SaTC: CORE: Small: Usable Key Management and Forward Secrecy for Secure Email

1 Introduction

Since its creation in the early days of the ARPANET, email has become a fundamental means of communication. As of 2017, there were 6.3 billion email accounts, owned by 3.7 billion users, sending over 269 billion email messages per day.[1] Yet, despite its popularity and ubiquity, email was created without security in mind and remains largely insecure today [20, 21, 32, 26]. Spam and malware is prevalent, though filtered by many email providers. Phishing and spearphishing remain problems, even impacting the 2016 U.S. presidential race. There is only limited protection from passive and active network attacks, as well as message forgery. Email archives are extensive, typically stored in plaintext, and vulnerable to hacking.

The technology needed to deploy secure email is well studied and has been available for years. However, to date, no secure email system has been widely used by the general public [38]. We call this the *secure email adoption problem*, and it is the primary focus of this proposal. Recent work indicates that users do not adopt software for a variety of reasons, including lack of concern, lack of perceived need, misconceptions about how to obtain security, and poor awareness of effective security practices [40, 1]. Two other primary factors have plagued secure email—deployability and usability. Email has for years had a well established, rich set of heterogeneous applications, on a variety of platforms, and these must all interoperate. Deploying secure email solutions in this environment is challenging. Meanwhile, the secure email software offered to the masses—PGP—has been plagued by a significant usability gap [64, 55, 45]. Users typically are unable to manage encryption keys, including (a) how to discover and validate public keys, (b) how to securely move a private key from one device to another, (c) how to keep their private key safe, and (d) how to backup and recover from lost private keys.

A natural question to ask is why secure email continues to flounder when secure messaging has seen such great success in recent years. WhatsApp and Facebook Messenger have over a billion users, and the market supports numerous other secure messaging applications with millions of users, among them Signal, Telegram, Line, and Viber. These applications typically use variants of off-the-record messaging [10], such as Signal [39], to provide both end-to-end encryption and forward secrecy. What keeps secure email from following a similar technical approach and adopting a variant of off-the-record messaging? What has kept secure email from likewise building communities that scale to billions of users?

We contend that the service expectations for email substantially complicate the choices for an email-based security architecture. Secure messaging applications use a centralized key server to exchange public keys, and each application only has to serve users of that application. Applying this architecture to email would encounter significant problems. First, because email is provided by a large, federated set of providers, it would be unlikely that they would agree to run a centralized key server with a single authority to manage it. Instead, a distributed system of key servers would be needed, along with a form of auditing [33] to ensure that providers were honest. Second, even with such a system in place, email is an open system; the expectation that anyone can email anyone

[1] From the Email Statics Report, 2017-2021, by The Radicati Group, `https://www.radicati.com/wp/wp-content/uploads/2017/01/Email-Statistics-Report-2017-2021-Executive-Summary.pdf`

else directly leads to the need for spam and malware filtering, along with protection from phishing. A system of key servers would allow any spammer to send encrypted malware to any user, bypassing a service provider's filters. The technology for filtering on encrypted data is not ready yet. Finally, users typically archive email, often for years, and a significant use case is search over those archives. This makes it challenging to provide encryption for webmail or for mobile clients, since in both cases email is typically stored on a service provider's infrastructure, where access to plaintext is currently needed for these features.

Given its history, email is likely to continue to be a primary form of communication on the Internet for years to come, so we should not give up on this problem. Rather, new approaches are needed to make progress on this pressing issue. Our approach is guided by three key insights. First, the best path to adoption of a secure system is to take an already usable system and then build as much security as possible into it [60]. Second, making PGP usable means abandoning manual key exchanges and the web-of-trust model in favor of automatic key distribution and authentication [6]. Third, for the reasons described above, we must move beyond key servers to new methods of key distribution that have a greater chance of adoption.

Accordingly, we propose research centered in the following areas:

- *Adoption Factors:* We will survey users to identify factors that will increase adoption. We propose to frame our questions within a risk communication framework, which has been effective in the areas of health and disaster preparedness. Using this framework, we will study user perceptions of email security threats, factors that increase user perception of the efficacy of secure email software, and factors that encourage propagation among communication partners.

- *Key Discovery and Authentication:* We will develop an adoptable and usable key discovery and authentication service for PGP by starting with trust-on-first-use (TOFU) key exchange, followed by a variety of methods that increase trust in the exchanged keys. Starting with TOFU key exchange prioritizes deployability and usability, while additional methods can increase security, for example by using a public ledger. Keys will be exchanged only with regular or approved contacts, so that it is not easy for spammers to deliver encrypted spam and malware to large numbers of users.

- *Key Storage and Portability:* We will develop usable key storage software, including key portability and backup. It is easy to automatically generate key pairs, but users need to be able to transfer a private key to other devices they use, and need backup and recovery in case a device with the private key is lost. We will explore a range of methods, including purely local storage, encrypted cloud storage, a smartphone app, and hardware tokens. For backup we will evaluate a variety of methods to help users store a master key in a secure location, without requiring technical expertise to manage the key. These methods may include storing the key on a USB drive or printing a QR code on paper. We expect that a variety of solutions will appeal to different types of users and use cases.

