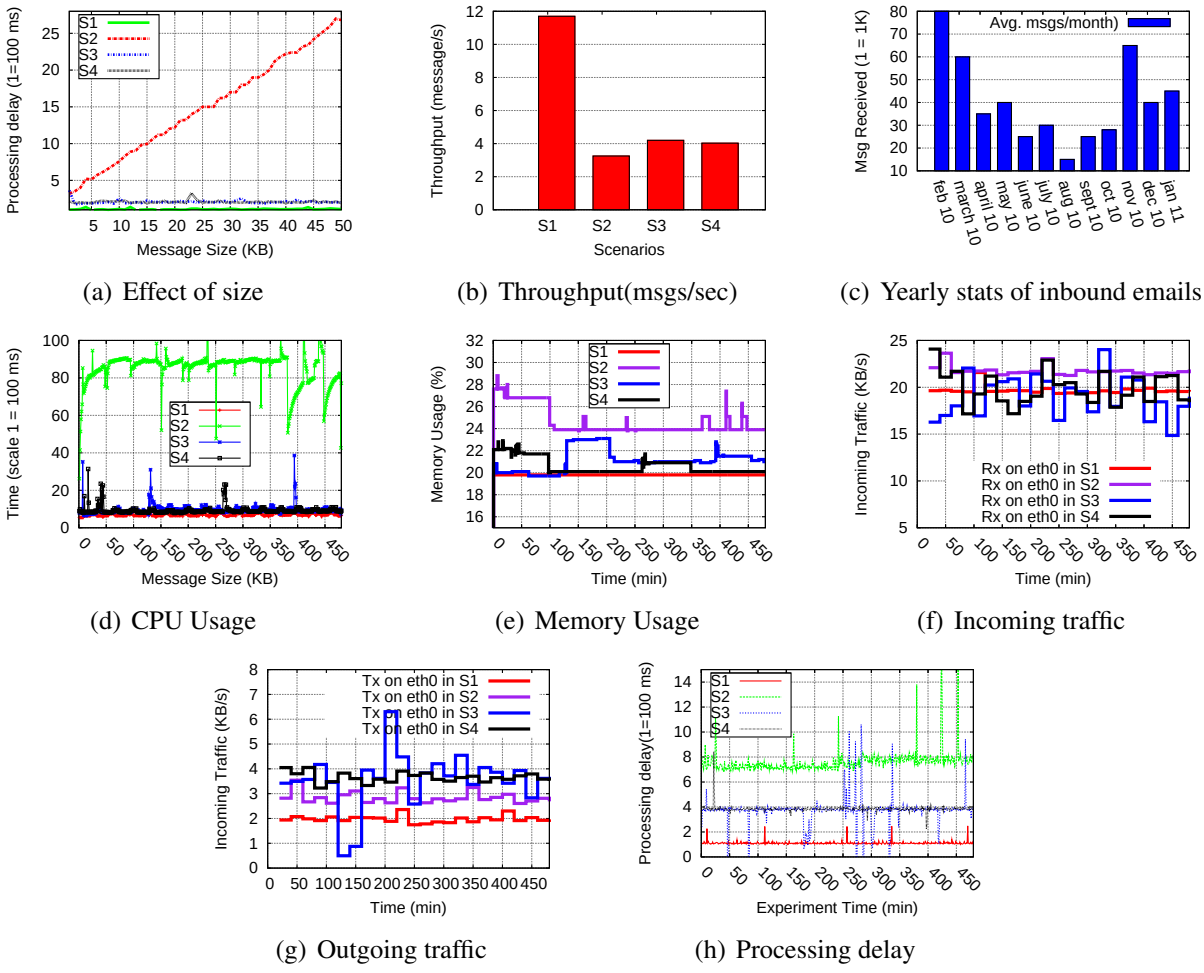
Figure 15(a) shows the average execution time of stage 1 GK selection protocol. During the experiment the GK, the recipient and the FoF remained on separate nodes. The average execution time of the protocol remained between 1.5 secs to 4.5 secs. GK Selection in stage 1 involves an extra step where the GKs are suggested by FoF of the recipient. Due to this the execution time in stage1 is higher than stage 3. Considering the fact that stage 1 is independent of the transmission of email messages, the worse case average delay of 4.5 secs is tolerable.

