SoK: Securing Email—A Stakeholder-Based Analysis

Abstract—**While email is the most ubiquitous and interoperable form of online communication today, it was not conceived with strong security guarantees, and the ensuing security enhancements are, by contrast, lacking in both ubiquity and interoperability. This situation motivates our research. We begin by identifying a variety of stakeholders who have an interest in the current email system and in efforts to provide secure solutions. We then use the tussle among stakeholders to explain the evolution of fragmented secure email solutions undertaken by industry, academia, and independent developers. We develop an evaluation framework for proposed or deployed secure email systems and identify how well they meet properties related to security, utility, deployability, and usability. We conclude with a fresh look at the state of secure email and discuss open problems in the area.**

Index Terms—**email, PGP, etc.**

1. Introduction

Email has been called *"probably the most valuable service on the Internet"* [1]. For its over 50-year history, email has been an open medium—if you know someone's email address, you can send email to them [2]—with seamless interoperability among users with a diverse range of desktop, mobile, and web client software. As an indication of its near-universal acceptance, an email address is often required to create online accounts and to make online purchases. As of 2019, there were an estimated 3.9 billion users of email sending over 293 billion email messages per day [3].

Despite its ubiquity, email was created without security in mind and remains largely insecure today [4], [5], [6], [7]. Email is only sometimes transmitted over an encrypted connection, with limited protection from passive network eavesdropping and active network attacks. Email can be easily forged, and spam and malware are prevalent, though filtered by many email providers. Email archives are typically extensive, stored in plaintext, and vulnerable to hacking. Phishing and spear phishing [8] remain problems.

Typically, it is believed that greater security could be provided by integrating end-to-end encryption and digital signatures into email systems. Deployment of these technologies in the form of S/MIME is widespread, with many popular tools such as Microsoft Outlook and Apple Mail supporting it. Still adoption of these S/MIME-enabled tools is largely limited to when there is significant motivation to protect intellectual property (enterprises) or conform with regulatory burdens (government). These products tend to be used more frequently in larger enterprises or governments, where IT support staff are available to help with administration, and non-enterprise users have fewer choices. PGP has long been championed as secure email for the masses but has been plagued by lack of adoption and severe usability issues surrounding key management [9], [10]. Services such as ProtonMail and Tutanota have filled this gap with a more usable webmail and mobile product, providing automated encryption between users of the same service. However, most people still use plaintext email services and even where end-to-end encryption systems are adopted, they are rarely interoperable.

Not only is ubiquitous adoption of end-to-end encrypted email elusive, none seems evident on the horizon. This is partly due to the openness and service expectations of email. For example, the fact that anyone can receive email from anyone else leads to a need to filter out spam, malware, and phishing attacks, solutions that today require the filtering service to have access to plaintext emails. Likewise, archive and search make it challenging to provide encryption for webmail or for mobile clients, since in both cases email is typically stored and indexed on a service provider's servers. Finally, actions such as acquiring public keys for recipients can be done almost transparently in a single organization, but complications arise when expecting interoperability with countless unrelated and uncoordinated organizations worldwide.

To better understand the current state of affairs and identify where future research and development efforts should focus, we conduct a stakeholder-based analysis of secure email systems. First, we identify a set of stakeholders with differing goals and show how the actions and interests of these groups helps to explain the history of failures and successes in secure email, leading to the current patchwork of partial secure email solutions. Next, based on the needs of these stakeholders we broaden the discussion of secure email beyond encryption and signing to include other security properties, as well as consideration of utility, deployment, and usability properties. We then rank the importance of each of the properties to each stakeholder and rank a representative set of secure email systems based on how well they meet these properties. This allows us to identify incompatibilities between stakeholders, illustrate how different solutions have evolved to meet their needs, and show which stakeholders are under-served. We conclude by discussing how this view of secure email can help shape future research and development efforts.

2. Background

To provide some context for our discussion of efforts to secure email, we first describe how email works and explain why it is insecure.

2.1. How Email Works

A series of protocols are used to send email, transfer it from the sender's email provider to the recipient's provider, and then retrieve it. Figure 1 shows the most

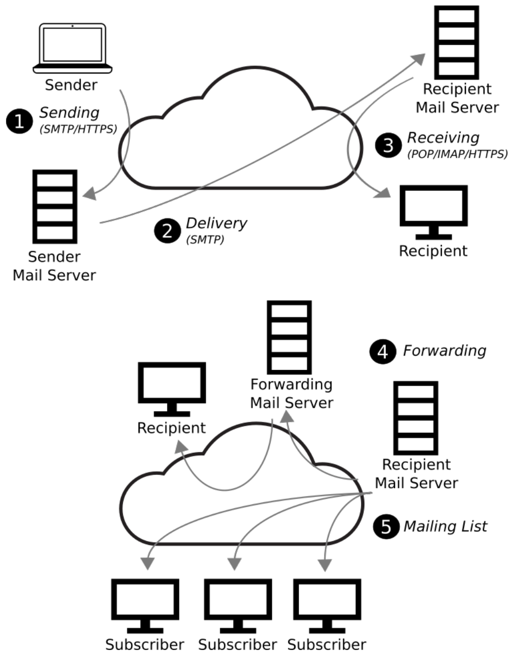

Figure 1. Overview of email operation and protocols. (1) Sending email generally uses SMTP or HTTPS between a client and its mail server. (2) Delivery of email between mail servers uses SMTP. (3) Receiving email generally uses POP, IMAP, or HTTPS. (4) A recipient mail server may forward email to another server. (5) A recipient mail server may forward an incoming email to a set of mailing list subscribers.

basic steps involved, in steps marked (1) through (3). When a user initiates sending an email, her client may use SMTP [11] to submit the message to her organization's mail server (also called a mail transfer agent or MTA). The sender's MTA uses DNS to locate the mail MTA for the recipient's domain, then uses SMTP to transfer the message. Finally, the recipient retrieves the message from her own organization's MTA, possibly using POP or IMAP. If either the sender or receiver is using webmail, then step (1) or step (3) may use HTTPS instead. Note also that the version of SMTP used to submit a message in step (1) is modified from the version of SMTP used to transfer messages [12].

This sequence of events is complicated somewhat by additional features supported by email, shown by steps (4) and (5) in Figure 1. First, a receiving MTA can be configured to forward email for a recipient on to another MTA; *e.g.,* forwarding email from `bob@company.org` to `bob@gmail.com`. This can repeat an arbitrary number of times. Second, a destination email address may correspond to a mailing list server which forwards the email to all subscribers on the list (a potentially large number). This adds numerous other recipient MTAs to the process.

2.2. Why Email is Insecure

Every aspect of email was initially designed, specified, and developed without consideration for security. As a result, these initial designs led to numerous problems, many of which persist today despite decades of work to fix them. Consider one example: as originally designed, there is no authenticity for email messages. This means anyone could send email to anyone else, using a forged sender email address.

To see how this is possible, it is useful to distinguish between the two parts of an email message: the envelope and the body. The envelope contains SMTP commands that direct MTAs regarding how the message should be delivered. In particular, the envelope specifies the sender's email address (`MAIL FROM`) and the recipient's email address (`RCPT TO`). The message body has a separate format [13], including the familiar *From*, *To*, *CC*, and *Subject* header fields. Email clients generally display the sender's email address shown in the *From* header in the body, rather than the one in the SMTP envelope. The original specifications contain nothing that validates the `MAIL FROM` command or prevents forgery of the *From* header. Combined, these features meant that anyone could forge an email to anyone else, leading to the emergence of what we call unsolicited email or spam [14] and, later, phishing and delivery of malware.

In addition to lacking authenticity, the original designs of protocols used to send, receive, and deliver email among clients and servers contained no protections for integrity or confidentiality. All messages were transmitted in the clear and could be intercepted and modified by anyone able to act as a man-in-the-middle. Email is also subject to privacy violations because it does not easily provide anonymity, deniability, freedom from tracing, and ephemerality.

3. Stakeholders

Our systematization of secure email centers around a set of key stakeholders that have either an interest in the current email ecosystem or efforts to deploy secure email. We identified these stakeholders by reviewing the research literature; reading online posts, discussion threads, and news articles regarding secure email; and by looking at press releases and features provided by secure email tools. While other groupings of stakeholders are possible, we believe our list covers the space of user and business interests and is ideal for demonstrating the tussle that exists between stakeholders, helping explain the history of research and development in this area and the lack of a universal solution. In Table 1 we briefly describe the stakeholders we have identified and in §4–§10 we examine their priorities as a foundation of our stakeholder-based analysis.

We note here that governments do not cleanly fit into any one stakeholder. Government services are provided by departments that need to communicate securely with each other, and we group them with enterprise organizations for this purpose. Some government functions are focused on national security and law enforcement, in which case end-to-end encrypted email can pose hurdles to investigations, and we count this function of government as a separate stakeholder. When governments send personnel into the field, they may need secure communications outside of the internal network, and in this situation their priorities align well with privacy-focused users. Likewise, when governments deal with informants or undercover operations, their priorities align with vulnerable users.