- *Forward Secrecy and Message Expiration:* We will develop forward secrecy for secure email, using an key ratcheting mechanism, such as used in the Signal Protocol [39] or puncturable encryption [24]. The main challenge is making the protocol transparent to the user, so that forward secrecy can be provided without any additional key management tasks incurred by the user. We will explore helping users avoid storing sensitive messages long term in their email providers by expiring messages and message keys.

- *Usability Studies:* We will conduct numerous usability studies to validate the adoptability and usability of each of these components of secure email. Our initial studies will be conducted in a laboratory environment, utilizing a novel methodology we have developed that tests whether two novice users can adopt a secure email system without help from an outside expert. Toward the end of the project we will conduct long-term usability studies with the best-of-class systems we have developed to understand how people integrate secure email into their everyday email habits.

For the software development tasks in this proposal, we will build on the MessageGuard platform we have developed [51]. MessageGuard integrates secure email into existing webmail providers such as Gmail and Yahoo Mail, a preference expressed by many users in our studies. We have previously used MessageGuard to develop a variety of secure email systems. A key feature of MessageGuard is its plugin interface, which allows us to easily extend it to include new key distribution and authentication methods. Using a common platform also allows us to easily compare different extensions of the system while maintaining a consistent UI and user experience, removing any confounding variables. We have already released MessageGuard as open source software, and plan to continue supporting it as a platform for secure email research system for the usable security community.

2 Related Work

Recent research related to our proposal includes measurement studies that analyze the use of TLS to encrypt email during transmission, usability studies of secure email systems, and forward secrecy of instant messaging applications.

2.1 Measurement Studies

A number of papers [20, 21, 32, 25] have measured the level of adoption and effectiveness of the authentication and encryption methods used by email providers. These papers conduct a variety of active measurements of email servers and passive measurements of ongoing email traffic. The general picture they paint is that top email providers encrypt messages with STARTTLS and use SPF and DKIM for authentication, but there is a long tail of organizations that are lagging in deploying these mechanisms.

Even when security solutions are deployed, they are often compromised by insecure practices. Of those servers that offer encryption, many use self-signed certificates, expired certificates, or broken chains, all of which cause the validation of the certificate to fail. There are numerous other cases in which email traffic uses weak cipher suites, weak cryptographic primitives and parameters, weak keys, or plain authentication over unencrypted connections. Stripping attacks can compromise STARTTLS, with Durumeric et al. [20] illustrating how these attacks lead to 20% of inbound Gmail messages being sent in cleartext for seven countries. Use of SPF is common, but enforcement is limited, and DNS records aren't protected with DNSSEC. There is little use of DKIM, and few servers reject invalid DKIM signatures [21]. It is evident that transport layer security alone is insufficient to protect sensitive email messages in transit.

2.2 Usability

In 1999, in their seminal paper on usable security, Whitten and Tygar conducted a usability study of PGP 5.0 [64]. It served as a wake-up call to the security community because a large percentage of

users failed to complete basic tasks installing and using a state-of-the-art secure email tool. In their study, 9 of the 12 users (75%) mistakenly sent messages unencrypted, and several even included their private key in their emails. Moreover, at the conclusion of the study, Whitten and Tygar found that participants were unclear as to how PGP protected their email.

Garfinkel and Miller [23] and Sheng et al. [55] replicated the work of Whitten and Tygar. Garfinkel and Miller showed that automatic key management was more usable than the manual key management from the original experiment. However, the study revealed that the tool "was a little too transparent" regarding its integration with Outlook Express. As a result, some users failed to read the instructions associated with visual indicators. Sheng at al. demonstrated that despite improvements made to PGP in the seven years since Whitten and Tygar's original publication, key management was still a challenge for users. Furthermore, they showed that in the new version of PGP, encryption and decryption had become so transparent that users were unsure if a message they received had been encrypted.

Recent research has continued to look at automatic and transparent encryption to better understand user preferences. In our work, we conducted a series of user studies with Private WebMail (Pwm), a secure email prototype that tightly integrates with the Gmail web interface [47]. Even though results showed the system to be quite usable, we found that some users made mistakes and were hesitant to trust the system due to the transparency of its automatic encryption. In a replication of this work, Atwater et al. [5] verified that participants responded positively to automatic key management. They created a mock-up of Mailvelope wherein a user's PGP key pair is generated automatically upon installation and shared automatically using a key server. A usability study found that it performed far better than previous PGP-based email encryption software. However, the mockup used by Atwater et al. did not simulate the delay that happens in practice when a sender has to wait for the recipient to generate and publish their public key, ignoring a significant impediment inherent to PGP-based systems.

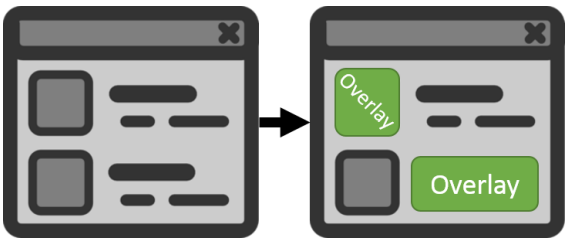
Bai et al. explored user attitudes towards different models for obtaining a recipient's public key in PGP [6]. In their study, they built two PGP-based secure email systems, one that used the traditional key exchange model (web of trust) and one that used a registration model based on a key directory. Users were provided with instructions on how to use each tool and given several tasks to complete. Afterwards, participants shared their opinions regarding the key exchange models. The results of this study showed that, overall, individuals preferred the key directory-based registration model, though they were not averse to the traditional key exchange model.