Both the stage 1 and stage 3 experiments finish with a success rate of 94.6% to 100%. Only of few nodes like kr, jo, eg and br the success rate was not 100%. A request is counted as a failure if there is packet loss noticed at some of the nodes and when the connection request is ignored as the node is already serving maximum number of connections. The amount of traffic on average was approximately 2.3 KB in stage 3 and 2.9 KB in stage 1 (slightly higher due to exchange of extra request and suggestion messages between the recipient and his FoF).

5.4 Performance of Email Processing with *LENS*

To analyze the system performance of email processing, we augment a standard mail processing system with *LENS* and measure its impact. For all the experiments, we used an SMTP server of 2.53 GHz Intel core2duo processor and 4 GB RAM and three senders, each running on a different machine connected via a local area network (this setup is similar to RE:). Our experimental setup consists of four different scenarios. For the first scenario (S1) the SMTP server is only running postfix without any spam filter. In the second scenario (S2) we used SpamAssassin as a content-based filter with Postfix. In the third scenarios (S3) we have used MailAvenger on top of postfix and enabled *LENS* community based filtering i.e. SMTP server is able to filter (accept/reject) emails at the RCPT To: command based on the recipient's CommList (containing 10K entries for all the experiments). Fourth scenarios (S4) is similar to S3 with additional functionality of filtering emails based on the voucher's issued by the authorized GKs of the recipient. We run different experiments using these four scenarios to measure the effect of message size, end to end throughput and CPU,

Memory and bandwidth consumptions.



(a) Effect of size (b) Throughput(msgs/sec) (c) Yearly stats of inbound emails

(d) CPU Usage (e) Memory Usage (f) Incoming traffic

(g) Outgoing traffic (h) Processing delay

Figure 16: Performance evaluation of email processing with *LENS*

5.4.1 Effect of message size

We measure how the size of the message effects the time required to process it. For this we sent 50K messages with varying sizes (1 KB to 50KB) after every second using all the four experimental scenarios. Figure 16(a) shows the processing delays of all the four scenarios. S2 exhibit a linear increase in the processing time with the increase in the size of the message and takes 2.6 secs to process a message of 50KB. On the other hand S1, S3 and S4 remained unaffected by the message size. S3 and S4 takes slightly more time (0.2 to 0.3 secs more) than S1, this is primarily due to the additional overhead from MailAvenger and *LENS* based filtering.

The cost of checking signature has been so far considered substantial. In order to calculate the computational complexity of signature generation/verification we perform OpenSSL [5] speed measurement run on the servers. It takes approximately 0.13 ms to sign and 0.07 ms to verify a message using 1024 bits RSA and SHA-1. An interesting thing to notice is the very close similarity in the processing delays of S3 and S4. This is based on the fact that in S4 the MS does not have

to traverse the CommList (which is fixed at 10K entries in the experiments) and only verify the signature issued by the GK. And apparently (based on the results) the processing time required to process the CommList is very close to the verification of signature. In short, *LENS* based email processing incurs negligible amount of additional processing delays and remains very close to S1 (no spam filtering).

5.4.2 Throughput

In order to measure the end-to-end throughput of the MS in the four scenarios, the senders simultaneously bombard the MS with sufficiently many messages to saturate it (sending 1000 messages of 8 KB each as rapidly as possible). Figure 16(b) shows the throughput of all the four scenarios. Even with the bombardment of messages S1 was able to receive 11 messages per second. This reduced down to 4.2, 4.03 and 3.2 in S3, S4 and S2 respectively.

5.4.3 CPU, memory, bandwidth and delays

We used the inbound email statistics from a large university over the period of one year i.e from February 2010 to January 2011. The overall inbound emails are averaged to approximately 42K per day (see Figure 16(c)). We approximated the total number of inbound emails to 50K/day and limit the transmission time to 8 hours. Based on that, after every 0.6 sec the MS receives an email. Based on this university's statistics and our approximations, we run an experiment by sending emails to the MS after every 0.6 of fixed size (e.g 8KB) for 8 hours (480 minutes) and calculate the usage of CPU, memory, bandwidth and processing delay for S1, S2, S3 and S4.

Figure 16(d) shows the CPU usage for all the four scenarios. Throughout the experiment S2 remained very expensive in terms of CPU usage (80%-90%). This is because content-based filtering has to apply different rules and filters. However, the CPU usage patterns of S3 and S4 remained similar to S1 (avg of 10%-12%).

