Usable Secure Mailing Lists with Untrusted Servers

Rakesh Bobba, Joe Muggli, Meenal Pant, Jim Basney and Himanshu Khurana
University of Illinois, Urbana-Champaign
{rbobba, jmuggli, mpant, jbasney}@ncsa.uiuc.edu, hkhurana@iti.uiuc.edu

ABSTRACT
Mailing lists are a natural technology for supporting messaging in multi-party, cross-domain collaborative tasks. However, whenever sensitive information is exchanged on such lists, security becomes crucial. We have earlier developed a prototype secure mailing list solution called SELS (Secure Email List Services) based on proxy encryption techniques [20], which enables the transformation of cipher-text from one key to another without revealing the plain-text. Emails exchanged using SELS are ensured confidentiality, integrity, and authentication. This includes ensuring their confidentiality while in transit at the list server; a functionality that is uniquely supported by SELS through proxy re-encryption. In this work we describe our efforts in studying and enhancing the usability of the software system and our experiences in supporting a production environment that currently is used by more than 50 users in 11 organizations. As evidence of its deployability, SELS is compatible with common email clients including Outlook, Thunderbird, Mac Mail, Emacs, and Mutt. As evidence of its usability, the software is being used by several national and international incident response teams.

Categories and Subject Descriptors
H.4.3 [Information Systems Applications]: Communications Applications—Electronic Mail; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Evaluation/methodology; E.3 [Data Encryption]

General Terms
Design, Security, Human Factors

Keywords
E-mail List Security, Proxy Re-encryption, Usability Study

1. INTRODUCTION
As more and more user communities are engaging in collaborative tasks, use of Email List Services (or simply Mailing Lists - MLs) is becoming common; i.e., emails exchanged with the help of a list server (examples of commonly used list server software include Mailman and Majordomo). Many tasks where MLs are used require exchange of private information, especially those that involve messaging in collaborations across multiple administrative domains. For example, a ML of security administrators that manage critical infrastructure would not want their emails disclosed to hackers. Specific instances include the multi-domain Open Science Grid ([http://www.opensciencegrid.org/](http://www.opensciencegrid.org/)) and TeraGrid ([http://security.teragrid.org/](http://security.teragrid.org/)) systems where the Incident Handling and Response policies recommend the use of encrypted and signed mailing lists. In general, use of encrypted and signed lists is recommended for incident response by IETF [5] and CERT [31]. Additional examples include a list of (1) health care and pharmaceutical researchers who want to protect patient privacy, (2) corporate executives who want to protect proprietary information, and (3) researchers engaged in collaborative research involving multiple university, government and industry institutions who want to protect their intellectual property.

For such MLs cryptographic solutions are needed that provide adequate protection (i.e., confidentiality, integrity, and authentication) for the private content from threats at the client side, at the network paths where the emails are in transit, and at the server side where the emails are processed for distribution to the list. That is, there is a need to develop Secure Mailing Lists (SMLs) as illustrated in Figure 1. Threats to the server side are an important concern in practice and lack of good solutions today has forced users to develop their own clunky ones. For example, several critical infrastructure security protection groups today use out-of-band means to distribute passwords to members and require members to use password-based-encryption (supported by commonly available GnuPG plug-ins) so that the list server does not have access to email plain-text, minimizing the trust that must be placed in the mail server.

To address this challenge for mailing lists we have earlier developed a prototype SELS (Secure Email List Service) [20]. The SELS protocol and software prototype satisfy requirements in the categories of security properties (i.e., confidentiality, integrity, and authentication), infrastructure compatibility, key management and performance. For confidentiality SELS uses proxy encryption techniques that allow the list server to transform email cipher-text between list
high-level goals were to evaluate and enhance the usability of SELS key management by (1) exploring two alternative key trust establishment techniques, (2) developing an effective password management solution that balances usability and security, and (3) identifying suitable interface cues that can be introduced to mitigate remaining vulnerabilities in environments where new software or plug-ins can be deployed.

A fully functional version of the software has been packaged and released for community use. We are actively supporting the software via a public email list and making new releases to fix bugs and add features. Our first user community is incident responders and several national and international incident response teams have adopted SELS for email exchange. Currently, these include the TeraGrid Security Working Group and the International Grid Trust Federation (http://www.igtf.net/). We report on our experiences in supporting the TeraGrid Security WG for a period of ten months. We present SELS usage statistics, discuss usability and security issues observed in practice and present software engineering enhancements. Success of SELS is clearly indicated by the fact that there has been a nearly four-fold increase in the number of encrypted emails exchanged by the TeraGrid users since they adopted SELS, with anecdotal evidence suggesting that this increase is largely due to better usability provided by SELS.

The rest of this paper is organized as follows. In Section 2 we give an overview of the SELS protocol and software prototype. In Section 3 we evaluate the usability of SELS. In Section 4 we describe the SELS production environment and our experiences with supporting the TG Security WG. In Section 5 we discuss related work and conclude in Section 6.

2. SELS OVERVIEW

In this section we provide an overview of the SELS system architecture focusing more on the list subscriber interaction. We refer the reader to [20] for further details. SELS provides confidentiality, integrity and authentication for emails exchanged in a mailing list via public key encryption and signing by interactions among the following entities.

- **List Moderator (LM).** LM is a user (or process) that creates a list to be maintained at the list server, authenticates users, and helps them subscribe to and unsubscribe from the list.
- **List Server (LS).** LS creates lists, maintains membership information (email addresses and key material), adds and removes subscribers based on information received from LM, and forwards emails sent by a valid list subscriber to all current subscribers of that list.
- **Users/Subscribers.** Users subscribe to lists by sending join requests to LM and send emails to the list with the help of LS.

The first major goal of SELS is to minimize trust in LS such that LS is unable to access email contents while still providing the necessary list management and email forwarding capabilities. As mentioned in the Introduction, threats to LS are an important concern. First, compromise of a
trusted list server by an attacker could result in significant disclosure of sensitive information. Second, for lists with members from multiple organizations, it is difficult to trust a single organization to host the list server. Traditional solutions that provide server-side protection tend to have a high overhead for key management. For example, the common practice of out-of-band password distribution for password-based encryption requires users to remember a long list of passwords to decrypt previously sent emails. Instead, SELS achieves this via proxy encryption where LS re-encrypts messages between list subscribers by using proxy private keys but without requiring access to the plain-text.

A high-level view of the three-step SELS protocol is presented in Figure 2. In the first step LM and LS create a list \( L \), which involves them establishing an ElGamal key pair each for the list and distributing the list public key computed to be the product of their public keys. Note that no single entity has access to the list private key, \( K_L \). In the second step, users subscribe to list \( L \) by contacting LM. LM creates a key pair for each user and sends it to that user. In addition, LM sends keying material to LS, which allows LS to establish the private proxy key for that user. These keys are all computed from LM and LS’s list keys by simple addition and subtraction of random numbers to ensure that the sum of each user’s private and private proxy keys is the list private key, \( K_L \), originally established by LM and LS. This invariant allows for the proxy re-encryption process to execute correctly. In the third step, users send email encrypted with the list public key, \( PK_L \), and optionally signed by their own private keys. LM executes proxy re-encryption on these emails once for each subscriber and sends the output to that subscriber. Each subscriber can then decrypt the message with their private key.