Lerner et al. [29] designed Confidante, a standalone email client that sends PGP-formatted encrypted messages through the users Gmail account. Key management relies on Keybase,[2] a key management system that includes a public key directory that links public keys to social media accounts. The paper reports on a user study with journalists and lawyers, showing that Confidante was easier to use than Mailvelope. There were concerns that Confidante required signing up with an extra service, which could hinder adoption, and highlighted continuing concerns about the balance between automation and trust.

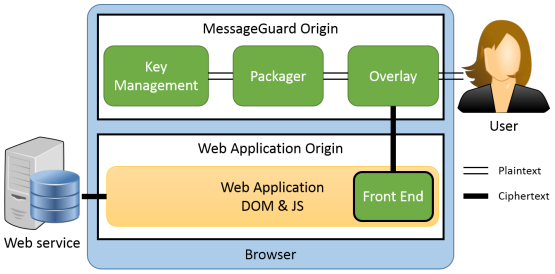
2.3 Forward Secrecy

PGP lacks forward secrecy, a property that prevents an attacker who has compromised long-term key material from being able to decrypt previously sent messages. In 2002, Brown et al. [12] proposed PGP extensions that utilize encryption keys with short lifetimes to limit the window of vulnerability. In 2004, Borisov et al. [10] proposed an alternative protocol to PGP called off-the-record messaging (OTR) that provided forward secrecy and deniability. Their work primarily

[2] `https://keybase.io`



(a) Secure overlays: Portions of the web application (left) have been overlayed with secure interfaces (right).



(b) Architecture: Plaintext is present only within the overlay, preventing the web application from ever accessing it.

Figure 1: MessageGuard

targeted instant messaging applications, but they proposed extensions to the protocol that would allow it to work with email while maintaining forward secrecy. However, OTR is not readily suited for application to email because the protocol requires that both participants be online during the key exchange phase. Marlinspike [31] further points out that OTR was not designed with asynchronous transports (e.g. SMTP) in mind—the protocol assumes that messages will be delivered in order, but most asynchronous transports, such as SMTP, can make no such guarantee.

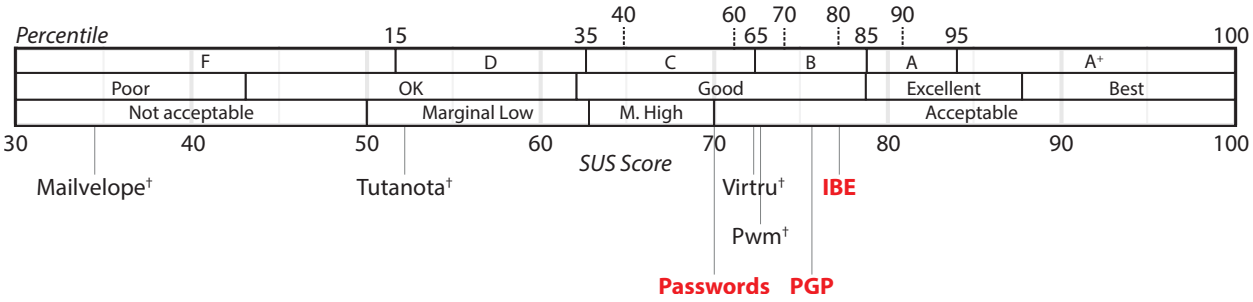
The Signal Protocol, developed by Open Whisper Systems, was developed particularly for asynchronous messaging. It has been incorporated into popular mobile messaging apps, including Signal, WhatsApp, Facebook Messenger, and Google Allo to achieve end-to-end encryption. There has been little attention given to using the Signal Protocol to improve the security of email. An open source project, Pond, used the Signal Protocol to implement an email-like system offering end-to-end encryption; however, the Pond system was incompatible with existing SMTP systems, and the project has since ceased development. Other similar projects have not seen wide adoption, possibly because of their failure to integrate closely with existing email services.

Recently, Green and Miers [24] published a paper introducing Puncturable Encryption, a novel approach to forward secure encryption. Puncturable Encryption allows a user to update their decryption key such that it cannot decrypt messages before a certain date. An interesting property of this system is that it does not require redistribution of keys after the decryption key is updated. This system achieves forward-secure messaging and only adds low overhead in the process.

3 Preliminary Work

We have completed significant preliminary work that serves as a foundation for the proposed research. This work uses a platform we have developed, called MessageGuard [51], which integrates secure email into existing webmail providers. MessageGuard operates as a browser extension, with secure overlays, shown in Figure 1a, that encrypt sensitive data without leaking it to the webmail provider. MessageGuard is designed to support A/B tests of key management alternatives by having a consistent UI that only differs according to key management features. This reduces confounding factors when comparing key management features. Figure 1b shows how MessageGuard uses a component-based architecture that enables us to plug in a variety of trust models and key management systems.

The design and implementation of MessageGuard was guided by seven IRB-approved laboratory usability studies involving a total of 220 participants [42, 47, 43, 44]. Several of the design choices for MessageGuard, which were established in these studies, include (1) using email composition



† These SUS scores are for other secure email systems tested with the same methodology [46, 43], and are reference points for the performance of our systems. Mailvelope is a PGP-based system, Tutanota a password-based system, Pwm an IBE-based system, and Virtru a custom key escrow scheme.