It is also important to recognize that not all governments are alike. Some align strongly with privacy advocates, such as when the European Union enacted strong privacy regulations with the General Data Protection Regulation (GDPR) in 2016. Others have looser regulations

TABLE 1. Stakeholders with an interest in email and secure email

Stakeholder	Description
Email Service Providers	Organizations that provide email services to industry and the general public
Enterprise Organizations	Large organizations in both government and industry
Privacy Enthusiasts	Users with strong privacy preferences who believe email should offer strong protection from corporate or government surveillance
Vulnerable Users	Users who deal with strongly sensitive information that could induce personal safety risks, including journalists, dissidents, whistleblowers, informants, and undercover agents; we also include criminals in this category
Secure Email Providers	Organizations that provide secure email services to the general public, such as ProtonMail or Tutanota
Typical Users	Users of standard, plaintext email services
Law Enforcement	National security organizations and state and federal law enforcement

that align more strongly with national security concerns, such as the U.S. Electronic Communications Privacy Act (ECPA) of 1986, which has been interpreted to mean that email stored online for over 180 days is abandoned and accessible without a warrant. There are many countries with no freedom of speech and thus a strong alignment with national security concerns that prioritize access to plaintext communications.

4. Email Service Providers

Email service providers are focused on meeting the needs of the typical users, including both business and personal use. Providers often require access to plaintext so they can scan incoming emails for spam and malware.

Providers have adopted several technologies to improve the security of email, including link encryption, domain authentication, and sender authentication. They have also invested significant effort in spam and malware mitigation, but we consider these areas outside of the scope of our analysis. We review current and planned efforts, the protection they offer, and measurements that reveal that these technologies have had a large variance of effectiveness to date.

4.1. Link Encryption

Email providers have adopted methods for encrypting email while it is in transit between MTAs or between an MTA and a client. We call this *link encryption* to distinguish it from end-to-end encryption, which occurs between the email sender and recipient. Link encryption is designed to prevent eavesdropping and tampering by third parties that may own untrusted routers along the path that email is being delivered [15].

By default, mail sent using SMTP is sent in the clear, but the connection may be upgraded to use TLS with the STARTTLS command [15]. Because this is negotiated in plaintext, it is trivial for an adversary to mount a downgrade attack by corrupting or stripping the STARTTLS command [4].

To fix this problem, a March 2016 Internet draft introduced Strict Transport Security for SMTP [16]. With STS, email providers may use DNS records to advertise a policy requiring TLS connections for SMTP. A receiving MTA can authenticate the policy it receives (ensuring it is valid for the sending domain) using a variety of mechanisms, such as with the Certification Authority system or with DANE [17], [18]. Sending MTAs can report on or refuse to deliver messages that cannot be delivered using TLS.

For communication between mail client and an MTA, a variety of encryption methods can be used. A client using POP or IMAP can start encryption using STARTTLS [19]. However, many servers are now disabling plain IMAP and POP and instead requiring clients to connect over TLS. Clients using webmail can use HTTPS to provide encryption between the recipient MTA and the client. Clients may also submit email using SMTP and the STARTTLS command.

One additional concern is that SMTP reveals significant metadata as email messages are relayed, such as sender and recipient email addresses. Some work tries to address this concern [20], [21], but no significant work beyond draft stages exists as of this writing.

4.2. Domain Authentication

Email providers have invested significant effort in providing *domain authentication*, which ensures that an email originated from a specific domain. Consider the case when Alice receives an email from `bob@gmail.com`. Domain authentication indicates that the email was sent by a server authorized to send email from `gmail.com`. This is contrasted with *sender authentication*, §4.3, which occurs when `gmail.com` authenticates `bob` to access and send email. A third step is *user authentication*, which occurs when Alice ensures that a human, such as Bob Smith owns the `bob@gmail.com` account.

Domain authentication has a long history rooted in identifying spam and filtering malware [14], [22], [23], [24]. These authentication methods provide assurance that the originating domain of an email, either as listed in the body or the envelope, actually sent the email.

4.2.1. DKIM. DomainKeys Identified Mail (DKIM) [25], [26] allows a server that originates email to include a digital signature over some portions of the email. The signature can include the entire message (header and body) or just selected fields in the header, but signatures are not applied to the envelope. The sending server hashes the fields to be signed and digitally signs the hash with the private key of the sender's domain. The sending server also inserts a header in the body of the email, indicating which domain should be used for checking the signature. A receiving email provider uses this header to determine the domain, retrieves the public key of the sending provider via DNS, then verifies the signature. Assuming email providers use DNSSEC [27] to ensure that public keys received via DNS are legitimate, this provides assurance that the sending email provider did originate the email and that the signed fields have not been modified in transit.

4.2.2. SPF. Sender Policy Framework (SPF) [14] allows an organization to publish a DNS record that specifies which IP addresses are allowed to originate email for their domain. A receiving provider can verify that the IP address of the server that originates the email is on the list of approved addresses for the domain in the envelope MAIL FROM field. These IP addresses do not necessarily need to be owned by the domain; they can, for example, be IP addresses from a separate email provider. SPF breaks mail forwarding, but an MTA can instead use remailing, which changes the envelope sender domain to match that of the forwarder.

4.2.3. DMARC. Domain Message Authentication, Reporting, and Conformance (DMARC) [23] builds on these technologies by allowing an organization to publish a DNS record indicating which of these services (DKIM, SPF) they support, along with a policy indicating what action should be taken if authentication fails. DMARC also requires identifier alignment. For SPF, this means that the domain in the envelope `MAIL FROM` address (which is authenticated with SPF) must match the domain in the body *From* address. For DKIM, this means that the domain used for signing must match the domain in the body *From* address. This links the authentication or signature verification done by SPF and DKIM to the *From* address seen by the user. DMARC also provides a mechanism for receiving email providers to send reports back to sending email providers regarding the outcomes of DMARC policy enforcement. This helps email providers identify misconfigurations and abuse.

4.2.4. ARC. Authenticated Received Chain (ARC) [24], [28] extends email authentication to handle cases when messages are potentially modified and then forwarded, such as by a mailing list. With ARC, a forwarder adds headers indicating the status of validity checks done on authentication added by the originator or other intermediate forwarder (*i.e.,* SPF, DKIM, DMARC, or ARC). The forwarder then adds a signature over the message body and header as it received it, then adds another signature (called a *seal*) over any ARC fields added by this or previous forwarders. When there are multiple forwarders involved, the set of ARC fields added forms an ARC chain.

4.3. Sender Authentication

The methods discussed above authenticate only the sending domain and do not guarantee that the sending user was authenticated by the domain. Most email domains do authenticate their users [29]. For example, if the sender is using webmail, then she may authenticate by logging into her webmail account over HTTPS. If the sender is using a desktop client, the mail domain can authenticate her with SMTP Authentication, which provides several methods that enable the sender to authenticate with the MTA by a username and password [30], [31], [32]. However, the measures a domain uses to authenticate a sender are not communicated to the recipient of an email message, nor can they be verified by the recipient. Sender authentication does not offer the same properties as digital signatures, and thus can't provide user authentication.

4.4. Measurement Studies of Adoption and Effectiveness

Several papers [4], [5], [6], [7] have measured the level of adoption and effectiveness of the encryption and domain authentication used by email providers. 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. However, even when security solutions are deployed, they are often compromised by insecure practices, such as self-signed certificates[1], expired certificates, or broken chains, all of which cause the validation of the certificate to fail. Email traffic often uses weak cipher suites, weak cryptographic primitives and parameters, weak keys, or password authentication over unencrypted connections. Of the techniques that rely on DNS, basic attacks such as DNS hijacking, dangling DNS pointers [33], and modifying non-DNSSEC lookups can enable circumvention. Stripping attacks can compromise STARTTLS, with Durumeric et al. [4] illustrating how these attacks caused 20% of inbound Gmail messages to be sent in cleartext for seven countries. As Mayer et al. [6] conclude, *"the global email system provides some protection against passive eavesdropping, limited protection against unprivileged peer message forgery, and no protection against active network-based attacks."*

5. Enterprise Organizations

Email security for enterprises often prioritizes scanning incoming email for malware and ensuring outgoing email does not reveal company intellectual property and conforms with regulations protecting consumer privacy. However, some enterprises deploy end-to-end encryption and digital signatures for internal communication, allowing them to exchange secure email without revealing plaintext to third parties.

Enterprises played a role in developing standards that could meet their needs, starting with PEM [34], [35], [36], [37], [38], and then leading to S/MIME. PEM was distinguished by having a hierarchical trust model with a single root CA and centralized revocation, largely precluding rogue certificate issues haunting later PKI systems.

5.1. S/MIME

S/MIME [39] is a standards suite for securing MIME data with both encryption and digital signatures. It was originally developed during the early 1990s by RSA Data Security, then later adopted by the IETF, resulting in standards in 1999 [40], [39], [41]. S/MIME has wide support on major platforms and products, such as IBM (Lotus) Notes and Microsoft Outlook mail clients [42, p.60-62].

[1] With the advent of free domain certificates with Let's Encrypt, it is possible that more providers are using verifiable certificates since these measurements were conducted in 2015—2016.

S/MIME clients use *certificate directories* to look up X.509v3 certificates associated with a given email address, which are generally issued by third-party Certification Authorities (CAs) operating under a centralized trust model, with CAs also responsible for certificate revocation information. X.509v3 extensions define information such as whether a certificate holds the public key of a CA or a non-CA entity (*e.g.,* useful for properly verifying certificate chains), and what the key may be used for (*e.g.,* signatures vs. encryption). S/MIME does not mandate a hierarchy with a single root CA—any organization can act as an independent, trusted root for its certificates, and this is its most common usage today. Enterprises often use *private key escrow* in conjunction with S/MIME, which enables the organization to decrypt emails and scan them for spam or malware, comply with regulations that require access to plaintext email, and provide recovery if a client loses its private key. Recent industry initiatives to facilitate interoperability between key management clients and key management servers are being advanced by the OASIS standards body with the Key Management Interoperability Protocol (KMIP) [43].