The second major goal of SELS is to drive the design and development process with deployability and adoption in mind. To that end SELS (1) uses the OpenPGP message format \([6]\) and standard GnuPG plug-ins \( \text{http://gnupg.org} \) on the client side allowing users to use their existing email clients and (2) uses open-source off-the-shelf components such as the Mailman list server. By using COTS GnuPG components as illustrated in Figure 3 on the client side SELS becomes compatible with any email client for which a GnuPG plug-in is available and that includes popular email clients such as Outlook, Thunderbird, Mac Mail, Emacs and Mutt. We believe this compatibility to be a crucial factor for successful deployment of SELS and this belief has been supported by experiences working with real user communities. The developed components for LM and LS use open-source GnuPG and Bouncycastle libraries as well as GnuPG key management functions.

In the world of secure email, OpenPGP and S/MIME are competing standards with native support of S/MIME in email clients being more prevalent. The proxy encryption technique used in SELS requires transformation of text between two public keys, which, in turn, requires that the keys share common parameters. The chosen cryptosystem, ElGamal, easily supports this as it is a group based cryptosystem. However, with the RSA cryptosystem this would imply that the modulus be shared between multiple key-pairs, which is inherently insecure. S/MIME supports RSA but not ElGamal (specifically for message encryption), limiting SELS compatibility to OpenPGP for now. Recently, Elliptic Curve Cryptography (ECC) has been added to S/MIME standards and is also being supported by some email clients. ECC is also a group based cryptosystem, therefore, it will enable the proxy encryption technique to be employed. In the future we plan to support ECC and, in turn, achieve compatibility with S/MIME for broader adoption of SELS.

We conducted a performance evaluation of SELS and focused on the most expensive operation — the proxy transformation step at the list server. Consequently, we measured throughput of the list server with varying list sizes and message sizes. Overall, we saw that even the worst throughput of 2.5 messages/sec for SELS with list size 10 and message size 100KB corresponds to a throughput of more than 200,000 messages per day. Since most mail servers in small and medium-sized organizations do not typically process more than 100,000 messages per day (of which only a subset are ML messages) we conclude that adding security to MLs will not impose an undue overhead on the mail servers.

3. USABILITY EVALUATION

3.1 Approach

The high-level goals of the usability study were to evaluate and enhance the usability (i.e., effectiveness, efficiency, and satisfaction) of the SELS key management system for list
subscribers with the constraint that the solution be compatible with existing deployed email clients. To address these goals we combined several usability techniques in executing this study. First, the groupware walkthrough technique (which is based on cognitive walkthrough) was used to determine relevant key management tasks and candidate usable solutions. Application of the technique suggests two possible solutions for key management with a common password management approach. The two key management approaches utilize different GnuPG key trust establishment mechanisms, namely, key signing and key trust assignment.

We then undertook two rounds of focused user studies to design and evaluate the key installation and password management technique as well as the overall usability of SELS. The first round explored the GnuPG key signing approach of trusting keys while the second round explored the key trust assignment approach. In each round we assessed the users’ ability to successfully complete specified key management tasks, we measured the time they took to do so, and then they filled out the SUS questionnaire to convey their satisfaction. Among the specific tasks were a set of vulnerability assessment tasks whereby users were required to place trust in messages that may or may not be correct. Since we are evaluating security software it is important to ensure that the system does not introduce new vulnerabilities for the users.

These focused studies were undertaken with a relatively high expert-to-novice ratio with novices being defined as those with some basic security concepts and experts being defined as those familiar with GnuPG tools and advanced security concepts. Keeping the intended users in mind (e.g., our current TeraGrid users) as well as the known limitations of common email clients in their ability to provide usable security [13, 32], we decided that all subjects must have at least a basic grasp of security concepts. This allowed us to understand how familiarity with tools and concepts, that novices would learn over time, would affect SELS usability and security. To further help in obtaining this understanding we explicitly asked users to perform basic GnuPG two-party secure email tasks in addition to SELS tasks.

One deviation of our work from most studies of usable security is that we asked the subjects to complete tasks in the context of secure mailing lists but without a particular scenario. For usability studies of security systems these scenarios are often used to motivate the need for security. However, in our case since all users had some background in computer security we felt that a scenario to motivate the need for security may not be very helpful. Instead, we asked the subjects to focus exclusively on interface cues and their knowledge of security concepts for making security decisions. An advantage of this is that the results of the study depend only on the usability of the system and the users’ knowledge of security concepts and not on the extent to which the scenario motivated security.

3.2 Groupware Walkthrough

In the early part of the SELS design and implementation process the authors undertook a groupware walkthrough with the goals of identifying key management tasks and problems as well as candidate solutions. The groupware walkthrough process involves specifying a description of tasks and teamwork and then using the technology to walk through the interfaces from the point of the team users in attempting to accomplish the tasks [27]. We defined these tasks and teamwork based on the SELS protocol described earlier and executed the walkthrough with the authors acting as multiple evaluators working simultaneously and recording results with detailed notes. Figure 4 identifies the tasks and required teamwork but details of subtasks have been omitted for simplicity. Based on interviews with several system administrators and security professionals we chose the following four email clients with available GnuPG plug-ins for the walkthrough: (1) Mac Mail, (2) Mutt, (3) Outlook, and (4) Thunderbird. The walkthrough identified the following three issues.

1. Installation of multiple keys. The List Moderator provides each subscriber with 1) a public key, common to all subscribers of the list, that is used to encrypt messages to the list and 2) a public-private key pair, unique to the subscriber, that is used to decrypt messages received on the list. The user must then add these keys to her GnuPG key ring and assign a passphrase to the private key. She must also place appropriate trust in these keys to be able to use them for sending and receiving secure e-mails as per the GnuPG trust model. In addition the user also obtains a list of public keys belonging to list subscribers from the list moderator for the purpose of signature verification and she must similarly install and trust these keys. A major difference with standard GnuPG use is that SELS requires users to import and install private keys while in standard GnuPG users typically only deal with public keys. In addition to this new concept, we observed that users would have to install several keys so unless the key installation steps were simple the user might get frustrated.

2. Managing and using multiple keys. Whenever a message is received on any list the user must remember the password corresponding to that list and enter it in order to decrypt the message. All email clients studied provide passphrase caching capabilities but limit the caching to only one passphrase, which further compli-
cates management and use of the multiple private keys. We note that these issues of installing and managing multiple keys arises primarily because SELS uses an untrusted server. If the server were fully trusted, then the subscriber would be able to use the same key pair for all lists.