Figure 2: SUS Scores for Passwords, PGP, and IBE variations of MessageGuard

interfaces to show users which emails are encrypted, which helps them avoid mistakenly sending sensitive information in the clear; (2) including contextual clues to help users understand how to use secure email correctly; and (3) implementing inline, context-sensitive tutorials to instruct users. We cover several of our more recent results below.

Two-person user studies. Our recent usability study comparing Pwm (an early version of MessageGuard), Tutanota, and Virtru [43] received an Honorable Mention award at CHI 2016. The study introduced a novel methodology where pairs of novice users are recruited to participate in the user studies. Conducting studies in this manner more realistically simulates grassroots adoption scenarios. This methodology varies compared to traditional user studies of secure email where a single participant works alone or interacts with a study coordinator. Participants reported being more at ease with this type of study and reported being better able to cope with mistakes since both participants were "on the same page". We prefer this methodology in secure messaging studies we conduct because it provides us with both the perspective of the user sending the secure message and receiving the secure message. We expect to continue using this methodology in many of the studies in our proposed work.

Usable PGP Our most recent paper involved applying lessons learned from earlier systems to develop a usable version of PGP. We initially conducted a two-person study of a modern PGP client, Mailvelope, that is a browser extension supporting PGP. We brought in ten pairs of novice users and gave them a task to exchange secure email messages. Only one of the ten pairs was able to successfully exchange an email message after one hour of effort [46]. Using insights from this study, we designed a PGP implementation in MessageGuard that automatically generates key pairs and automatically distributes public keys using a centralized key server. Our implementation also has integrated tutorials, an approachable description of public key cryptography, and automatically-generated invitations to install MessageGuard when receiving an encrypted message.

We conducted a study comparing PGP with IBE and password-based encryption, using MessageGuard's A/B testing capabilities, so that these different trust models had the minimum possible user interface differences. Figure 2 illustrates the SUS scores for the systems tested in the study, using a scale developed from a large number of systems tested with SUS [52].[3] Each of these key management implementations received higher usability scores than other secure email systems

[3]The percentiles are for all applications, not just security applications.

that support those key management approaches. The PGP version resulted in a usability score that rivaled the IBE (key escrow) system, and is the first fully-functioning PGP key management implementation to have received high usability scores in a laboratory user study.

Short-lived Keys Most recently, we conducted the first usability study of short-lived keys for secure email [34]. Participants from our recent secure email user studies had expressed a need to use secure email tools only a few times a year. We had also observed users expressing concerns over the permanence of personal information on the Internet. Support for short-lived keys has the potential to address both of these problems. However, the short-lived keys usability and security space is underdeveloped and unexplored. Early work From Schneier and Hall [53] and Brown et al. [13] explored this space, but stopped short of an implementation. Later work on OTR and Signal has only been adopted by secure messaging systems.

We designed and implemented both a short-lived keys and a long-term keys secure email prototype and conducted a laboratory usability study where users experienced both prototypes. Users indicated that they trusted the short-lived keys prototype more and liked the option of having their sensitive messages become unreadable. Participants generally expressed a desire to control when emails expire and were split on whether messages should automatically become unreadable after an expiration period. This initial research informs our interest in pursuing these issues in more depth.

4 Proposed Research

Building on our preliminary work to design usable secure email, we propose to design and evaluate key management solutions for users to successfully manage public and private keys. We will explore this primarily in the context of PGP, with users generating and maintaining a long-term key pair, though our research results can be equally applicable to S/MIME with self-signed keys. Our key management methods will essentially replace the manual key exchange and the web of trust for something much more usable.

Our proposed research will continue development within the MessageGuard platform because it integrates secure email into webmail systems. Users have expressed a strong preference for secure email that works with the email systems they already use. However, we believe the research results we generate will be equally applicable to desktop systems, since usability for key management plagues all secure email systems, regardless of platform. In addition, we believe the key management results will apply to key management for popular instant messaging systems.

A important aspect of our research is to focus on solving the secure email adoption problem. We will examine adoption within a well-known framework known as the Extended Parallel Processing Model, which indicates that certain criteria must be met before people will adopt a new behavior—they must perceive a risk as susceptible and of sufficient severity, they must identify a solution as effective in mitigating risk, and they must have confidence that they can successfully use the solution. We will incorporate this model in both an initial survey for our first task and in every usability study we conduct, to explore how the design of key management solutions can encourage users to adopt secure email in their everyday tasks.

The remainder of this section describes our research plan to:

- conduct a formative survey to identify user attitudes that affect the adoption of secure email;
- develop usable key discovery and authentication methods that start with trust-on-first-use exchange of public keys, then add additional security through stronger methods;

- develop methods for usable key storage, including portability among different devices, and backup and recovery;
- integrate forward secrecy and message expiration into our secure email platform; and
- conduct numerous studies to validate the adoptability and usability of each of these components of secure email.