The centralized certificate management supported by S/MIME is a good match for internal communication within enterprises and governments. This has led to some large deployments of S/MIME in commercial or government silos [44]. Companies using S/MIME typically retain access to private keys in order to maintain access to all emails, to comply with regulations, to scan email for spam or malware, or to detect fraud or insider trading. Several works have examined usability deficiencies with S/MIME implementations, noting difficulties knowing which CAs to trust [45], difficulties with certificate management [46], and inconsistency in handling certificates [42, p.60–67]. Automatically creating and distributing signing and encryption keys at account creation is considered good practice [47].

5.2. Hosted S/MIME

Because running an S/MIME system involves deployment overhead, some providers offer hosted S/MIME [48], in which an organization uploads private keys to an email provider, and the provider automatically uses S/MIME for some emails (*e.g.,* to other users of the same provider). Encryption in this case is only *provider-to-provider*, rather than end-to-end. As noted by Garfinkel and Miller [49], individuals do not have the benefit of a dedicated IT staff, so it is a burden to manage their own public key pairs and certificates. Hosted S/MIME, with providers managing keys for individual users, could provide a solution for this problem, if users are willing to trust their providers.

6. Privacy Enthusiasts

Privacy enthusiasts prefer end-to-end encrypted email in order to avoid government surveillance or commercial use of their data generally. They differ from vulnerable users in that there is not an immediate personal safety risk driving their usage of secure email. Privacy enthusiasts have historically favored PGP, which was developed as "public key cryptography for the masses" and "guerrilla cryptography" to counter authorities [50].

6.1. PGP

PGP's history is a fascinating 25-year tale of controversy, architectural zig-zags, name ambiguity, and patent disputes, with changes in algorithms, formats and functionality; commercial vs. non-commercial products; corporate brand ownership; and circumvention of U.S. crypto export controls.[2] The current standard for the PGP message format is OpenPGP [53], [54], a patent-unencumbered variation of PGP.

Despite evolving formats or encryption algorithms, PGP enthusiasts until recently have largely remained faithful to PGP's distinguishing concepts:

- **PGP key packets and lightweight certificates:** PGP key packets hold bare keys (public or private). Public keys are kept in *lightweight certificates* (*cf.* [50]), which are not signed certificates in the X.509 sense, but instead contain keys and a User ID, in the form of a username and email address. To help client software determine which keys to trust, PGP also includes *transferable public keys* [53], which include one or more *User ID packets* each followed by zero or more *signature packets*. The latter attest the signing party's belief that the public key belongs to the user denoted by the User ID.
- **PGP's web of trust:** The web of trust is a model in which users personally decide whether to trust public keys of other users, which may be acquired through personal exchanges or from public servers, and which may be endorsed by other users they explicitly designate to be *trusted introducers* [55].
- **PGP key packet servers:** Users publish their public key to either closed or publicly accessible key packet servers, which contain a mapping of email address to the public key. Clients query to locate the public key associated with an email address.

Users typically store private keys on their local device, often encrypted with a password, though hardware tokens are also available.

6.2. Problems with PGP

PGP's design around the web of trust has allowed quick deployment in small groups without bureaucracy or costs of formal Certification Authorities [56], but leads to other significant obstacles:

- **Scalability beyond small groups:** Zimmerman notes [50, p.23] that *"PGP was originally designed to handle small personal keyrings"*. Scaling PGP requires acquiring large numbers of keys, along with a manual trust decision for each key, plus manual management of key storage and the key lifecycle.
- **Design failure to address revocation:** Zimmermann writes [50, p.31], *"If your secret key is ever compromised...you just have to spread the word and hope everyone hears about it"*. PGP does have methods to revoke keys, but distribution of these to others is ad hoc.

[2] PGP was distributed as freeware on the Internet in 1991, leading to an investigation of Zimmermann by the United States Customs Office for allegedly violating U.S. export laws. He published the PGP source code in book form in 1995 [51], and the case was subsequently dropped in 1996 [52].

- **Usability by non-technical users:** Zimmerman [50, p.31] says *"PGP is for people who prefer to pack their own parachutes"*. There is no system help or recovery if users fail to back up their private key or forget their passphrase. Furthermore, users must understand the nuances of generating and storing keys, trusting public keys, endorsing a public key for other users, and designating others as trusted introducers. The poor usability of PGP has received significant attention [9], [10].
- **Trust model mismatch:** Zimmerman notes [50, p.25] that *"PGP tends to emphasize [an] organic decentralized non-institutional approach"* reflecting personal social interaction rather than organizational relationships. The PGP web of trust was designed to model social interaction, rather than decision-making processes in governments and large enterprises. It is thus not a one-size-fits-all trust model.

6.3. Abandoning the Web of Trust

A variety of efforts have explored improving the usability of PKI for email, primarily focusing on automating discovery of public keys. While much of this work has been done in the context of PGP, the work applies to S/MIME as well, since either the centralized infrastructure (S/MIME) or the web of trust (PGP) could be augmented or replaced with the methods we discuss here. All of this work abandons the web of trust around which PGP was originally designed.

6.3.1. Trusted public key servers. Recently, the usable security community has studied automated use of trusted public key servers. Recent work [57], [58] showed that automated trusted public key servers have high usability when integrated into a PGP-based email system. Bai et al. [59] found users prefer key servers to manual key exchange, even after being taught about the security limitations of a key server.

6.3.2. Audited public key servers. An alternative to trusting public key servers is to build infrastructure that enables public key servers to be audited. A typical way to do this is to publish certificates in a manner that makes them visible to other parties, allowing monitors to examine a history of all certificates or key packets that a key server has made available for any entity, allowing them to detect rogue certificates or keys and server equivocation [60], [61].

6.3.3. Trust-on-first-use (TOFU). Another approach is to exchange keys in-band and have clients trust them on first use. This has been the subject of several research projects [62], [49], [63]. Since 2016, the developer community has been integrating TOFU into PGP implementations in the MailPile, PEP [64], LEAP [21], and Autocrypt [65] projects. A common critique of TOFU is that users cannot distinguish valid key changes from an attack. Recent work by developers in the PEP and LEAP projects is aiming to address this problem with additional methods to authenticate public encryption keys, such as using a trusted public key server, auditing public key servers, and asking the user to compare key fingerprints [66], [67].

6.3.4. Identity-based encryption. Identity-based encryption (IBE) [68] uses a trusted server to store a master private key and generate individual private keys for users. The trusted server also advertises a master public key, which clients can use to derive a public key for any email address. Users can validate their ownership of an email address with the IBE server in order to retrieve their generated private key. IBE simplifies key management for clients but also substantially complicates revocation [69]. Ruoti et al. [70], [71] integrated IBE into a webmail system, demonstrating how automating interactions with key management results in successful task completion and positive user feedback.

6.3.5. Social Authentication. Another way to disseminate public keys is to associate them with public social media accounts. The Keybase project helps users to post a signed, cryptographic proof to their account, simultaneously demonstrating ownership of a public key and ownership of the account. By aggregating proofs across multiple social media accounts for the same person, a client can establish evidence that ties a public key to an online persona, under the assumption that it is unlikely that a person's social media accounts are all compromised simultaneously. The Confidante email system [72] leverages Keybase for distribution of encryption public keys, with a study finding it was usable for lawyers and journalists.

6.3.6. Short-lived keys and forward secrecy. Schneier and Hall [73] explored the use of short-term private keys to minimize the damage resulting from the compromise of a private key. Brown and Laurie [74] discuss timeliness in destroying a short-lived key and how short-lived keys complicate usability by requiring more frequent key dissemination. Off-the-Record Communication [75] expanded on this vision with a protocol that provides forward secrecy on instant messaging platforms and can be applied to encrypt all but the initial email message sent between two parties. This work led to the double ratchet algorithm used in Signal [76]. Puncturable encryption [77] also provides forward secrecy that can be used for email and allows a recipient to revoke decryption capability for specific messages or time periods.

6.3.7. Encrypting Proxies. One approach to making interactions with PKI transparent to users is to layer encryption and signing below client software. Levien et al. [78] places this functionality between the email client software and the MTA, while Wolthusen [79] uses the operating system to intercept all network traffic and then automatically apply email encryption. Currently, several companies (*e.g.,* Symantec) offer automated encryption of emails by intercepting them as they traverse a corporate network.

7. Vulnerable Users

Vulnerable users deal with strongly sensitive information that could induce personal safety risks.

7.1. Pseudonymity

A primary concern for vulnerable users is to remain anonymous: *i.e.,* unlink the contents of the email from their true email address, their IP address, and/or the identity of their mail server. Senders generally opt for pseudonymity [80] where more than one email sent from the same pseudonymous account can be established as having the same origin, but no further information is known.[3] One method for pseudonymity is to use *layered encryption*, in which messages are routed through multiple non-colluding servers, with each server unwrapping a layer of encryption until the message is delivered to its destination, with the same happening for replies in reverse. This idea was championed by the cypherpunk movement [81], [82] and adapted to the email protocol with remailers [83], [84], [85]. The second is to simply register a webmail account under a pseudonymous email address. Users might additionally opt to use Tor to access their mailbox to hide their IP address.[4] One real-world example of this technique is Satoshi Nakamoto, the inventor of Bitcoin [87], who corresponded over webmail for many months while remaining anonymous. In both cases, pseudonymity is only realized as indistinguishability from a set of plausible candidates—the set of other users at the time of use [88]—which may be small, depending on the system and circumstances.