3. Prior GnuPG experience. While conducting the walkthrough it became clear that prior experience of secure email use with GnuPG would greatly benefit the users; however, it was not clear 1) how much prior experience would benefit the users and 2) whether lack of such experience would make the software unusable.

Analysis. The results from our walkthrough provided an opportunity to fix the usability problems early on and to design a focused user study to evaluate the usability and security of the system. For installing keys we designed a sequence of three emails sent from the list moderator with the following contents: (1) the list moderator’s public key that the user needs to import and trust, (2) the list public key as well as the user’s private key (for the list) that the user needs to import and trust, and (3) a set of subscriber keys that the user needs to import. GnuPG allows two ways of achieving trust: signing keys or using the GnuPG trust model. We decided to study both ways of achieving trust with the initial assumption and default implementation that the first approach will work because prior usability studies in secure email indicated that users find it very challenging to deal with the GnuPG trust model \cite{12, 29, 32}. For managing and using multiple keys we proposed a simple approach, namely, recommend that users use the same passphrase to protect all private keys and use the passphrase caching tools to manage that passphrase. Clearly, this is a tradeoff between usability and security but we believe that as long as the users use a strong password their security is only minimally weakened by this approach. To evaluate these approaches we needed a user study that asked users to evaluate SELS with the suggested approaches for installing and managing multiple keys. To characterize how familiarity with the GnuPG interface (or lack thereof) affects the usability of SELS \cite{12}, we used the System Usability Scale (SUS) \cite{4}, which has proven to be a good guide for gauging the usability aspects of effectiveness, efficiency and satisfaction. SUS is designed to give a quick assessment of overall usability and consists of ten questions rated on the Likert scale. SUS has proven effective in practical settings \cite{29} and is also beginning to be used in usability studies of secure software (e.g., Polaris \cite{9}).

We conducted two studies as discussed earlier. In Study I we evaluated the key signing approach and in Study II we used the GnuPG trust assignment approach. All the goals of Study I and Study II were identical to allow for an effective evaluation of proposed approaches.

3.3 Recruiting Users

Our process of recruiting users was guided by the desire to (1) get representative users for the system’s target audience and (2) be able to study the impact of familiarity with a similar interface on SELS usability. For studying the impact of familiarity we needed to categorize our users into experts and novices as follows:

- **Expert.** A user who uses GnuPG to send and receive secure emails at least occasionally and has security-related knowledge experience from one or more of the following: classroom, job, personal interest.

- **Novice.** A user who has used GnuPG at most rarely but has security-related knowledge experience from one or more of the following: classroom, job, personal interest. This knowledge includes basic understanding of concepts of confidentiality, integrity and authentication as well as how these concepts are enabled by public key cryptography.

To recruit users we sent out flyers outlining the study and asking for volunteers. Based on the responses we selected 20 users for the study. Among the users there were two system administrators, seven computer science graduate students, and eleven professional engineers.

3.3.3 Setup and Sequence

The two studies were conducted in a computer lab and administered by two people, a studyadmin who administered the study remotely via email and a studymonitor who observed the users during the study and took notes. Users were asked to choose one of the following four email clients for the

---

\footnote{All of these steps were approved by the Institutional Review Board and signed consent forms were obtained from each user up front.}
study: 1) Thunderbird, 2) Outlook, 3) Mac Mail or 4) Mutt. A GnuPG plug-in was installed on each of these clients and a key-pair was pre-generated for the users. A standard PC with Debian Linux was used for Mutt and Thunderbird and a standard PC with Windows XP was used for Outlook and Thunderbird. A Macbook Pro was used for Mac Mail.

The study was divided into five parts as described below. Part four differed in Study I and Study II. These differences are noted below.

Part I: Background Questions.
User provides background information. This information helps us classify him/her as an expert or novice.

Part II: TPSE Effectiveness.
Studyadmin sends email containing keying material and instructions to the user. The user is asked to 1) import a public-key and place trust in it by signing it, 2) send a signed and encrypted email using the imported key, and 3) decrypt and verify a received email. After completing these tasks the user was asked to fill a SUS questionnaire about the experience of exchanging two-party secure email using GnuPG (TPSE). The user has an option to provide additional feedback if he/she desires.

Part III: TPSE Vulnerability.
The user received six email messages and was asked to decide whether he/she trusted the message based on the security cues provided by the email client’s GnuPG plug-in and their general knowledge of security. He/she was asked to forward the message to “studyadmin” with their trust decision (yes or no) and optionally an explanation for their decision. The five message types used are described in Table 1.

Part IV: SELS Effectiveness.
The user was subscribed to a mailing list hosted by the SELS server and received the email messages with the list keys as described previously. The user was asked to 1) import keying material for a list sent by the list moderator, 2) send signed and encrypted email to the list, and 3) decrypt and verify an email received on the list. Importing keying material for a SELS list involved 1) importing the list moderator’s GnuPG public-key and placing trust in it by signing it, 2) importing an encryption (decryption) GnuPG key-pair to be used for encrypting (decrypting) messages for (on) the list and placing trust in it by signing it, and 3) importing the GnuPG public-keys of all members of the list. Additionally we also recommended that users set the passphrase of the imported key-pair for the list to be same as that of their GnuPG key-pair for ease of use. After completing these tasks, the user was asked to fill a SUS questionnaire about the experience of using SELS for exchanging signed and encrypted email using a list. The user had an option to provide additional feedback if he/she desires.

In Study II users were asked to perform a variation in tasks for part four with the difference being in the way they trust keys. Instead of signing each key explicitly users were asked to place “ultimate” trust in the list moderator’s public key when they received it. Therefore, they did not have to place explicit trust in any keys that were already signed by the list moderator (i.e., the list encryption/decryption key pair and other subscriber’s public keys) as their email clients were able to leverage the transitive trust enabled by the GnuPG key trust assignment model.

Part V: SELS Vulnerability.
This part is similar to part III except that the email messages are sent over the SELS mailing list. The message types used are described in Table 1.

3.3.4 SELS Training and Documentation
Typically when new security software or new features in existing software are evaluated, users are given some training in the software/features. For SELS we decided that engaging in TPSE exercises served as sufficient training because if users are able to install keys and then use them to send and receive secure emails they should be capable of using SELS. Therefore, the initial TPSE usability evaluation served as both training for SELS as well as providing data for evaluating the usability issue of prior experience with GnuPG. To help users in completing these exercises we provided a minimal set of instructions in the emails that described the tasks and included key material. These instructions were independent of any email client in keeping with SELS objectives. For additional clarification, we referred the study user to the the SELS online documentation.

3.4 Observations and Analysis
For each user we took detailed notes of their ability to execute assigned tasks as well as the time taken for each task. The time taken between emails sent to the user that provided the instructions and emails sent back from the user with the results allowed for exact measurements of time taken to complete the tasks. For study parts two and four the studymonitor offered help to the user in the following ways (if requested): (1) when users were stuck they were asked to look at the instructions carefully, (2) when users found additional instructions on specific clients to be incorrect they were offered correct instructions, and (3) when they failed to proceed they were helped as much as needed so that they could proceed to the next part with this part being counted as a failed task.