4.1 Adoption Factors

In this task, we will conduct a survey of user attitudes that affect the adoption of secure email. We plan to use ideas from the field of risk communication to frame our survey questions. Camp [16] has noted that risk communication is an established and practiced field that has not been drawn upon as a resource for the study of how to execute computer security risk communication. Stewart and Lacey [56] studied the propagation of security advice in the context of risk communication theory, indicating that current attempts to fill the knowledge gap are inappropriately treated with a "broadcast of facts" that fails to take into account either audience or existing knowledge. They introduced three concepts from other disciplines which they suggest would be beneficial if applied to information security: bounded rationality, mental modeling, and the Extended Parallel Processing Model (EPPM) [65, 66, 30].

EPPM is a model from risk communication theory that characterizes the conditions that must be met in users' minds if they are to act on provided information. It is widely used in the domains of public health and disaster awareness and prevention. The EPPM defines four key elements that influence individuals' responses to risk, with two factors that correspond to the perception of risk, and two that correspond with the efficacy of the response. Perception of risk includes both the individual's perception of the likelihood that a given risk (e.g. account hijack) will occur, and their perception of the severity of that risk. Efficacy includes both the individual's perception of the efficacy of a certain response (e.g. two factor authentication) and their confidence that they can successfully use this strategy or solution (known as self-efficacy). The EPPM thus indicates how security researchers and developers can construct effective solutions: as long as perceptions of efficacy are stronger than perceptions of the risk, users will take the recommended action to avoid the risk.

We have existing work in progress that is using the EPPM to broadly study mental models that users have when they interact with encryption. In this task we will complement that work by formulating a survey to evaluate how users perceive risk and efficacy with respect to secure email. We anticipate using the Mechanical Turk platform to survey U.S. residents on these issues. A preliminary set of issues we will investigate include:

Risk Perception: If risk perception is very low, the users are unlikely to see the need for adopting any response to the threats they face. We want to survey users about their perception of the likelihood and severity of various threats that affect their use of email. These may include phishing, account hijack, data mining by email providers, and so forth. In cases where users are lacking information on the likelihood or severity of threats, we want to identify messages that can help users better assess threats.

Response Efficacy: If response efficacy is low, then users are unlikely to feel any solutions are capable of mitigating the threats they face. We want to survey users on the efficacy of a variety of secure email solutions to alleviate these threats. If messages from their regular correspondents are

signed, does this help alleviate phishing attacks? Does encryption help alleviate fears of account hijack or data mining? In addition, issues related to how secure email is presented to users may affect perceptions of efficacy. For example, users could encrypt individual messages (or threads) individually, or the system could automatically encrypt messages once keys are discovered. Secure email messages could be integrated into the user's normal inbox or separated into their own *private conversations* folder. Secure messages could retain the full features of email or be presented more like instant messaging (i.e. a single thread per conversation partner).

Propagation: A key element of efficacy includes how users encourage their communication partners to participate in adopting a secure tool. We anticipate the software will send enrollment emails or invitations to regular correspondents who are not currently using the software. Presentation of these invitations are critical – what message elements will convince a recipient this is trustworthy? Will invitations be tolerated or be considered a turn-off? What kind of platform, such as a browser extension or mobile client, would recipients be likely to install or use? Webmail depot?

4.2 Key Discovery and Authentication

In this task, we focus on discovering and authenticating keys for a recipient of secure email. Building a global PKI that enables every user to discover and verify the public key of every other user is a very difficult problem. It is unlikely that one solution will provide seamless interoperability for all of the billions of email users and for the diverse uses of email. Furthermore, no single party controls the email ecosystem, and widespread deployment of secure email needs the cooperation of numerous stakeholders.

Realistically, however, most people have relatively small numbers of correspondents that they interact with – their regular contacts and occasional new contacts. In the latter cases, people are more likely to contact someone who is well known (e.g. a government representative, a journalist), or to whom they have been introduced (e.g. friend-of-a-friend, through employment, or chance encounters through mutual interests) rather than a complete stranger.

The strongest advances in this area have been made by secure messaging applications, such as WhatsApp, Signal, and Telegram, which automate key discovery by using a centralized server. However, this approach is not a good fit for email. A key server lets anyone discover the public key that matches a given email address (or phone number). For email, this would enable anyone to send unsolicited, encrypted email that bypasses content filters operated by the email provider and potentially delivers a phishing attack or malware.

Based on these observations, we identify two important research questions. First, how can we automate key distribution and authentication for email, without requiring public disclosure of all keys and without relying on trusted third parties? Second, how can we provide key discovery that works for a wider variety of use cases?

The solution we plan to investigate is bootstrapping encryption with regular and approved correspondents using a trust-on-first-use (TOFU) key exchange, with ratcheted trust. This model works as follows:

- *Automatic signing:* An email client automatically signs outgoing messages to a user's frequent contacts, including the user's public encryption key. Upon receiving email, clients collect and track the public encryption keys. Users could manually initiate secure communication with a new contact by sending an introductory email and asking the email client to include a signature.

- *Automatic encryption:* When an email client regularly receives the same public encryption key from a given user, then it is able to initiate encrypted emails to that person. The client could do this automatically or could present an encryption option to the user. The client can also allow the user to initiate encryption with a new contact.
- *Ratcheting trust:* The email client adds additional methods when available, such as checking for a DKIM signature on the email, consulting a privacy-preserving key server [33], or examining a public ledger [28, 4].