7.2. Other Concerns

Vulnerable users have a variety of other concerns that we list here but that we do not consider further in our analysis.

Traceability. Email senders for some time have abused the browser-like features of modern email clients to track a recipient [89]. A patchwork of mechanisms to prevent tracking is in use by mail providers, servers, clients, and browsers that is typically not comprehensive.

Deniability. Deniability considers a case where the recipient wants to authenticate the sender, but the sender does want the evidence to be convincing to anyone else. Cryptographers have suggested new signature types [90], [91], [92] to provide deniability, but these typically require trusted third parties and/or a robust PKI and have near-zero deployment in email clients.

Ephemerality. Once sent, a sender loses control over an email and the extent to which its contents will be archived. In order to automate a shorter retention period, emails might contain a link to the message body which is deposited with and automatically deleted by a trusted service provider or a distributed network [93], [94].

8. Secure Email Providers

Secure email providers provide end-to-end encryption between users of their service, such as ProtonMail [95], Hushmail [96], and Tutanota [97]. These services typically offer email that is mostly indistinguishable from non-secure email services, supporting both webmail and mobile clients, with automatic encryption between users. Combined, these providers serve millions of users.

A secure email provider automatically encrypts email between users of the system; the public key for a recipient is generally retrieved from a trusted public key server run by the email provider, and the email is encrypted with this key. This process is entirely transparent to the user, and there is typically no method to verify the fingerprint of the public keys.[5] To send email to a user of a different email provider, the sender typically chooses a *shared secret* (password) and the system emails the recipient a link to the email provider's information depot. The recipient must click a link to visit the secure email provider's website, then enter a password to have their web browser download and decrypt the message using JavaScript. Out-of-band distribution of the password is left to the user. Many secure email systems do not offer secure email that is interoperable with other secure email clients, though some are beginning to allow import and export of private keys.

An important concern for secure email systems is private key storage. In many systems, the private key is stored on the email provider's servers, encrypted with a password chosen by the user, so that the user can easily access secure email from a variety of devices. This requires the user to choose a password strong enough to resist offline password attacks [99] in case the private key storage is compromised or the service provider is malicious. In addition, this same password is often used to authenticate the user to the secure email system. In this case, the browser should not transmit the password to the service provider when authenticating the user. Possible approaches are hashing and salting the password in the browser before sending it to the server to retrieve the encrypted private key (Tutanota) or using the Secure Remote Password protocol [100] (ProtonMail).

Secure email systems that use webmail introduce additional potential attacks. On any particular visit, the email provider (perhaps under subpoena from a government) could insert malicious JavaScript that reveals their password, encryption keys, or plaintext. Alternatively, other JavaScript loaded on the same page, such as a third-party library, can manipulate the Document Object Model (DOM) or the execution environment. Additional methods are needed to provide code signing and privilege separation for JavaScript in the browser [101], [102].

We note here another approach uses a browser extension to enable users to send signed and encrypted email with their existing webmail provider. One category of extensions extends webmail to include PGP functionality, typically automating some key management tasks (Mailvelope and FlowCrypt). There are also browser extensions that provide automated S/MIME-based encryption and signing (Fossa Guard), encryption with a symmetric key held by the service (Virtru), and encryption using a password shared out of band (SecureGmail).

9. Typical Users

Some work has examined the question of why most people don't use encrypted email. Renaud et al. [103] found support for four reasons for non-adoption—lack of concern, misconceptions about threats, not perceiving

[3] Note that pseudonymity and message authentication techniques, such as DKIM or digital signatures, are not always contradictory. The latter techniques bind messages to keys and keys become pseudonyms. Pseudonymity forgoes the further binding (via a PKI) of keys to identities.

[4] OnionMail [86] provides email servers as Tor hidden services.

[5] Fingerprint comparison is common with secure messaging applications, but the feature is often ignored by users [98].

a significant threat, and not knowing how to protect themselves. An earlier survey of 400+ respondents by Garfinkel et al. [47] found that half indicated they didn't use encrypted email because they didn't know how, while the rest indicated they didn't think it was necessary, didn't care, or thought the effort would be wasted. Other work reports that users are unsure about when they would need secure email [104] and are skeptical that any system can secure their information [105]. It is not clear that users want to use digital signatures or encryption for daily, non-sensitive messages [106], [107]. Overall, work in this area demonstrates that usability is not the only obstacle to adoption, and that users don't perceive significant risk with email, lack knowledge about effective ways to mitigate risk, and don't have self-confidence about their ability to effectively use secure systems.

The usable security and privacy community is increasingly utilizing new approaches to address broader questions of adoption of security and privacy practices. Users are often rational when making decisions about whether to follow security advice; Herley [108] makes the case that users sometimes understand risks better than security experts, that worst-case harm is not the same as actual harm, and that user effort is not free. Sasse [109] has likewise warned against scaring or bullying people into doing the "right" thing. As a result, effort is being made to understand users' mental models [110], [111], [112], [113] when they interact with secure software and using risk communication techniques to better understand adoption or non-adoption of secure software [114], [115], among other methods.

10. Law Enforcement

Law enforcement prioritizes access to plaintext emails, either through broad surveillance or exceptional access such as with a warrant. This need for access to plaintext communications has led to calls for so-called encryption back doors, leading to regular debates on whether this is desirable or feasible. This debate originally surfaced in the U.S. in the 1990s and has been rekindled regularly. Proponents cite fears that widespread use of end-to-end encryption will enable criminals and terrorists to "go dark" and evade law enforcement. In response, privacy advocates decry growing mass surveillance, point to a history of abuses of wiretapping [116], and suggest that market forces will ensure there is plenty of unencrypted data for use by law enforcement regardless [117].

A 2015 paper from Abelson et al. [118] highlights risks of regulatory requirements in this area, reiterating many issues discussed in their earlier 1997 report [119]. Identified risks include reversing progress made in deploying forward secrecy, leading to weaker privacy guarantees when keys are compromised; substantial increases to system complexity, making systems harder to secure; and the concentration of value for targeted attacks. Their report also highlights jurisdictional issues that create significant complexity in a global Internet. More broadly, whenever service providers have access to keys that can decrypt customer email, this allows plaintext to be revealed due to incompetent or untrustworthy service providers, by disillusioned employees, by government subpoena, or by regulatory coercion.

11. Comparative Analysis of Stakeholder Concerns and Secure Email Systems

In the previous sections we reviewed the history of secure email using a framework of key stakeholders. In this section, we synthesize these stakeholder's goals and priorities into a set of seventeen properties that represent their desired security, utility, deployability, and usability features. Using these properties, we compare stakeholder priorities and evaluate existing secure email systems. The definition of each property is given in Appendix A, along with an explanation of how a given secure email system is rated to have full support, partial support, or no support in terms of meeting this property.

11.1. Stakeholder Priorities

Table 2 lists the primary stakeholders we identified in Table 1, along with ratings for how important each of the seventeen properties is to each stakeholder. A particular property can be a high priority, a low priority, or a non-priority. In some cases, we rank a stakeholder as highly valuing partial support of a property. We also identify several cases where a stakeholder has a high priority that the property is *not* met, meaning it is antithetical to their goals.

There are several cases where we believe there is a disagreement within a stakeholder group regarding the priority of a given property. A good example of this is preventing exceptional access to email (S4)—typical email users are split between those who advocate for government surveillance of email and who are willing to accept government access to email on presentation of a warrant, and those who strongly prefer end-to-end encryption that would prevent exceptional access. Likewise, privacy enthusiasts are split on whether there is a high priority on ensuring that private keys are accessible only to users (S3), with a minority placing a high priority on this property but others accepting password-protected cloud storage of a private key. Privacy enthusiasts are also split on whether persistent access to email is a high priority (T4), along similar lines. Finally, while many email service providers place a high priority on not being required to deploy new email-related servers to support a given technology (D2), this is likely not a high priority for larger providers. For example, large providers have shown a willingness to more rapidly adopt best practices such as STARTTLS and DKIM.

Table 2 illustrates the reality that there are significant disagreements between stakeholders in the secure email space and that no single solution will satisfy them all. In particular, we note major disagreements over the importance of private-key accessibility (S3), exceptional access (S4), sender pseudonymity (S7), server-side content-processing (T3), and persistent access (T4). An important result of our analysis is that some of the most fundamental disagreements occur over the utility properties of a secure email system. Email service providers, typical users, and enterprise organizations all place a high value on server-side content processing and persistent access to email. Yet, these are mostly low priorities for the other stake holders and, in some cases, antithetical to the principles held by vulnerable users.

TABLE 2. Stakeholder priorities.