Figure 16(e) shows the Memory usage. For S1 the memory usage remained almost constant to 20%. In S3 and S4 the memory usage remained between 20% to 23.3%. S3 remained higher than S4 most of the time due to CommList look-up. S2 constantly consumed the most memory (from 23.9% to 29.1%).

Figure 16(f) and 16(g) shows the Inbound and outbound traffic for all the four scenarios. Since we have kept the message size constant there is no clear distinguishable pattern between the four scenarios and the traffic patterns show random changes and overall remain similar. The inbound traffic remains between 16 KB/s to 22 KB/s and the outbound between 2 KB/s to 4 KB/s. The outbound traffic mostly constitutes the SMTP transactional traffic.

Figure 16(h) shows the processing delay for all the four scenarios. The processing delay of all the scenarios almost remained constant due to fix message frequency and size. However, the interesting point to note is that that processing delays in S3 and S4 remained lower than S2 by nearly half and are very close to S1.

Based on the results presented in this section, we conclude that *LENS* (S3 and S4) are computationally quite efficient and nearly similar to the scenarios where there is not spam filtering being used.

6 DISCUSSION

In this section, we discuss some design decisions and questions related to the usage of *LENS*.

False positives and negatives: In *LENS* spam prevention is based on social communities and GK formation, and not on the text filters and tokens as used in content-based filtering. Hence, not being based on content filtering, *LENS* does not encounter any false positive/false negatives generated by imprecise content signatures. However, based on false deduction of social context malicious nodes could become part of the community and result in false-negatives (let spam pass through), but still there will be no false positives (dropping good emails).

Weakness of trust relationships and privacy concerns: In [10] Bilge et al. showed the ease with which an attacker can automatically clone Facebook user profiles and convince large fractions of the victim's friends to establish a friendship relation with the cloned (malicious) contact. Email network can be turned into a social network/graph based on the communication pattern. In email social networks it is not the same to clone a profile (which is the user's email address itself) based on publicly available data like in facebook or any other OSN. *LENS* identifies the notion of trust between two users with the notion of friendship in email social networks. In *LENS* when two users add each other as friends it means that both trust each other to be non-spammer and not necessarily some close friends. This trust is based on personal acquaintance or exchange of messages over time. The notion of friendship in an email social network is not necessarily associated to full trust with respect to the other individual security practices, nor to his ability to wisely choosing its friends. *LENS* address the existence of weak links (or malicious nodes) in social relations by simply reporting community members that prove to be malicious over time.

Friendship information is private and *LENS* does not exchange contacts, CommList, PKList and VoucherList between different MSs. All the information is kept on the user's MS just like user's very private/personal messages, email contacts and authentication (for e.g password) information. This data is normally protected under privacy and data protection laws.

Change in Internet infrastructure: Basic requirement for any practical system is that it does not make major changes in the Internet infrastructure (e.g. SMTP or other protocols). *LENS* does not make changes in existing protocols (like SMTP). Community formation, GK selection and trust management run as independent components. *LENS* based email processing (using CommList and vouchers issued by the GK) run as an independent daemon (like spamd for SpamAssassin) to monitor the SMTP transaction and based on the results take different actions.

7 CONCLUSION

We designed, implemented and evaluated *LENS*, a novel spam protection system that is incrementally deployable and has low processing overheads. *LENS* takes an unusual trusting approach and each recipient enlists the help of the extended social network as gatekeepers against spam, including nodes that are strangers to the recipient.

We evaluated *LENS* using empirical OSN datasets, and prove the system to be scalable with large fraction of users. Further we demonstrate, using real email traces, that *LENS* is quite effective

in accepting all the inbound emails efficiently. *LENS* remains lightweight for email processing. Our system evaluations show that *LENS* consumes up to 75% less CPU and 9% less memory as traditional solutions like SpamAssassin.

References

[1] Email address limit in webmail by providers. http://www.emailaddressmanager.com/tips/email-address-limit.html.

[2] Enron email dataset. http://www.cs.cmu.edu/ enron/.

[3] Internet 2010 in numbers. http://royal.pingdom.com/2011/01/12/internet-2010-in-numbers/.

[4] Mail avenger. http://www.mailavenger.org/.

[5] Openssl. http://www.openssl.org/.

[6] Postfix. http://www.postfix.org/.

[7] Spamassassin. http://spamassassin.apache.org/.

[8] Tracking the high cost of spam. http://www.redcondor.com/company/.