Clients may incorporate multiple methods of key distribution and authentication, then use indicators for different trust levels, such as *yellow/secure* for TOFU, then *green/trusted* after comparing fingerprints. Similar proposals have been made by PGP developers recently,[4] but no usability or adoption studies have been performed yet to evaluate the effectiveness of this approach. In addition, there are numerous possibilities for ratcheting trust that must likewise be evaluated.

One advantage of this approach is that public keys are only sent to frequent contacts (or approved new contacts on demand) and are not publicly discoverable, so users are not exposed to a flood of encrypted malware. Another advantage is that it prioritizes usability first, to encourage adoption, then increases security where possible. This mirrors the approach used in secure messaging applications, such as Signal, which prioritize usability first (using a key server) and then ratchet trust later (using an authentication ceremony [62]. The downside of this approach is that users will need help when keys change [22]. In particular, users may be susceptible to attacks if they must manually verify keys whenever they change [54, 57, 19]. Methods for ratcheting trust may help eliminate this problem by relying more on automation in these cases.

This approach enables us to experiment with a wide variety of systems for ratcheting trust. For example, we are interested in exploring designs for privacy-preserving key servers, which would enable us to solve the problem of encrypted malware, while also strengthening trust in exchanged keys. Many other methods for ratcheting trust are also possible. DKIM signatures are increasingly being deployed and could be used to indicate the strength of a signature if the DKIM hash includes the public key (which could be in a header or in the email body). A client could publish a public key in a public ledger such as Blockstack [4], for additional verification. A client could check whether the recipient belongs to keybase and retrieve keys and proof of ownership from the keybase servers, similar to Confidante [29].

To cope with lack of search on encrypted archives, we will explore separating secure email into its own folder of *private conversations*. This could help establish a separate "mode" for secure email as compared to standard email, with different service expectations. We will also explore using different presentation for private conversations, such as a single thread per correspondent, no subject header, no carbon copies, etc, so that this mode is clearly distinct from unencrypted conversations.

4.3 Key Storage and Portability

In this task, we focus on providing key storage and portability. To use public key cryptography effectively, users need methods to securely store their keys, without risk of losing their private key, while ensuring that their keys are portable across all their devices. The MessageGuard software automatically generates a keypair for the user when it is installed. However, if the user moves to a new device, the keypair needs to be transferred safely. The user should not need to be aware of the details of public key cryptography to accomplish this task.

[4] See autocrypt (`http://autocrypt.readthedocs.io/en/latest/`), LEAP (`https://leap.se/`), and PEP (`https://pep-project.org/`)

We will design and develop usable key storage software for MessageGuard that explores the design space of fully automated solutions to alternatives that give the user greater control and choice in how keys are stored. We will explore a range of methods, including purely local storage with secure sharing among devices, encrypted cloud storage using a master password, storage on a smartphone, and use of a hardware token such as Yubikey. Several of these methods were recommended by Garfinkel in 2005 [22] but have not been explored by the usable security community.

We will use usability studies to explore the tradeoffs of these options, concerning both storage and portability among devices, and how this affects efficacy for users. For example, both local and cloud storage will require using a master password to encrypt the keys. There are longstanding challenges helping users to establish strong passwords that they can also remember [2, 67]. However, alternatives to passwords typically don't retain the benefits they offer and have yet to displace passwords as the most commonly used mechanism [9]. Using a smartphone or hardware token, on the other hand, imposes different tradeoffs. These solutions require a user to keep the item in physical proximity to the device using their keys, which renders attacks by remote users infeasible but also imposes a burden on users.

We will also explore the related issue of backup and restoration, to handle the case when a user loses their key material. This could be forgetting a master password for cloud or local storage, having a cell phone stolen, or misplacing a hardware token. We will compare a variety of methods that help users work with an offline master keypair, generating subkeys that can be given to authorized devices. If a users phone is lost, she can generate a new subkey for a replacement phone using the master key. We will evaluate a variety of methods to help users store a master key in a secure location, without requiring technical expertise to manage the key. These methods may include storing the key on a USB drive, printing a QR code on paper, or other methods. We expect that a variety of solutions will appeal to different types of users and use cases.

4.4 Forward Secrecy and Message Expiration

Traditional secure email systems do not offer forward secrecy—a compromise of the user's private key enables an attacker to decrypt all previously encrypted messages. We will identify and analyze alternatives for adding forward secrecy to secure email. The novel aspect of our research will be to identify the usability impact of incorporating forward secrecy into secure email. We plan to determine what actions, decisions, and cues are necessary to impact the user experience positively.

We will survey and analyze forward secrecy protocols [17] to determine which are suitable for adoption into the asynchronous nature of email. We have identified several initial candidates. First, we will explore the Double Ratchet used in the Signal Protocol [39]. Second, Green and Miers [24] introduced puncturable encryption for asynchronous messaging. Their work outlines potential approaches for integrating the protocol into legacy email systems. The result of our study will be a comparison of the security and usability properties of various alternatives. We will identify necessary changes to support forward secrecy in email. For example, we may need to include metadata along with each message. We will build a prototype using MessageGuard to increase our understanding and to utilize in usability studies.