	S1: *Protection from eavesdropping*	S2: *Protection from tampering and injection*	S3: *Private keys only accessible to user*	S4: *Prevents exceptional access*	S5: *Responsive public key revocation*	S6: *Provides a public key audit trail*	S7: *Supports sender pseudonymity*	S8: *Easy to detect phishing*	T1: *Supports user choice of email providers*	T2: *Supports user choice of identity provider*	T3: *Supports server-side content processing*	T4: *Provides persistent access to email*	D1: *No client software updates needed*	D2: *No email server updates needed*	D3: *No infrastructure updates needed*	U1: *Effortless same system encryption key discovery*	U2: *Effortless encryption/signing key validation*
Stakeholder	*Security*								*Utility*				*Deploy.*			*Usab.*	
Law Enforcement				☠			☠										
Email Service Providers	▦	▦						■	░	░	■	■	■	⚔	■		
Typical Users	▦	▦		⚔				■	■	░	■	■	■				
Enterprise Organizations	▦	▦		☠	■	■		■	░	░	■	■	░	░	■	■	■
Secure Email Providers	■	■	▦	■	■	░		■	░	░	░	░				■	■
Privacy Enthusiasts	■	■	⚔	■	■	░	░	■	■	■	░	⚔	░			■	■
Vulnerable Users	■	■	■	■	■	■	■	■	■	■	☠	☠				░	░

■ high priority for full support ▦ high priority for partial support ░ low priority □ a non-priority or not applicable
⚔ there is disagreement within the stakeholder group about the priority of this property ☠ high priority for no support

TABLE 3. Cryptographic approaches used to enhance email security.

	#	Cryptographic Approach	Reference
MTA-based	C1	Plaintext email	§2.1
	C2	Link encryption	§4.1
	C3	Provider-to-provider encryption	§5.2
	C4	Provider-to-provider signing	§4.2
	C5	Layered encryption	§7.1
End-to-End	C6	End-to-end encryption	§5.1 and §6.1
	C7	End-to-end signing	

TABLE 4. Key management schemes used to enhance email security.

#	Key Management Scheme	Reference
KM1	Certification Authority	§4.1
KM2	Certificate directory	§5.1
KM3	Manual key distribution	§6.1
KM4	Web of trust (WoT)	§6.1
KM5	WoT + key packet server	§6.1
KM6	Trusted public key server	§6.3.1
KM7	Audited public key server	§6.3.2
KM8	Trust on first use	§6.3.3
KM9	Shared secrets	§8
KM10	Key escrow	§5.1

The tussles among stakeholders help explain the history of how this space has evolved. The needs of typical users are largely met by email service providers; these two stakeholders disagree mainly on deployment properties that affect only the service provider (D2, D3), along with a tussle over exceptional access (S4). End-to-end encryption is not a priority for email service providers and typical users, and this explains why it is not pursued more broadly. The needs of some enterprise organizations to deploy secure email explains why they often adopt S/MIME based products and do not use typical email providers. They need encryption within the organization, plus escrow of private keys and content processing. They also have the IT budget to provide a seamless user experience. Privacy enthusiasts overlap significantly with enterprise organizations, but disagreements on private key storage (S3), server-side content processing (T3) and persistent access (T4) make finding common ground difficult. Secure email providers have emerged, with priorities that mostly match those of privacy enthusiasts, some of whom would previously have used PGP-based services. Some privacy enthusiasts would prefer the private key is only accessible to themselves (S3), but due to the loss of grass-roots support for PGP, the only apparent feasible alternative is password-protected keys used in secure webmail. The services offered by secure email providers have supported vastly more users of secure email than PGP ever did. No major commercial provider currently meets the needs of vulnerable users.

TABLE 5. A COMPARATIVE EVALUATION OF EXISTING SYSTEMS USED TO ENHANCE EMAIL SECURITY.

					S1: Protection from eavesdropping	*S2: Protection from tampering and injection*	*S3: Private keys only accessible to user*	*S4: Prevents exceptional access*	*S5: Responsive public key revocation*	*S6: Provides a public key audit trail*	*S7: Supports sender pseudonymity*	*S8: Easy to detect phishing*	*T1: Supports user choice of email providers*	*T2: Supports user choice of identity provider*	*T3: Supports server-side content processing*	*T4: Provides persistent access to email*	*D1: No client software updates needed*	*D2: No email server updates needed*	*D3: No infrastructure updates needed*	*U1: Effortless same system encryption key discovery*	*U2: Effortless encryption/signing key validation*
Family	#	System	Components	Format	*Security*								*Utility*				*Deploy.*			*Usab.*	
MTA-based	SYS1	Baseline email	C1	Plaintext	○	○		○			○	○	●		●	●	●	●	●		
	SYS2	Email + TLS and DKIM	C2, C4, KM1	Plaintext	○	◐		○			○	○	●		●	●	●	○	○		
	SYS3	Mixminion remailer [85]	C5, KM2	Plaintext	◐	○		○			●	○	●		●	●	○	○	○		
Walled Garden	SYS4	Corporate S/MIME	C6, C7, KM1, KM10	S/MIME	◐	◐	○	○	●	○	○	○	○	○	○	●	○	●	○	●	◐
	SYS5	Hosted S/MIME [48]	C3, C4, KM1, KM10	S/MIME	◐	◐	○	○	●	○	○	○	○	○	●	●	●	○	○	◐	◐
	SYS6	ProtonMail [95]	C6, C7, KM6, KM9	PGP	●	●	◐	◐	●	○	○	○	○	○	○	◐	○	●	○	●	◐
Open System	SYS7	PGP	C6, C7, KM5	PGP	●	●	●	●	○	○	○	○	●	●	○	○	○	●	○	○	○
	SYS8	Autocrypt [65]	C6, C7, KM8	PGP	●	●	●	●	○	○	○	○	●	●	○	○	○	●	●	◐	○
	SYS9	Key continuity [49]	C6, C7, KM1, KM8	S/MIME	●	●	●	●	○	○	○	◐	●	●	○	○	○	●	●	◐	○
	SYS10	Enhanced CT [60]	C6, C7, KM7	S/MIME	●	●	●	●	●	●	○	○	●	○	○	○	○	●	○	●	●
Proprietary	SYS11	Virtru [120]	C6, C7, KM10	Proprietary	●	●	○	◐	●	○	○	○	●	○	○	●	○	●	○	●	○

● full support, ◐ partial support, ○ no support, blank means not applicable

11.2. Secure Email Systems

We next evaluate a variety of secure email systems to determine how well they meet stakeholder needs. We began this exercise by identifying cryptographic approaches, in Table 3, and key management schemes, in Table 4. These are taken directly from our review of the history of secure email in prior sections. We then identified secure email systems that represent a broad cross-section of these primitives, starting with deployed systems and then augmenting this list with several from academic literature that represent different points on the design space not covered by deployed systems.

Table 5 evaluates the secure email systems according to our set of properties. These are divided into several related groups. Appendix B describes each system and explains ratings given for the properties.

The *MTA-based* systems generally are implemented by MTAs. Baseline email scores well on relevant utility, deployability, and usability benefits, but is lacking in security. Email with TLS and DKIM (considered best practices today) only halfway meets one of the security properties (S2), while sacrificing some deployability (D2, D3). A mixminion remailer supports sender pseudonymity (S7) and some protection from eavesdropping (S1), while preserving the utility of email, but requires new client software and email server updates.

Walled gardens, or closed systems, provide encrypted email primarily for clients of a particular enterprise or email provider. These systems do not provide choice of email provider (T1) or identity provider (T2). Corporate S/MIME provides partial support for confidentiality (S1) and integrity (S2), with full support for responsive revocation (S5), but sacrifices some utility, incurs some deployment costs, and offers only partial usability benefits. Hosted S/MIME, because it uses provider-to-provider encryption, provides some protection from eavesdropping (S1) and tampering (S2), while also providing server-side content processing (T3). Relative to corporate S/MIME, it shifts the deployment costs toward the provider (D2) and away from the end user (D1). ProtonMail provides full support for confidentiality and integrity, and partial support for ensuring the private key is only accessible to a user (S3) and preventing exceptional access (S4).

Open systems are designed to provide secure email between individual users, without requiring subscription to a particular provider's services, usually by working with a user's existing email account. Because of their use of end-to-end encryption, these systems sacrifice some utility. PGP, as commonly deployed using key packet servers, meets security properties that can be considered fundamental to secure email (S1, S2, S3, S4), with deployment burdens placed on the end user (D1). It supports user choice of both email provider (T1) and key server (T2), but otherwise has poor usability. Autocrypt scores similarly, being based on PGP, with easier encryption key discovery (U1) because it automatically treats keys in incoming emails as trusted. Key continuity [49] is the one system that provides some protection against phishing (S8), but otherwise scores identically to Autocrypt. Enhanced Certificate Transparency [60] fares the best among the rated security properties, while also scoring high on usability, but it depends on the adoption of additional email infrastructure (D3).

We include one *proprietary* system, Virtru [120], because of its different approach. Virtru encrypts email using a symmetric key, then stores the key at a key escrow server, which delivers the symmetric key only to authorized recipients of an email. Because the escrow system is separate from the email provider, no single entity has access to both encrypted email and the encryption keys. It does not ensure the private key is accessible only to the user (S3), but in return is able to provide persistent access to email (T4), similar to corporate and hosted S/MIME. However, relative to those systems, it is able to support user choice of email providers (T1) and have full support for protection from eavesdropping and integrity.

We note that detecting phishing is not well supported across all systems. Phishing continues to be a persistent problem [121], [122], recently leading to a $100 million bank transfer fraud [123] and a hack of the DNC that affected the 2016 U.S. Presidential campaign [124]. Measures that mitigate spam can help by also catching some phishing emails, yet the use of end-to-end encryption prevents server-side scanning for these attacks.