[9] Messagelabs intelligence report: Spam intercepts timeline. http://www.messagelabs.co.uk/, July 2005.

[10] Leyla Bilge, Thorsten Strufe, Davide Balzarotti, and Engin Kirda. All your contacts are belong to us: automated identity theft attacks on social networks. In *Proc. of WWW*, 2009.

[11] J. Bonneau, J. Anderson, R. Anderson, and F. Stajano. Eight friends are enough: social graph approximation via public listings. In *Proc. Second ACM EuroSys Workshop on Social Network Systems*, 2009.

[12] P. O. Boykin and V. Roychowdhury. Personal email networks: An effective anti-spam tool. *IEEE COMPUTER*, 2004.

[13] A. Chapanond, M. S. Krishnamoorthy, and B. Yener. Graph theoretic and spectral analysis of enron email data. *Computational and Mathematical Organization Theory*, October 2005.

[14] Paul Alexandru Chirita, Jörg Diederich, and Wolfgang Nejdl. Mailrank: using ranking for spam detection. In *Proc. of CIKM*.

[15] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. Planetlab: An overlay testbed for broad-coverage services. *ACM SIGCOMM Computer Communication Review*.

[16] G. Danezis and B. Wittneben. The economics of mass surveillance-and the questionable value of anonymous communications. In *Proc. of WEIS*, 2006.

[17] H. Ebel, L-I. Mielsch, and S. Bornholdt. Scale-free topology of e-mail networks. *Phys. Rev. E*, Sep 2002.

[18] S. Garriss, M. Kaminsky, M. J. Freedman, B. Karp, D. Mazi'eres, and H. Yu. Re: Reliable email. In *Proc. of NSDI*, 2006.

[19] Shuang Hao, Nadeem Ahmed Syed, Nick Feamster, Er G. Gray, and Sven Krasser. Detecting spammers with snare: Spatio-temporal network-level automatic reputation engine. In *USENIX Security*, 2009.

[20] Lisa Johansen, Michael Rowell, Kevin Butler, and Patrick Mcdaniel. Email communities of interest. In *Proc. of CEAS*, 2007.

[21] Mohamed Ali Kaafar and Pere Manils. Why spammers should thank google? In *Proceedings of the 3rd Workshop on Social Network Systems*, SNS '10, pages 4:1–4:6, New York, NY, USA, 2010. ACM.

[22] J. Klensin. Simple mail transfer protocol, j. klensin. The Internet Society, RFC 5321, October 2008.

[23] H. Lam and D. Yeung. A learning approach to spam detection based on social networks. In *Proc. of CEAS*, 2007.

[24] Z. Li and H. Shen. Soap: A social network aided personalized and effective spam filter to clean your e-mail box. In *Proc. of IEEE INFOCOM (to appear)*, 2011.

[25] Naftaly Minsky. Reducing spam via trustworthy self regulation by email senders. In *the MIT Spam Conference*, March 2010.

[26] A. Mislove, A. Post, P. Druschel, and KP Gummadi. Ostra: Leveraging trust to thwart unwanted communication. In *Proc. of NSDI*, 2008.

[27] Alan Mislove, Massilmiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proc. of IMC*, 2007.

[28] J. Shetty and J. Adibi. The enron email dataset database schema and brief statistical report. Technical report, Information Sciences Institute, 2004.

[29] M. Sirivianos, K. Kim, and X. Yang. Introducing social trust to collaborative spam mitigation. In *Proc. of IEEE INFOCOM (to appear)*, 2011.

[30] J. Travers and S. Milgram. An experimental study of the small world problem. *Sociometry*, 1969.

[31] Steve Webb, James Caverlee, and Calton Pu. Social honeypots: Making friends with a spammer near you. In *Proc. of CEAS*, 2008.

[32] Christo Wilson, Bryce Boe, Alessandra Sala, Krishna P.N. Puttaswamy, and Ben Y. Zhao. User interactions in social networks and their implications. In *Proc. of EuroSys*, 2009.

[33] M. W. Wong. Sender authentication: What to do. http://spf.pobox.com/whitepaper.pdf, July 2005.

[34] B. Wu and B. Davison. Identifying link farm spam pages. In *Proc. of the 14th International WWW conference*, 2005.

[35] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao. Sybillimit: A near-optimal social network defense against sybil attacks. *Networking, IEEE/ACM Transactions on*, 2009.

[36] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. In *Proc. of SIGCOMM*, 2006.