Forward secrecy will open up new approaches for secure email that mimic the properties of current instant messaging applications. Based on our prior user studies, we find that some users are resistant to maintaining sensitive information indefinitely on an email server. We will explore ways to provide short-lived email messages via the management and deletion of ephemeral encryption keys. The short-lived paradigm is a significant departure for email compared to the typical design of email as a permanent archive. The primary challenge is to design a usable email variant supporting forward secrecy that does not require additional key management tasks. One intriguing possibility

is to automatically organize secure email in an "Private Conversations" folder that is separate from the regular inbox, with an encrypted conversation for each contact.

One way to represent short-lived email messages to users is through message expiration times. Tungare has sketched out some design ideas for email expiration times [59], but little work has been done evaluating expiration times for email and how this interacts with short-lived keys. Here we will build on our preliminary work, which indicated users strongly prefer this feature and want control over expiration times [34]. One of the difficulties with fine-grained control over messages expiration times is that this does not scale easily when there are large numbers of secure messages, as opposed to a small set of messages seen in a typical lab study. The efficacy of this feature may also be impacted by how secure email is presented. For example, if secure emails are presented in a format that looks more like secure messaging threads, then users may be more comfortable with old messages expiring. Moreover, if expiration is presented as a privacy feature, similar to Snapchat but for email, then this may increase confidence in allowing messages to automatically expire. We will explore a variety of ways for users to exert control, such as setting default expiration times, or cancelling expiration for high priority messages, while maintaining scalability.

For users that desire long-term storage of encrypted messages, we will explore the use of separate keys for long-term storage from the ephemeral keys providing forward secrecy of transmissions. This distinction follows security recommendation 7-3 from NIST's recent publication on Trustworthy Email [18] that suggests the use of PGP and S/MIME for encrypting messages in transit only, and the use of user-controlled keys for long-term storage of sensitive information. We will be the first to design and conduct usability studies that focus on this recommendation.

4.5 Evaluation

The usable security community has often focused exclusively on the criteria of usability to evaluate secure email solutions. We will ground our evaluation in the EPPM model, encompassing both risk perception and efficacy. Our goal will be to ensure that participants view MessageGuard as effective in mitigating threats they may encounter, while also helping them feel confident that they can be effective in using MessageGuard.

Our evaluation methods will include both qualitative and quantitative measures. The qualitative measures will be based on interviews with participants regarding their perception of risk (both susceptibility and severity) and efficacy (both solution efficacy and self-efficacy). We will assess qualitative results using standard open coding practices borrowed from grounded theory, building from codes to concepts to themes [27]. The quantitative measures will examine frequency of mistakes and a measure of the usability of the software using the System Usability Scale (SUS) developed by Brooke [11, 7]. We select SUS as our standard usability metric because it has an established track record in the usability community and we have evidence that it is reliable across different sets of participants [47]. SUS has been used in hundreds of usability studies [8] and has been shown to give the most reliable results as compared to other measures [58].

We will conduct two kinds of user studies as detailed below. Before conducting any studies with human subjects, we will obtain IRB approval for our protocol. PIs Seamons and Zappala have extensive experience obtaining IRB approval for several recent projects, and Dr. Seamons has been conducting usability research with IRB approval for the past nine years.

Lab usability studies involving two novice users: Once software is ready for user evaluation, we will use lab studies with novice users. We have pioneered an approach where two users are brought into the lab together and asked to evaluate communications software [46, 62]. This study format has the benefit of allowing us to gather data for different types of first-use cases (i.e., sending

a secure email first vs. receiving a secure email first), since the two users assume different roles. Typically, we investigate user experience with installation and use of the software, gathering both qualitative and quantitative measures relevant to adoption and usability, as described above.

Longitudinal user studies: Once lab studies indicate a secure email solution is usable and likely to be adopted, we will conduct longitudinal studies to evaluate adoption and user behavior "in the wild". Design of this study requires some care. We want to be able to determine if a user will adopt MessageGuard purely for its benefits. One way to do this is to recruit a set of people to use MessageGuard for a period of one month. When the recruited users wish to correspond securely with someone, the system could generate an invitation email that contains instructions on how to install the tool, with instructions to read the secure email on a depot if they choose not to install the tool. This would enable us to study adoption of MessageGuard without forcing its installation. Pending IRB approval, we could also test automatically generating invitation emails to frequent correspondents. We will also release MessageGuard on extension "stores" for browsers, to see if there is natural adoption outside of the enrolled participants.

We are also interested in using a longitudinal study to collect a variety of data. We can instrument MessageGuard to collect telemetry data, indicating how often it was adopted, how often it was used for sending secure emails, and so forth. At the conclusion of the study we can provide an optional survey for all users of MessageGuard to gather their feedback. We could also provide participants in the study with a monthly privacy report, indicating how many of their messages were protected by MessageGuard. With IRB and user approval, MessageGuard could also report on the most frequently used words in their plaintext emails, or show likely advertisements that might be shown, given the contents of their emails. It would be interesting to see if privacy reports would affect adoption.

4.6 Research Schedule

The proposed research will contribute to one Ph.D. dissertation and one M.S. thesis. The following outlines the major research activities during each year of the proposal.