12. Discussion

Based on our systematization and analysis, we summarize the state of secure email, discuss the relative success of secure messaging, and suggest several research and development directions.

12.1. The State of Secure Email

We make the following observations on the general state of secure email.

12.1.1. Email today offers little protection. The vast majority of email users subscribe to systems that can be categorized as MTA-based (SYS1 or SYS2), with email sent as plaintext. While in some cases their email may be protected with link encryption and domain authentication, lack of adoption and the availability of easy attacks mean that plaintext email is subject to passive and active eavesdropping, as well as message forgery. Providers could create an interoperable Hosted S/MIME (SYS5) standard, which would automate provider-to-provider encryption and integrity while still providing the utility that typical users value. This could satisfy users who trust their email provider with plaintext. One complexity is that this could further exacerbate the split between large email providers, who have the resources to deploy additional security services, and smaller providers for whom this is a more daunting task. This split already exists with respect to deployment of link encryption and domain authentication. This suggests that work is needed by developers to simplify deployment of secure email services.

12.1.2. A one-size-fits-all solution is unlikely. It is clear from both Table 2 and Table 5 that stakeholders have conflicting priorities and that the needs of different stakeholders dictate diverging solutions. As such, it is unlikely that any single secure email system will be suitable for all users and their divergent use cases. Furthermore, no single party controls the email ecosystem, and widespread deployment of secure email needs cooperation of numerous stakeholders. No one stakeholder has the capability to build (or the ability to demand) a secure email system that provides seamless interoperability for all of the billions of email users and for all the diverse uses of email. This means that even in the best case, with different solutions being adopted by different parties, there will almost surely be interoperability challenges that act as natural roadblocks and will require significant investment to overcome, if this is even possible.

12.1.3. The PGP web of trust remains unsuccessful after 25 years. The web of trust that is central to the original design of PGP—including manual key exchange and trusted introducers—has largely failed. Its use is generally limited to isolated, small communities. Its appeal is that it allows quick, interoperable deployment in small groups without bureaucracy or costs of formal Certification Authorities, but in practice the downside is poor usability and lack of responsive revocation. Secure webmail has supplanted traditional PGP clients. As such, PGP has become more about the format of messages and keys, than the methods used to distribute and verify keys. PGP developers are moving toward systems that use automated key distribution and authentication, with the traditional manual trust decisions left to a small minority with specialized needs.

12.1.4. Secure email systems trade off privacy and utility. Systems ensuring the private key is accessible only to the user are incompatible with current server-based content-filtering and usually cannot offer persistent access. Only hosted S/MIME provides some protection from eavesdropping and also meets both of these utility properties. Unfortunately, this means many secure email systems lack important features such as search and spam filtering that are more difficult to provide without support from a provider's computing resources. Providing persistent access to users that value the privacy of their private key poses another challenge. Notably, if users are willing to sacrifice utility, then usability is generally not a significant hurdle, since many systems have adopted automated key management.

12.1.5. Choice of identity provider leads to key discovery problems. Open systems provide the ability for users to choose their identity provider, but generally struggle with effortless encryption key discovery. An illustration of this tradeoff can be seen by examining Enhanced Certificate Transparency. The authors of Enhanced CT suggest that a user's email provider can naturally serve as their identity provider. This makes encryption key discovery easy because any email client can parse a recipient's email address and translate the domain name into the identity provider, *e.g.,* using DNS. The authors also mention that users could choose their identity provider, but do not design a critical piece of the system—mapping a user's email address to their preferred identity provider at that point in time. This is a non-trivial problem. Autocrypt and Key Continuity use TOFU key exchanges for this reason—it's a simple way to allow for user choice of identity provider, with easy key discovery, though this sacrifices the ability to easily validate and thus trust keys. It remains to be seen if this problem can be solved in a way that a system would receive full marks for all usability properties.

12.1.6. Vulnerable users are not well served. Aside from vulnerable users, every stakeholder representing a class of user has their needs met by systems available today. Typical users are served by current offerings from email service providers (SYS1 or SYS2). Enterprises (and their employees) are served by corporate S/MIME (SYS4), which provides a combination of security, utility, and usability that matches their priorities. Deployment cost are likely what hinders its broader adoption among enterprises. Privacy enthusiasts are served by secure webmail services, with their stronger emphasis on end-to-end encryption and good usability, while sacrificing utility in order to meet these priorities. In contrast, there is no system that clearly serves vulnerable users well. PGP (SYS7) is perhaps the best option, given its use by investigative journalists [125], but it does not meet all of the security priorities of vulnerable users. No system except for remailers provides sender pseudonymity (S7), and these do not typically meet other security properties important to vulnerable users. The small size and desire for anonymity among members of this stakeholder group (journalists, dissidents, whistleblowers, informants, under-cover agents, criminals) does not lend itself to commercial solutions, and volunteer organizations in this area have historically struggled.

12.2. Isn't Secure Messaging the Answer?

The lack of adoption of end-to-end encrypted email is often contrasted with the wider success of secure messaging applications. WhatsApp and Facebook Messenger have over a billion users, while iMessage, Signal, Telegram, Line, and Viber have millions. These tools are typically designed to automate encryption for users, including automatic key exchange via a trusted key server and automatic end-to-end encryption of messages. The best of these tools provide forward secrecy and message deniability [75], [76] in addition to end-to-end encryption. It is recommended practice to encrypt all messages, however some applications make encryption optional, resulting in many users failing to turn encryption on [126].

Based on our analysis of secure email systems in Table 5, we can see that secure messaging applications are similar to other walled gardens, since they use a centralized key server and do not provide interoperability across different applications. The most similar approach among email systems is ProtonMail, which also uses a trusted key server, allowing users of the service to automatically encrypt messages to each other. Using a trusted key server means that users may be unaware of the security and usability tradeoffs they are making. Users of secure messaging applications are typically only warned to check the encryption keys if they change, and numerous studies have shown that these applications fail to help users understand how to do this successfully [127], [98], [128].

The success of secure messaging applications has led to some calls to abandon secure email in favor of these applications [129]. Given the usability failures of efforts to bring encrypted email to the masses, typified by PGP, this is understandable. However, there are important reasons to not give up on email. In contrast to messaging's walled gardens, email's open nature gives it fundamentally different uses, including easily communicating with strangers, sending long, content-rich messages, permanently archiving messages, searching past conversations, and attaching files. While email's additional features are part of the reason ubiquitous end-to-end encryption is so elusive, they are also why email is likely to continue to be a primary form of communication on the Internet for years to come. As such, there is still an important need to increase the security and privacy of email-based communication.

12.3. Research and Development Directions

This then begs the question—what would it take for secure email to have the success of secure messaging applications? Systems such as ProtonMail offer an analogous experience but are not nearly as popular. Given the openness of email, providing interoperability is clearly a major challenge to be solved. Other problems include coping with spam and malware, managing private keys, and archive vulnerability. We note that each of these issues would likely also plague secure messaging applications if they attempted to mimic the openness and features of secure email.

12.3.1. Interoperability. Interoperability among secure email systems is a complex topic. Because email has typically been an open system, allowing anybody to email anyone else, users may expect that secure email should likewise be open. However, we are far from achieving this today with secure email, since the primary secure email systems in use are walled gardens.

A natural first step is to develop standard methods for similar systems to interoperate. For example, a ProtonMail user could email a Tutanota user if these webmail systems opened access to their public key servers and agreed on a convention for identifying the key server of a given email domain, *e.g.,* via a DNS record. A similar approach could be used by corporate S/MIME systems to open their certificate directories to querying by outside entities. Even in this case, however, issues of privacy, spam, and trust arise. Some providers are unwilling to expose the public keys of their users to outside queries. If it is easy to identify the public key of any given email address, then users may receive *encrypted* spam, which would likely be harder for email providers to detect. Thus, queries may need to be subject to rate-limiting, restricted only to those who regularly email a user, require user approval, use domain whitelisting, or integrate other methods.

Moving beyond similar systems, interoperability becomes much thornier, given differing trust models used by secure email systems. For example, would it be desirable for a secure webmail system such as ProtonMail to allow its users to accept a trust-on-first-use exchange of keys from an Autocrypt user? Likewise, could a user of Enhanced CT trust a certificate signed by a corporate S/MIME user, if the corporation's server was not likewise audited? Would differences in trust be communicated meaningfully to users of the system and be actionable? Thus, even if formats and protocols were universally agreed upon, it's not clear whether interoperability is always desired or meaningful.

Finally, opening any system to interoperability means users will need help deciding which organizations or providers to trust. We argue it is both infeasible and unnecessary to expect that every individual or organization can be globally trusted by the others. Rather, we argue it is advisable to work toward a much more limited goal of establishing trust among communicating parties when they need it. Any individual user or organization has a relatively small set of other users or organizations that it needs to trust. Developing infrastructure and protocols with this end in mind would appear to be necessary to leverage any gains made in technical interoperability.

12.3.2. Coping with spam and malware. Another major problem for secure email is coping with spam and malware. Even if interoperability was a solved problem, authentication of an email sender is not the same as authorization to send email [1], and building a system that provides the former but not the latter simply means users will get authenticated spam and phishing emails.