A total of 12 users participated in Study I with 3 expert users and 9 novice users. Out of the 12 participants, 8 participants chose Thunderbird, 2 participants chose Mac Mail, 1 participant chose Mutt and 1 participant chose Outlook. All users except for one were able to complete enough tasks to allow for user study conclusions. The one that failed involved Microsoft Outlook that kept crashing so this client was excluded from further studies.

A total of 8 users participated in Study II with 3 expert users and 5 novice users. All of them chose to use Thunderbird. While the number of expert users per study is small, previous research shows that a small number of users per class is sufficient to uncover most of the usability issues in a system. Specifically, 3 and 5 users can uncover about 67% and 85% of the usability issues respectively. However, our quantitative measures are preliminary as larger studies are needed to quantitatively measure usability [23].

Interestingly, the GnuPG plug-in for Outlook seemed to crash only when used in conjunction with its key manager — a scenario that was not explored in our groupware walk-through.
Table 1: Message Types Sent to Users during GnuPG and SELS Focused Studies

<table>
<thead>
<tr>
<th>Message Type and Description</th>
<th>Two Party Secure Email (TPSE) using GnuPG</th>
<th>Secure Email List Service (SELS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encrypted and signed correctly</td>
<td>This message is encrypted for the user and signed with a trusted key.</td>
<td>This message is signed and encrypted by a valid member of studylist, with a trusted signature key and the correct list encryption key.</td>
</tr>
<tr>
<td>Encrypted with wrong key</td>
<td>The email message is encrypted with a key that does not belong to the user. Hence the user cannot decrypt it.</td>
<td>The email message is encrypted with a key for which the user has no secret-key and delivered directly to the user but made to look like a message delivered on the list by forging the headers.</td>
</tr>
<tr>
<td>Encrypted and signed with forged “From”</td>
<td>The email message is encrypted with the user’s key, but signed with a key that does not match the “From” address.</td>
<td>The email message is encrypted with the list key, but signed with a key that does not match the “From” address.</td>
</tr>
<tr>
<td>Encrypted correctly but signed with a missing key</td>
<td>This email message is encrypted with the user’s key, but is signed with a key for which the public-key is not available to the user.</td>
<td>The email message is encrypted with the list key, but is signed with a key for which the public-key is not available to the user.</td>
</tr>
<tr>
<td>Encrypted with forged “To”</td>
<td>The user is made to believe that this encrypted message was sent to the user and someone else by forging “To” header.</td>
<td>The user is made to believe that this encrypted only message was sent on the list by forging the headers. It is encrypted such that the user can decrypt it correctly.</td>
</tr>
</tbody>
</table>

Table 2: Usability observations from Study I

<table>
<thead>
<tr>
<th>User Type</th>
<th>Key Install Success Rate</th>
<th>Key Install Time (Avg./Std. Dev. min.)</th>
<th>SUS Score</th>
<th>Changed Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert</td>
<td>2 of 3 (66.6%)</td>
<td>6.5 / 2.12</td>
<td>85.83 / 5.20</td>
<td>3 of 3 (100%)</td>
</tr>
<tr>
<td>Novice</td>
<td>6 of 8 (75%)</td>
<td>8.83 / 2.76</td>
<td>79.38 / 9.33</td>
<td>5 of 8 (62.5%)</td>
</tr>
</tbody>
</table>

Table 3: Usability observations from Study II

<table>
<thead>
<tr>
<th>User Type</th>
<th>Key Install Success Rate</th>
<th>Key Install Time (Avg./Std. Dev. min.)</th>
<th>SUS Score</th>
<th>Changed Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert</td>
<td>3 of 3 (100%)</td>
<td>4 / 0</td>
<td>74.17 / 20.21</td>
<td>2 of 3 (66.6%)</td>
</tr>
<tr>
<td>Novice</td>
<td>4 of 5 (80%)</td>
<td>8.4 / 2.7</td>
<td>61.5 / 10.98</td>
<td>5 of 5 (100%)</td>
</tr>
</tbody>
</table>

Table 4: Security observations from Study I and Study II

<table>
<thead>
<tr>
<th>User Type</th>
<th>% of Correctly Formed Msgs. Trusted (Avg./Std. Dev.)</th>
<th>% of Incorrectly Formed Msgs. Trusted (Avg./Std. Dev.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert</td>
<td>100 / 0</td>
<td>100 / 0</td>
</tr>
<tr>
<td>Novice</td>
<td>93.75 / 17.08</td>
<td>100 / 0</td>
</tr>
</tbody>
</table>

Key Management: Installing and Managing SELS Keys.

The first of the two key management usability issues evaluated by this work is installation of per-list keying material by SELS users. The results of Study I are presented in Table 2. 66.6% of expert users (2 out of 3) and 75% of novice users (6 out of 8) were able to successfully complete the key installation tasks for the second part of the study (i.e., for TPSE) while 66.6% of expert users (2 out of 3) and only 25% (2 out of 8) of novice users were able to complete key installation tasks for the fourth part of the study (i.e., for SELS). Two-thirds of the users who failed to complete the key installation task for SELS and all the users who failed to complete the key installation task for TPSE did so because they either didn’t know how to sign keys or what keys to sign in the case of SELS. In particular, users could not understand what a ‘key-id’ is and how to find the ‘key-id’ of a key. Expert users took 6.5 minutes and 11 minutes on an average to complete key installation for TPSE and SELS respectively. On the other hand, novice users took 8.83 minutes on an average for TPSE key installation and 25.5 minutes on average for SELS key installation when they could successfully complete it. The increase in key-installation time from TPSE to SELS is due to the fact that SELS involves importing secret keys while TPSE only involves importing public keys and, furthermore, SELS involves importing multiple keys while our TPSE exercises involved importing only one key.

These results allowed us to quickly conclude that the SELS key installation process was too cumbersome for novice users. There are two ways to trust keys in GnuPG: using key signing and using explicit key trust assignment. We adopted the first approach for Study I because we assumed that understanding the GnuPG key trust assignment model would be
very complex for users. This assumption was based in part on the complexity of this model as it allows for transitive trust establishment and in part on previous usability studies with secure email. However, it turns out that the key signing approach is not appropriate based on our usability observations. Instead, the key trust assignment model where the users place explicit trust in the list moderator’s key and then use that trust to place transitive trust in all other SELS keys (as the list moderator’s key signs all other keys) turns out be more usable.