Year 1

1. *Adoption Factors:* Conduct a survey using Mechanical Turk (United States participants only), to assess risk perception, response efficacy, and adoption issues as outlined in Section 4.1.
2. *Key discovery and validation*
   - Design and implement PGP plug-in that uses TOFU to bootstrap encryption with regular email partners.
   - Analysis of DKIM signatures, public ledger, privacy-preserving key server, and other methods for ratcheting trust, and implementation of selected options.
3. *Lab Usability Studies:* 50 users each
   - Evaluate the adoptability and usability of TOFU encryption.
   - Compare methods for ratcheting trust

Year 2

1. *Key storage and backup*
   - Analysis of key storage options and implementation of selection options

- Analysis of key backup and recovery options and implementation of selected options

2. *Lab Usability Studies:* 50 users each
   - Compare private key storage options
   - Compare key backup and recovery options
   - Evaluate entire system for resistance to attacks
3. *Longitudinal Study:* Study long-term adoption and use, 1 month, 50 users.

Year 3

1. *Forward Secrecy and Message Expiration*
   - Analysis of forward secrecy options and implementation of selected option.
   - Analysis of and implementation of message expiration
2. *Lab Usability Studies:* 50 users each
   - Evaluation of forward secrecy
   - Evaluation of message expiration options
3. *Longitudinal Study:* Refine tool based on feedback from initial longitudinal study, then repeat study for long-term adoption and use, 1 month, 50 users.

5 Broader Impacts of the Proposed Work

The broader impact of this research falls into three categories: benefit to society, benefit to the security community, and impact on education.

Benefit to Society Success and progress on this important research problem has high potential to transform online communication by all citizens since it is specifically targeted towards novice users. The proposed research will benefit national security by providing all citizens with an easy way to protect their online email communications. The principles and lessons learned on this project have the potential to reach beyond just email systems, and can impact other forms of online communication. For instance, usable key management advances developed in this research have the potential to impact key management for popular instant messaging applications like WhatsApp.

Benefit to the Security Community The proposed work contributes to the security community by developing MessageGuard as an open source platform for deploying and testing key management systems for secure communication. This enables researchers in the field to avoid the burden of building an extension that integrates with common webmail providers and focus on new key management research. In particular, the architecture of MessageGuard enables A/B testing of key management alternatives, which has been difficult to do in the past because of significant differences between email clients for the various approaches to implementing secure email.

Impact on Undergraduate and Graduate Education The proposed work will contribute to **education** through the broad dissemination of results through research publications, graduate student mentoring, and graduate courses at BYU.

The PIs teach graduate seminars in usability, as well as computer and network security. The proposed research will enhance their future graduate seminars. All course materials will be made

available on the Web as a resource to other educators and students. The proposed work will form the basis for one Ph.D. dissertation and one M.S. thesis for the graduate students supported under this grant, thereby increasing the nation's supply of highly trained security experts.

The proposed research will also enhance undergraduate education at BYU. In 2012, the NSA and DHS designated BYU a National Center of Academic Excellence in Information Assurance Education (CAE/IAE) for the years 2012 through 2017. PI Seamons is responsible for security courses in the Computer Science Department at BYU, and for enhancing security topics across the curriculum. PI Zappala teaches an web development course that includes exposure to web security technology and attacks. Currently, there are no available resources for teaching the forward secrecy properties of the Signal Protocol that is used in popular instant messaging applications. PI Seamons will create a module on forward secrecy for use in his undergraduate security course. Currently, there is no available textbook or educational material that explains the Signal Protocol that has been incorporated into popular instant messaging applications. He will make this publicly available for others to use.

The PIs' research groups have a strong track record of involving undergraduate students in research, which will increase the pool of students trained in cybersecurity. Supplementary funding for undergraduate research will be obtained from BYU, and experience in research will strongly influence students to pursue graduate education or careers in security. A number of previous undergraduates who participated in research with the PIs have pursued graduate education as a direct result of external funding supporting their research.

6 Prior NSF Work

The PIs have conducted prior research funded by the NSF.

TWC: Small: Middleware for Certificate-Based Authentication, CNS-1528022, 2015–2018, $496,900. Kent Seamons: PI, Daniel Zappala: co-PI. The focus of this project is to repair and strengthen TLS certificate authentication. **Intellectual Merit:** We have developed an operating system approach to transparently overriding broken certificate validation for all TLS connections on a system. Our approach provides greater application coverage and stronger security guarantees than competing approaches. The system also provides plugins that support easy experimentation and adoption of alternatives to the CA system. Publications from this project include [49, 36, 3, 61, 35, 62, 48, 37, 50]. **Broader Impact:** We have created TrustBase, a middleware platform for researchers to experiment with new, novel authentication alternatives to the CA system.

NeTS Small: WiFu: A Software Toolkit for Wireless Transport Protocols, CNS #0917240, $298,216, 2009-2013. Daniel Zappala, BYU, PI.

This work addressed the problem of providing reliable transport across a multi-hop wireless network. **Intellectual Merit:** The PI led a team that designed and built WiFu, an open source toolkit for developing experimental wireless transport protocols [14]. WiFu provides for user-space development of reliable transport and rate control algorithms, greatly simplifying the implementation effort required. We used WiFu to evaluate several wireless transport protocols and found that their simulation results were not matched by experimental data [14, 68]. We also designed several new rate control models, one based on an optimization framework [63] and another based on first principles [41]. **Broader Impact:** The WiFu toolkit is available as an open source project on GitHub [15]. A new graduate course was taught based on mesh networking, with materials for running a mesh network in an indoor, university environment posted on the Web.