One possibility is to try to work around this problem. A secure email client could accept encrypted email only from regular or accepted contacts; rejecting encrypted email from unapproved senders could serve as a viable substitute for spam and malware filtering. Spam and malware could still be propagated by compromising accounts and spreading it to others who have approved those users, but the attack surface would be significantly limited. Unfortunately, email providers are not likely to embrace such a system since it arguably offers less spam and malware protection for users as compared to current practice.

A better alternative might be to build secure email systems that allow for server-side content processing even when private keys are only accessible to users. One possibility is to develop improved methods for server-side content processing on data that is encrypted [130], [131], [132]. Alternatively, clients could send encrypted email and a decryption key to a trusted cloud computing environment [133], [134], *e.g.,* based on SGX secure enclaves, where the email could be decrypted and filtered for malware and spam. Likewise, a trusted computing environment could be used for storing and searching archives. Another possibility is to move email storage to edge devices owned by an end-user where content processing can be performed, with encrypted backup in the cloud to provide fault tolerance and portability.

12.3.3. Removing private key management barriers. Our analysis of key management schemes focused on public key distribution and revocation, mirroring the emphasis of work in the academic and developer communities. However, there are numerous open questions regarding how non-enterprise users will manage the full key life cycle, including private key storage, expiration, backup, and recovery [135, §13.7], [136]. These questions are complicated by issues such as whether to use separate keys for encrypting email during transmission, as opposed to those for long-term storage [44]. Storing keys in trusted hardware where they can't be exfiltrated solves some storage issues, but also requires users to create backup hardware keys and revoke keys stored in lost or stolen devices. It is worth noting that major browsers now support synchronizing passwords across user devices, and one part of solving key management problems may involve using similar techniques to synchronize private keys.

Another option is to extend S/MIME-based systems primarily used by enterprises so that they are also available to users. A primary advantage of this approach is that if a user loses their private key (*e.g.,* by losing their device storing the key) they are not locked out of their accounts—they can simply get a new certificate issued by recertifying their identity with a Certification Authority. This requires trusting the authority, but in return users could receive help with both account recovery and revocation. For this to work, users need more usable methods to interact with Certification Authorities than are available now.

12.3.4. Addressing archive vulnerability. One of the consequences of high-profile phishing attacks in recent years has been the digital theft of the extensive information stored in long-term email archives of various individuals, companies, and organizations. It is ironic that the most active areas of research into securing email are largely orthogonal to such widely reported compromises. While this issue might be categorized as a general data security issue, the way email products and architectures are designed (*e.g.,* emails archived by default, mail servers accessible by password) are contributing factors toward this vulnerability. It is unclear if technical solutions, revised social norms about email retention, or other approaches could be helpful in addressing this issue.

13. Conclusion

Deployment and adoption of end-to-end encrypted email continue to face many technical challenges, particularly related to key management. Our analysis indicates that conflicting interests among stakeholders explains the fragmented nature of existing secure email solutions and the lack of widespread adoption by the general public. Our analysis suggests it is time to acknowledge that a one-size-fits-all (*i.e.,* for all target scenarios, environments, and user classes) solution or architecture will not emerge. In this light, a significant barrier to progress is opposition to any new product or service that doesn't meet one stakeholder's particular needs, though it works well for others. A path forward is to acknowledge the need for alternate approaches and support advancement of alternatives in parallel. Divided communities and differing visions can lead to paralysis if we insist on a single solution, but it can also be a strength if we agree that multiple solutions can co-exist.

References

[1] S. M. Bellovin, "A look back at "security problems in the TCP/IP protocol suite"," in *ACSAC*, 2004.

[2] C. Partridge, "The technical development of Internet email," *IEEE Annals of the History of Computing*, vol. 30, no. 2, 2008.

[3] The Radicati Group, "Email statistics report, 2019-2023," 2019.

[4] Z. Durumeric, D. Adrian, A. Mirian, J. Kasten, E. Bursztein, N. Lidzborski, K. Thomas, V. Eranti, M. Bailey, and J. A. Halderman, "Neither snow nor rain nor MITM...: An empirical analysis of email delivery security," in *IMC*, 2015.

[5] I. D. Foster, J. Larson, M. Masich, A. C. Snoeren, S. Savage, and K. Levchenko, "Security by any other name: On the effectiveness of provider based email security," in *CCS*, 2015.

[6] W. Mayer, A. Zauner, M. Schmiedecker, and M. Huber, "No need for black chambers: Testing TLS in the e-mail ecosystem at large," in *IEEE ARES*, 2016.

[7] R. Holz, J. Amann, O. Mehani, M. Wachs, and M. A. Kaafar, "TLS in the wild: An Internet-wide analysis of TLS-based protocols for electronic communication," in *NDSS*, 2016.

[8] U. Rivner, "Anatomy of an attack," RSA blog, 1 April 2011. [Online]. Available: http://web.archive.org/web/20110413224418/http://blogs.rsa.com:80/rivner/anatomy-of-an-attack/

[9] A. Whitten and J. D. Tygar, "Why Johnny can't encrypt: A usability evaluation of PGP 5.0," in *USENIX Security Symposium*, 1999.

[10] S. Ruoti, J. Andersen, L. Dickinson, S. Heidbrink, T. Monson, M. O'Neill, K. Reese, B. Spendlove, E. Vaziripour, J. Wu, D. Zappala, and K. Seamons, "A usability study of four secure email tools using paired participants," *ACM Transactions on Privacy and Security*, vol. 22, no. 2, April 2019.

[11] J. C. Klensin, "Simple Mail Transfer Protocol," RFC 5321, October 2008.

[12] R. Gellens and J. Klensin, "Message submission for mail," RFC 6409, November 2011.

[13] D. Crocker, "Standard for the format of ARPA Internet text messages," RFC 822, August 1982.

[14] D. S. Kitterman, "Sender Policy Framework (SPF) for authorizing use of domains in email, version 1," RFC 7208, April 2014.

[15] P. E. Hoffman, "SMTP service extension for secure SMTP over Transport Layer Security," RFC 3207, February 2002.

[16] D. Margolis, M. Risher, N. Lidzborski, W. Chuang, D. Long, B. Ramakrishnan, A. Brotman, J. Jones, F. Martin, K. Umbach, and M. Laber, "SMTP Strict Transport Security," Mar. 2016, work in progress. [Online]. Available: https://tools.ietf.org/html/draft-margolis-smtp-sts-00

[17] R. L. Barnes, "DANE: Taking TLS authentication to the next level using DNSSEC," *IETF Journal*, vol. 7, no. 2, 2011.

[18] P. E. Hoffman and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA," RFC 6698, Aug. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6698.txt

[19] C. Newman, "Using TLS with IMAP, POP3 and ACAP," RFC 2595, June 1999.

[20] L. Levison, "Dark Internet Mail Environment architecture and specifications," March 2015. [Online]. Available: https://darkmail.info/downloads/dark-internet-mail-environment-march-2015.pdf

[21] E. Sparrow, H. Halpin, K. Kaneko, and R. Pollan, "LEAP: A next-generation client VPN and encrypted email provider," in *CANS*, 2016.

[22] J. Fenton, "Analysis of threats motivating DomainKeys Identified Mail (DKIM)," RFC 4686, September 2006.

[23] M. Kucherawy and E. Zwicky, "Domain-based Message Authentication, Reporting, and Conformance (DMARC)," RFC 7489, March 2015.

[24] K. Andersen, B. Long, S. M. Jones, and M. Kucherawy, "Authenticated Received Chain (ARC) protocol," IETF, Internet-Draft draft-ietf-dmarc-arc-protocol-09, Jul. 2017, work in progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-dmarc-arc-protocol-09

[25] D. Crocker, P. Hallam-Baker, and T. Hansen, "DomainKeys Identified Mail (DKIM) service overview," RFC 5585, July 2009.

[26] M. Kucherawy, D. Crocker, and T. Hansen, "DomainKeys Identified Mail (DKIM) signatures," RFC 6376, September 2011.

[27] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "DNS security introduction and requirements," RFC 4033, March 2005.

[28] S. M. Jones, J. Rae-Grant, J. T. Adams, and K. Andersen, "Recommended Usage of the Authenticated Received Chain (ARC)," IETF, Internet-Draft draft-ietf-dmarc-arc-usage-02, Jun. 2017, work in progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-dmarc-arc-usage-02

[29] P. Hoffman, "Allowing relaying in SMTP: A series of surveys," *Internet Mail Consortium Report*, vol. 16, 2002.

[30] R. Siemborski and A. Melnikov, "SMTP service extension for authentication," RFC 4954, July 2007.

[31] R. Siemborski and A. Menon-Sen, "The Post Office Protocol (POP3) Simple Authentication and Security Layer (SASL) authentication mechanism," RFC 5034, July 2007.

[32] R. Siemborski and A. Gulbrandsen, "IMAP extension for Simple Authentication and Security Layer (SASL) initial client response," RFC 4959, September 2007.

[33] D. Liu, S. Hao, and H. Wang, "All your DNS records point to us: Understanding the security threats of dangling DNS records," in *CCS*, 2016.

[34] S. T. Kent, "Internet privacy enhanced mail," *CACM*, vol. 36, no. 8, 1993.

[35] J. Linn, "Privacy enhancement for Internet electronic mail: Part I: Message encryption and authentication procedures," RFC 1421, February 1993.

[36] S. Kent, "Privacy enhancement for Internet electronic mail: Part II: Certificate-based key management," RFC 1422, February 1993.