The evaluation results of this approach from Study II are presented in Table 3. All the expert and 80% (4 out of 5) of novice users were able to complete key installation tasks for the second part of the study (i.e., for SELS) while all the experts and novices were able to complete key installation tasks for the fourth part of the study (i.e., for SELS). While expert users took 4 minutes and 12.6 minutes on average to complete key installation for TPSE and SELS, respectively, novice users took 8.4 minutes and 18.2 minutes on an average when they could successfully complete it. As can be seen from the results, there is a significant improvement in SELS key installation success rate in the case of novice users. We believe that the reason for this is that users find placing explicit trust in one key, namely, the list moderator’s key, much easier than signing multiple keys. The second usability issue that we dealt with is managing and using multiple keys. The approach that we took to address this is to leverage the GnuPG key management capabilities whereby the plug-ins help locate the appropriate keys (for encryption, decryption, signing, and verification) but use a simplified password management solution. In using secret keys the user is prompted for his password for decryption and signing. Since all clients that we dealt with supported password caching for exactly one password, we recommended users (in SELS instructions) to set the passwords for all secret keys to be the same one. While all the expert users set the passphrase for the list key-pair to be the same as that of their GnuPG key-pair only 37.5% (3 out of 8) of novice users chose to change the passphrase in Study I. All the users who set the passphrase as recommended had little difficulty in sending (receiving) messages to (on) the list while users who did not set the passphrase as recommended had trouble figuring out which passphrase to use. 40% of users who didn’t follow the recommendation initially did so later after realizing the ease of use it afforded. To address this issue we improved the instructions for Study II where we explained the consequences of password change in that it would be easier to manage decryption functions. Users were positively affected by the inclusion of this explanation in the instructions. Consequently, in Study II all novice users chose to change the password so that they have a simple password to deal with. We note that once novice users chose to use a single password, they had no difficulty with sending and receiving signed and encrypted email as GnuPG plug-ins made it very straightforward. Only one expert user chose not to change the password but successfully completed the tasks.

To capture the effectiveness and satisfaction of interaction with SELS we asked users to fill out a SUS questionnaire. In study I, Experts gave TPSE and SELS SUS scores of 85.83 and 76.67 respectively on average while Novices gave SUS scores of 79.38 and 54.44 on average respectively. In study II, Experts gave TPSE and SELS SUS scores of 74.17 and 74.16 respectively on average while Novices gave 61.5 and 52. We believe that the key installation and management functions played a big role in the SUS scores that the software received though other factors such as prior familiarity with GnuPG have affected these scores. In order to account for such prejudices we look at the ratio of SELS SUS scores to TPSE SUS scores. In conducting the groupware walkthrough we realized that prior experience with GnuPG would help users in using SELS. Looking at the ratio of SUS scores between SELS and TPSE and comparing key installation success rates across SELS and TPSE will also help us better understand whether prior experience of GnuPG is necessary to use SELS. The ratio of SUS scores going from SELS to TPSE is 0.89 for experts on average, with a standard deviation of 0.1, and 0.68 for novices on average, with standard deviation of 0.18, in Study I. The average ratio of SELS SUS score to TPSE SUS score is 0.94 with standard deviation of 0.09 for experts and 0.84 with standard deviation of 0.13 for novices. We see that ratio of SELS and TPSE SUS scores for novices improves significantly in Study II when compared to Study I and is comparable to that of expert users. Furthermore, the same number of novices completed SELS key installation as TPSE key installation in Study II while few novices could complete SELS key installation in Study I. This indicates 1) that the key management techniques adopted in Study II are significantly better than those in Study I and 2) that if the key management tasks are defined well with adequate instructions then even novices can quickly learn to use SELS effectively.

Vulnerability.

After designing usable security features/software it is important to evaluate these usable features for vulnerabilities. Such an evaluation helps identify limitations of usable features as well as early opportunities to address security problems. In keeping with our usability design we conducted our vulnerability evaluation on the interface mechanisms, namely, the cues provided by interfaces. We asked users to make a trust decision on both correctly and incorrectly formed emails where an incorrect email was either encrypted incorrectly, signed incorrectly, signed correctly, or both. The results for parts three and five for both Study I and Study II are shown in Table 3. In these studies users received two correctly formed and four malformed emails from the studyadmin. For Study I, the table shows all experts trusted all correctly formed TPSE and SELS messages. All novices trusted all correctly formed SELS messages but some did not trust some TPSE messages (6.25% with a standard deviation of 17.68). We see interesting results in the case of malformed messages for both TPSE and SELS. An average of 16.67% malformed TPSE messages and 8.33% malformed SELS messages were trusted by experts. In the case of novices however the average increases to 18.75% for TPSE and 15.63% for SELS. For both experts and novices the most commonly trusted malformed message is the Encrypted and signed but with forged “From” message. This case required the user to look carefully at the email headers to come to a right decision. The slight decrease in average of malformed messages trusted, going from TPSE to SELS, is due to the fact that a few users who failed to detect the mismatch between the

4 Surprisingly, in Study II an expert user gave SELS a better SUS score than TPSE. In the exit interview he remarked "given that the interface cannot be changed SELS is very well integrated with the email client".
email headers and security banner displayed by the email client’s GnuPG plug-in, for the above mentioned malformed messages, did so for SELS.

For Study II, Table 4 shows that both novices and experts trusted all correctly formed messages. An average of 8.33% malformed TPSE messages and 16.67% malformed SELS messages were trusted by experts. In the case of novices however the average increases to 30% for TPSE and 35% for SELS. The increase in average going from TPSE to SELS in the case of experts is due to the fact that one expert user tended to trust unsigned messages that came or appeared to come on the list. However, the user wondered in his responses whether unsigned messages from the list should be trusted. The user even recommended in his comments that we mandate signed messages on the list.

In the case of novices the average of trusted malformed messages increased compared to Study I because apart from trusting Encrypted and signed but with forged “From” messages, some users tended to trust unsigned messages as long as the sender was known or a member of the list. Another observation made during Study II is that users that trusted encrypted but unsigned messages did not trust encrypted and signed messages if they could not verify the signature, i.e., they did not have and could not fetch the public-key of the sender. This is because the GnuPG plug-in for Thunderbird, the email client used by majority of users in our studies, alerted the users with a yellow banner that said “Unverified signature” whenever it could not verify the signature on a message. Whereas for encrypted-only messages it displayed a blue banner, as opposed to a green banner for an encrypted and signed message which was decrypted properly and whose signature was verified. The blue banner did not attract the user’s attention to the fact that message was unsigned and hence could be untrustworthy. This leads us to believe that most users would not have trusted Encrypted and signed but with forged “From” if the message signer’s key was unknown to the receiver. Thus making the above attack, which many experts did not detect, viable only as an “insider attack.”

Summary. Effectiveness of SELS is demonstrated by the fact that all novices in Study II were able to successfully install list keys, send and receive messages on the list, and trust correctly formed messages. An equally important measure of effectiveness is the vulnerabilities provided by SELS. While this number is greater than ideal (35% for novices), it is only slightly different from that for the TPSE evaluation indicating that SELS introduces minimal additional vulnerabilities, if any. Efficiency of SELS is demonstrated by the fact that novices were able to complete key installation in 18 minutes. The study also indicates that effectiveness and efficiency may improve with time as experts were able to successfully complete the key installation and send and receive messages in around 12 minutes and were more resistant to malformed messages (16.67%). In addition, many users were not familiar with the email client used in the study and we assume that gaining familiarity will help as well. Furthermore, user satisfaction for SELS, measured by the SUS score, was similar to that of the underlying email client and GnuPG plug-in combination (i.e., SELS SUS score to TPSE SUS score ratio is close to 1). This implies that users will find SELS almost as satisfying as secure email in general.

While we do not make changes to existing interfaces in SELS, our study recommends three modifications for GnuPG plug-ins to further improve the usability and security of SELS. We recommend that GnuPG plug-ins for email clients, 1) support caching of multiple passwords, 2) flag encrypted only emails as untrusted and 3) alert users when signer and sender do not match. The first will allow users to establish different passwords for each private key in a usable manner thus making the system more secure. The second will strongly recommend users to trust only signed messages coming on SELS (which was the primary problem for the slightly increased vulnerability measurement for SELS). Interestingly, this feature is already provided for the Mac Mail GnuPG plug-in, and we recommend that all email clients do so. The third will help decrease vulnerabilities for both two-party secure email and SELS.

4. SELS DEPLOYMENT EXPERIENCE

After a successful usability study the SELS software was hardened and installed in a production environment to support users. Our first user community was the TeraGrid Security Working Group (or simply, TG-WG) that has been using encrypted group email for several years to exchange sensitive information about vulnerabilities, incidents and recovery procedures for TeraGrid high-performance computing systems distributed across 11 sites. Until SELS came along they were using password-based symmetric-key encryption in PGP with the passwords being distributed in telephone conference calls. This approach required secure distribution and maintenance of multiple passwords (a security issue) and mapping passwords to current and prior emails (a usability issue). The community adopted SELS in the hope for a more usable and secure solution. In this section we describe our experiences in supporting the TG-WG over a ten-month period from January through October 2008. We describe major issues in areas of usability, trust and security, and software engineering that arose while supporting TG-WG and how they were resolved. Overall, SELS has been very successful in supporting TG-WG as evidenced by the large number of encrypted emails exchanged by the group.

<table>
<thead>
<tr>
<th>SELS Release</th>
<th>New Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5.5</td>
<td>New Trust Model, Decryption only user key pair.</td>
</tr>
<tr>
<td>0.5.8</td>
<td>Bounce Messages for wrong LK and HTML messages.</td>
</tr>
<tr>
<td>0.6</td>
<td>Two key lengths (1024 and 2048)</td>
</tr>
<tr>
<td>0.7</td>
<td>Generate and store Revocation Certificate for LK, Remove email address from User key pair.</td>
</tr>
<tr>
<td>1.0</td>
<td>Key Update, Delete a Subscriber, Mailman Patch, Improved error handling</td>
</tr>
</tbody>
</table>

4.1 SELS Usage Statistics

The List Moderator for TG-WG created two lists, we will refer to them in this paper as List-A and List-B, on the SELS List Server hosted at NCSA. List-A has 52 subscribers and an average of 32 encrypted emails were exchanged per month. List-B has 50 subscribers and an average of 2 encrypted emails were exchanged per month. This
4.2 Usability and Security

About 8 percent of the TG-WG list subscribers, that is 4 out of 52 needed support while installing keys and sending messages using SELS. Given prior GPG experience we deemed all TG-WG users to be “experts”. Therefore, this observation matches our usability study results in that most expert users were able to install keys and send messages. Most of these subscribers who had trouble were using an email client without a GnuPG plugin, e.g. Microsoft Entourage, but were able to import these keys via the command line. However, when they wanted to send a message they encrypted it using the GPG command-line interface and sent an email with the encrypted message as an attachment. Since SELS supports encrypted attachments only when using PGP/MIME their messages were dropped at the server. Improved dissemination of compatibility information (e.g., via FAQs) provided the needed resolution; in particular the need to paste the encrypted message in the body of the email for Entourage. Some of the subscribers who had trouble did not realize that they needed to install a separate set of keys for each list and hence had trouble sending and receiving email on one of the lists. Again, improved instructions via email and on the SELS web site by the SELS team resolved the issue. After the users installed the second set of keys they were successful.

While most of the TG list users were comfortable with installing and trusting SELS keys, some of them had concerns about setting “Ultimate” trust in the moderator’s key. We chose this mechanism to place trust in SELS keys as it was simple, i.e., required users to perform only one operation, and it enabled transitive trust in any key signed by the moderator without requiring any explicit action from users. In order to address this concern we adopted a recommenda-

<table>
<thead>
<tr>
<th>Bug Number</th>
<th>Description</th>
<th>Fix</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>Minor coding error in LM (Code Review)</td>
<td>Code fixed</td>
</tr>
<tr>
<td>17</td>
<td>Add a check to see if Java is installed correctly (Code Review)</td>
<td>Check added</td>
</tr>
<tr>
<td>18</td>
<td>Minor coding error in LM (Code Review)</td>
<td>Code fixed</td>
</tr>
<tr>
<td>19</td>
<td>Minor coding error in LM (Code Review)</td>
<td>Code fixed</td>
</tr>
<tr>
<td>20</td>
<td>GnuPG prompts different on Windows and “mix platforms. Add functionality to support both.</td>
<td>This fix was added. LM Code is platform independent again.</td>
</tr>
<tr>
<td>21</td>
<td>Fix file name syntax for revocation certificates on Windows.</td>
<td>This fix was added. LM Code is platform independent again.</td>
</tr>
<tr>
<td>22</td>
<td>Support bounce for HTML messages from Outlook 2007 and Outlook 2003 (User Feedback)</td>
<td>Bounce support added to LS code.</td>
</tr>
<tr>
<td>23</td>
<td>Race condition for GPG interactive command “—edit-key” used in LM code.</td>
<td>This bug was hard to fix in absence of a multi-platform expect like module for Python. So used a different approach, using Java, where GnuPG interactive commands are no longer required.</td>
</tr>
</tbody>
</table>


data has been collected since the TeraGrid lists were created 10 months ago. Prior to using SELS, TG-WG used password based symmetric key encryption. In 2006 and 2007, using that approach, the average number of encrypted emails exchanged per month on List-A was 9 and and on List-B was 3. Conversations with some of the list members indicated that the increase in number of messages on the lists was due to the ease of use with which members could exchange secure (encrypted) messages using SELS.

4.3 Features and Enhancements

SELS 0.5 was the first version that was used for TeraGrid lists. This version had the capability to bounce plaintext messages back to the sender if the list was configured to allow only encrypted or encrypted and signed messages. TeraGrid lists were initially configured to allow plaintext messages, but were changed to only allow encrypted messages a couple of months later, to avoid sending sensitive messages in plaintext by mistake. Over the last ten months SELS was enhanced with many features. Based on user feedback two new bounce messages were added for two cases in SELS re-
lease 0.5.8. SELS software supports messages sent in a text format or messages encrypted as PGP/MIME when sent in some other format. So if a message is composed as HTML and not sent as PGP/MIME, it will be dropped at the List Server. Similarly a message encrypted with an incorrect key is dropped at the server. These messages were being dropped at the SELS List Server without any notification to the sender which left the sender wondering where his/her message went. To improve usability, SELS was modified to always generate a bounce message whenever a message is dropped.

While SELS was being used by the TG-WG, the SELS Team felt the need to support additional clients based on users’ preferences.

1. GMail with FireGPG: SELS can be used easily with Gmail and the FireGPG plug-in.

2. PGP Desktop: Some TG-WG users use PGP Desktop as a key manager with email clients such as Outlook 2007 and Apple Mail with SELS. A few changes to the SELS code were needed to make SELS compatible with both PGP and GnuPG. One major change was to revert the encryption algorithm between release 0.7 and 1.0. SELS is back to using CAST5 instead of AES256 since CAST5 is better supported by PGP.

Some TG-WG users were using email clients that do not have available GnuPG plug-ins, e.g., Outlook 2007 and Entourage. However these users are able to use SELS by using the GPG command line for key management and encrypting or decrypting email messages and using Outlook 2007 or Entourage for sending and receiving messages to/from the List Server.

Other features added include support for automatically generating and storing a revocation certificate for the list key LK and functions that automate key updates for subscribers. In Table 5 we summarize new features developed in SELS since the TG-WG has been supported. We also undertook a software architecture and code review that helped us streamline our code and fix many bugs. In Table 6 we present major bug fixes undertaken during this time.

4.4 SELS Production Environment

The SELS production environment, shown in Figure 5, consists of primary and backup industrial-grade, rack mount host servers that support Linux virtual machines for each individual SELS instance. Both servers feature redundant power supplies (one of which is connected to an uninterruptible power supply) as well as hardware-based RAID mirrored disk drives. A monitoring script on a remote server watches both hosts for network connectivity, which indicates that the hosts are online and working properly.

A virtual machine (VM) is created for each instantiation of SELS, then replicated to the backup host. Each VM is configured with the hostname of the list, e.g., sels-example.ncsa.edu, but has its primary network connection set up with a different name, e.g., sels-example1 or sels-example2. This allows each VM to be addressed separately, but the common hostname ensures that the web interface to the Mailman list server works correctly when creating new email lists. Then on either the primary or the backup VM, the SELS list name (sels-example) is set up as a virtual Ethernet interface, i.e., eth0:0. This DNS configuration allows the backup VM to function equally well as the primary VM. However, in order for a transition to the backup to function properly at the software level, all changes to the primary VM are synchronized to the backup by a script that securely copies SELS software changes, Mailman list information, and new user data (user proxy-keys).

During the testing and pilot implementation phases of the project, several changes were made to the service infrastructure. Initially, all operating system software packages on the VMs were automatically updated. However, this caused a SELS failure (luckily prior to the pilot stage), so an automatic update configuration was modified to prevent any updates of specific packages, namely mailman and sendmail (which we use as the Mail Transfer Agent or MTA). Whenever updates to either of these two packages are available, a manual update is performed on the backup VM. If careful testing demonstrates that SELS functionality has been maintained, or after a fix to the package upgrade can be determined (usually a configuration file tweak), then the package update is rolled over to the primary VM. This periodic validation ensures that the backup VM can function properly as well as the primary SELS server. Initially, we manually modified mailman python files in order to include SELS functionality. However, by slightly modifying the SELS code, we were able to amend the mailman instance after it was upgraded via a simple patch script.

In the duration of supporting TG-WG, there have been no hardware failures, so no fail overs have been needed. We decided against using automatic fail overs to prevent the instance where both the primary and backup VMs are both listening to the email list IP address. Manual fail overs are precipitated when the remote monitoring script indicates that the primary VM has experienced a hardware failure or that network has been disconnected from the primary VM. One such instance did occur: a power glitch restarted the network switches, although the UPS protected the primary VM, which merrily hummed along. The temporary network outage caused the mailman daemon to stop processing emails properly, so it had to be restarted. To take this problem into account, the monitoring script now checks whether mailman is working correctly and restarts the daemon if this is not the case.

5. RELATED WORK

Secure Mailing Lists. Secure mailing lists have been used for a while in the US Department of Defense as part of the Defense Message System (DMS). This system uses enhanced S/MIME techniques presented in [10]. DMS satisfies the identified security properties of strong confidentiality, authentication and integrity. It uses a hardware lockbox at the server to provide strong confidentiality combined with an externally supported key distribution system. DMS uses digital signatures to provide authentication and integrity. Additionally, it uses an outer layer signature to provide authentication at LS for encrypted messages. However, to achieve these security properties DMS uses specialized email clients that can process messages to provide these properties and a hardware lockbox at the server. In contrast, SELS is a purely software based solution and imposes no additional burden on client-side software and only a software plugin on

[Details on DMS confidentiality property were provided by Stephen Kent of BBN in personal communications.]
proxies, again with the list server decrypting the emails. In these approaches, the sender manages the recipient list and addresses all of the intended recipients directly. However, the problem of sending confidential messages to multiple recipients has been addressed in the past via multi-recipient email encryption \cite{30}, multi-party certified email \cite{34}, secure group communication and broadcast encryption. A major difference between these approaches and ours is that by using a mailing list we remove the user's burden of managing recipient addresses and public keys while still satisfying the confidentiality requirement. In these approaches the sender must manage the recipient list and address all of the intended recipients directly. In multi-recipient email encryption, Wei et al. \cite{30} combine techniques from identity-based mediated RSA and re-encryption mixnets to enable a sender to encrypt messages to multiple recipients with only two encryptions (as opposed to one encryption for each recipient in the trivial case). To do so, they use a partially trusted demultiplexer that is akin to LS in terms of its security properties but also use an additional fully trusted CA. If their scheme were to be adapted for mailing lists it would require deliv-

Proxy Encryption. Previous proxy encryption schemes enable unidirectional and bidirectional proxy transformations by first setting up a transformation agent that is given the proxy key and then sending messages to the agent for transformation \cite{2,17,22}. Unidirectional schemes only allow transformations from some entity $A$ to another entity $B$ with a given proxy key while bidirectional schemes additionally allow transformations from $B$ to $A$ with the same proxy key. For SELS we need a proxy encryption scheme that allows for the transformation from one entity, $LS$, to many subscriber entities (i.e., to all list subscribers). The El Gamal based unidirectional proxy encryption scheme of Ivan and Dodis \cite{17} is closest in nature to SELS with the additional relationship between the proxy keys (i.e., $\forall i, K_{U_i} + K_{U_i} = K_{L_i}$) imposed to allow for a single list encryption key, $PK_{L_i}$, to suffice. Extending the RSA based unidirectional scheme of \cite{17} in a similar manner will not work because it would require the sharing of the modulus across all list subscribers. Jakobsson et al. \cite{15} and Zhou et al. \cite{35} allow for proxy transformation without the need for distributing proxy keys but use costly threshold crypto-systems to ensure the necessary security. Ateniese et al. \cite{1} and Green et al. \cite{15} and Canetti and Hohenberger \cite{2} extend proxy encryption schemes with useful properties such as non-interactivities, which for SELS might allow for generation of proxy keys without involving both $LM$'s and $LS$'s decryption keys. We believe that while deployment of applications based on these novel schemes faces challenges with infrastructure compatibility and lack of commonly available tools, it is an open problem to overcome these challenges. For example, our experiences suggest that users are unlikely to move to a different email client just to be able to use an advanced secure email solution. However, advanced future systems based on these schemes are likely to provide strong security guarantees and may prove to be very useful in practice.

Multi-recipient Email Encryption. The problem of sending confidential messages to multiple recipients has been addressed in the past via multi-recipient email encryption \cite{30}, multi-party certified email \cite{34}, secure group communication and broadcast encryption. A major difference between these approaches and ours is that by using a mailing list we remove the user's burden of managing recipient addresses and public keys while still satisfying the confidentiality requirement. In these approaches the sender must manage the recipient list and address all of the intended recipients directly. In multi-recipient email encryption, Wei et al. \cite{30} combine techniques from identity-based mediated RSA and re-encryption mixnets to enable a sender to encrypt messages to multiple recipients with only two encryptions (as opposed to one encryption for each recipient in the trivial case). To do so, they use a partially trusted demultiplexer that is akin to $LS$ in terms of its security properties but also use an additional fully trusted CA. If their scheme were to be adapted for mailing lists it would require deliv-

Usability Techniques employed in our study, namely, groupware walkthrough and focused user study with particular attention to skill level, are promising techniques that have not been fully applied to secure systems. Groupware walkthrough was proposed by Pinelle and Gutwin \cite{27} to allow for the inclusion of context in groupware usability evaluations and is based on the often used cognitive walkthrough technique. Contextual information such as dynamic nature of group work and variability of tasks for multiple concurrent users allows for the identification of usability problems that may not be possible with other techniques. Faulkner and Wick \cite{10} present an extensive analysis of the benefits of user studies that employ a mix of novice and expert users. In particular, they argue quantitative between-group comparisons offer exclusive insights into usability problems.

Usability of Secure Email has received considerable attention since the seminal work of Whitten and Tygar \cite{32}, which identified several shortcomings of PGP. Garfinkel et al. \cite{11,13} demonstrate the potential high success of digitally signed email in an e-commerce context with a user study. Gaw et al. \cite{14} study the social issues that affect the adoption of secure email. Garfinkel and Miller \cite{12} and Roth et al. \cite{28} explore alternative key management techniques that make secure email easier to adopt and they demonstrate the effectiveness of their techniques with user studies. However, to date all usability studies focus on secure two-party email exchange. To the best of our knowledge ours is the first study to focus on secure mailing lists. Furthermore, our study is significantly different in that we utilize usability techniques to improve secure email software usability without imposing software enhancement requirements.

Usability of Secure Email has received considerable attention since the seminal work of Whitten and Tygar \cite{32}, which identified several shortcomings of PGP. Garfinkel et al. \cite{11,13} demonstrate the potential high success of digitally signed email in an e-commerce context with a user study. Gaw et al. \cite{14} study the social issues that affect the adoption of secure email. Garfinkel and Miller \cite{12} and Roth et al. \cite{28} explore alternative key management techniques that make secure email easier to adopt and they demonstrate the effectiveness of their techniques with user studies. However, to date all usability studies focus on secure two-party email exchange. To the best of our knowledge ours is the first study to focus on secure mailing lists. Furthermore, our study is significantly different in that we utilize usability techniques to improve secure email software usability without imposing software enhancement requirements.

Usability Techniques employed in our study, namely, groupware walkthrough and focused user study with particular attention to skill level, are promising techniques that have not been fully applied to secure systems. Groupware walkthrough was proposed by Pinelle and Gutwin \cite{27} to allow for the inclusion of context in groupware usability evaluations and is based on the often used cognitive walkthrough technique. Contextual information such as dynamic nature of group work and variability of tasks for multiple concurrent users allows for the identification of usability problems that may not be possible with other techniques. Faulkner and Wick \cite{10} present an extensive analysis of the benefits of user studies that employ a mix of novice and expert users. In particular, they argue quantitative between-group comparisons offer exclusive insights into usability problems.
opment of client-specific plugins. In SELS the sender needs to execute only one encryption allowing compatibility with existing messaging formats and tools thereby avoiding the need to develop client-specific plugins. In multi-party certified email [34], the sender must maintain each recipient’s public key and encrypt the message individually to each recipient. This overhead is avoided in SELS via the use of mailing lists while still providing confidentiality.

In secure group communication either a trusted group controller (e.g., LKH [33]) distributes session keys to group members or the group members generate session keys in a distributed manner (e.g., TGDH [21]). In either case, list subscribers would have to maintain state on current session keys and update them on every membership change (whereas in SELS existing subscribers are not affected by the joins and leaves of other members). This makes the use of secure group communication techniques impractical for secure mailing lists as it goes against the nature of the largely offline email use. So-called “stateless” broadcast encryption schemes (e.g., [17], [3]) allow for encryption of messages to a dynamic set of group members without the members requiring to maintain state and executing key updates on membership changes. However, they vary the encryption key and cipher-text sizes depending on the group membership. This variation cannot be supported by today’s email standards making such solutions difficult to implement. SELS, on the other hand, addresses the confidentiality and deployability requirements of secure mailing lists in a practical way.

6. CONCLUSIONS

In this work we have described the process of going from the existing SELS prototype [20] to a usable and deployed software solution. We conducted an usability study whose high-level goals were to evaluate and enhance the usability (i.e., effectiveness, efficiency, and satisfaction) of the SELS key management system for list subscribers with the strong preference that the solution be compatible with commonly used email clients. We have deployed SELS and report on our experiences in supporting the TeraGrid Security Working Group for a period of ten months. Success of SELS is clearly indicated by the fact that there has been a nearly four-fold increase in the number of encrypted emails exchanged by the TeraGrid users since they adopted SELS with anecdotal evidence suggesting that this increase is largely due to better usability provided by SELS. The SELS software is now available for community use. We look forward to continuing to support and improve the software based on input from the user community.

7. ACKNOWLEDGMENTS

The authors would like to thank James Marsteller and Tim Brooks for facilitating the adoption of SELS by the members of TeraGrid Security Working Group. We thank Pooja Agarwal for helping with unit testing and code review and all the users that participated in the usability evaluation. We also thank Von Welch for helpful discussions through the course of this effort. This work is funded by Office of Naval Research under Grant Nos. N00014-06-1-1108 and N00014-07-1-1173. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Office of Naval Research.

8. REFERENCES