[37] D. Balenson, "Privacy enhancement for Internet electronic mail: Part III: Algorithms, modes, and identifiers," RFC 1423, February 1993.

[38] B. Kaliski, "Privacy enhancement for Internet electronic mail: Part IV: Key certification and related services," RFC 1424, February 1993.

[39] B. Ramsdell and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) version 3.2 message specification," RFC 5751, January 2010.

[40] B. C. Ramsdell, "S/MIME version 3 message specification," RFC 2633, June 1999.

[41] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile," RFC 5280, May 2008.

[42] H. Orman, *Encrypted Email: The History and Technology of Message Privacy*. Springer, 2015.

[43] "Key management interoperability protocol specification version 1.3," OASIS Standard, 2016.

[44] R. Chandramouli, S. L. Garfinkel, S. J. Nightingale, and S. W. Rose, "Trustworthy email," *Special Publication (NIST SP) 800-177*, 2016.

[45] A. Kapadia, "A case (study) for usability in secure email communication," *IEEE S&P Magazine*, vol. 5, no. 2, 2007.

[46] A. Fry, S. Chiasson, and A. Somayaji, "Not sealed but delivered: The (un) usability of S/MIME today," in *ASIA*, 2012.

[47] S. L. Garfinkel, D. Margrave, J. I. Schiller, E. Nordlander, and R. C. Miller, "How to make secure email easier to use," in *CHI*, 2005.

[48] Google, "Hosted S/MIME by Google provides enhanced security for Gmail in the enterprise," 2019, https://security.googleblog.com/2017/02/hosted-smime-by-google-provides.html.

[49] S. L. Garfinkel and R. C. Miller, "Johnny 2: A user test of key continuity management with S/MIME and Outlook Express," in *SOUPS*, 2005.

[50] P. R. Zimmermann, *The official PGP user's guide*. MIT press, 1995.

[51] P. Zimmermann, *PGP source code and internals*. MIT Press, 1995.

[52] E. Lauzon, "The Philip Zimmermann investigation: The start of the fall of export restrictions on encryption software under first amendment free speech issues," *Syracuse L. Rev.*, vol. 48, p. 1307, 1998.

[53] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer, "OpenPGP message format," RFC 4880, November 2007.

[54] M. Elkins, D. D. Torto, R. Levien, and T. Roessler, "MIME security with OpenPGP," RFC 3156, August 2001.

[55] P. Zimmermann, "PGP marks 10th anniversary," 5 June 2001.

[56] S. E. McGregor, E. A. Watkins, M. N. Al-Ameen, K. Caine, and F. Roesner, "When the weakest link is strong: Secure collaboration in the case of the Panama papers," in *USENIX Security Symposium*, 2017.

[57] E. Atwater, C. Bocovich, U. Hengartner, E. Lank, and I. Goldberg, "Leading Johnny to water: Designing for usability and trust," in *SOUPS*, 2015.

[58] S. Ruoti, J. Andersen, T. Monson, D. Zappala, and K. Seamons, "A comparative usability study of key management in secure email," in *SOUPS*, 2018.

[59] W. Bai, M. Namara, Y. Qian, P. G. Kelley, M. L. Mazurek, and D. Kim, "An inconvenient trust: User attitudes toward security and usability tradeoffs for key-directory encryption systems," in *SOUPS*, 2016.

[60] M. D. Ryan, "Enhanced certificate transparency and end-to-end encrypted mail," in *NDSS*, 2014.

[61] M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten, and M. J. Freedman, "CONIKS: Bringing key transparency to end users." in *USENIX Security Symposium*, 2015.

[62] V. Roth, T. Straub, and K. Richter, "Security and usability engineering with particular attention to electronic mail," *International Journal of Human-Computer Studies*, vol. 63, no. 1, 2005.

[63] C. Masone and S. W. Smith, "Abuse: PKI for real-world email trust," in *EuroPKI*. Springer, 2009.

[64] V. Birk, H. Marques, Shelburn, and S. Koechli, "pretty Easy privacy (pEp): Privacy by default," IETF, Internet-Draft draft-birk-pep-00, Jun. 2017, work in progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-birk-pep-00

[65] A. Team, "Autocrypt level 1 specification, release 1.1.0," April 2019.

[66] H.-C. Hsiao, Y.-H. Lin, A. Studer, C. Studer, K.-H. Wang, H. Kikuchi, A. Perrig, H.-M. Sun, and B.-Y. Yang, "A study of user-friendly hash comparison schemes," in *ACSAC*, 2009.

[67] S. Dechand, D. Schürmann, T. IBR, K. Busse, Y. Acar, S. Fahl, and M. Smith, "An empirical study of textual key-fingerprint representations," in *USENIX Security*, 2016.

[68] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Crypto*, 1984.

[69] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *CCS*, 2008.

[70] S. Ruoti, N. Kim, B. Burgon, T. Van Der Horst, and K. Seamons, "Confused Johnny: When automatic encryption leads to confusion and mistakes," in *SOUPS*, 2013.

[71] S. Ruoti, J. Andersen, T. Hendershot, D. Zappala, and K. Seamons, "Private webmail 2.0: Simple and easy-to-use secure email," in *UIST*, 2016.

[72] A. Lerner, E. Zeng, and F. Roesner, "Confidante: Usable encrypted email: A case study with lawyers and journalists," in *IEEE EuroS&P*, 2017.

[73] B. Schneier and C. Hall, "An improved e-mail security protocol," in *ACSAC*, 1997.

[74] I. Brown and B. Laurie, "Security against compelled disclosure," in *ACSAC*, 2000.

[75] N. Borisov, I. Goldberg, and E. Brewer, "Off-the-record communication, or, why not to use PGP," in *WPES*, 2004.

[76] T. Perrin and M. Marlinspike, "Double ratchet algorithm, revision 1," signal.org, 2016.

[77] M. D. Green and I. Miers, "Forward secure asynchronous messaging from puncturable encryption," in *IEEE S&P*, 2015.

[78] R. Levien, L. McCarthy, and M. Blaze, "Transparent Internet e-mail security," in *NDSS*, 1996.

[79] S. D. Wolthusen, "A distributed multipurpose mail guard," in *IAW*, 2003.

[80] I. A. Goldberg, "A pseudonymous communications infrastructure for the Internet," Ph.D. dissertation, UC Berkeley, 2000.

[81] A. Narayanan, "What happened to the crypto dream?, Part 1," *IEEE S&P Magazine*, vol. 11, 2013.

[82] ——, "What happened to the crypto dream?, Part 2," *IEEE S&P Magazine*, vol. 11, 2013.

[83] I. Goldberg, D. Wagner, and E. Brewer, "Privacy-enhancing technologies for the Internet," in *IEEE COMPCON. Digest of Papers*, Feb 1997.

[84] I. Goldberg, "Privacy-enhancing technologies for the Internet, II: Five years later," in *PETS*, 2003.

[85] ——, "Privacy enhancing technologies for the Internet III: Ten years later," in *Digital Privacy: Theory, Technologies and Practices*, A. Acquisti, S. Gritzalis, C. Lambrinoudakis, and S. De Capitani di Vimercati, Eds. Auerbach Press, 2007.

[86] "Onionmail," 2019, https://onionmail.info.

[87] S. Nakamoto, "Bitcoin: A peer-to-peer electionic cash system," Unpublished, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[88] R. Dingledine and N. Mathewson, "Anonymity loves company: Usability and the network effect," in *WEIS*, 2006.

[89] S. Englehardt, J. Han, and A. Narayanan, "I never signed up for this: Privacy implications of email tracking," PETS, 2018.

[90] D. Chaum, "Designated confirmer signatures," in *EUROCRYPT*, 1995.

[91] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *EUROCRYPT*, 1996.

[92] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *ASIACRYPT*, 2001.

[93] R. Geambasu, T. Kohno, A. A. Levy, and H. M. Levy, "Vanish: Increasing data privacy with self-destructing data," in *USENIX Security Symposium*, 2009.

[94] S. Wolchok, O. S. Hoffman, N. Henninger, E. W. Felten, J. A. Haldermann, C. J. Rossback, B. Waters, and E. Witchel, "Defeating Vanish with low-cost sybil attacks against large DHTs," in *NDSS*, 2010.

[95] "Protonmail," 2019, https://protonmail.com/.

[96] "Hushmail," 2019, https://www.hushmail.com/.

[97] "Tutanota," 2019, https://tutanota.com/.

[98] S. Schröder, M. Huber, D. Wind, and C. Rottermanner, "When SIGNAL hits the fan: On the usability and security of state-of-the-art secure mobile messaging," in *EuroUSEC*, 2016.

[99] D. Florêncio, C. Herley, and P. C. van Oorschot, "An administrator's guide to Internet password research," in *USENIX LISA*, 2014.

[100] T. D. Wu, "The secure remote password protocol," in *NDSS*, 1998.

[101] L. Meyerovich and B. Livshits, "ConScript: Specifying and enforcing fine-grained security policies for JavaScript in the browser," in *IEEE S&P*, 2010.

[102] S. Van Acker, P. De Ryck, L. Desmet, F. Piessens, and W. Joosen, "WebJail: Least-privilege integration of third-party components in web mashups," in *ACSAC*, 2011.

[103] K. Renaud, M. Volkamer, and A. Renkema-Padmos, "Why doesn't Jane protect her privacy?" in *PETS*, 2014.

[104] S. Ruoti, J. Andersen, S. Heidbrink, M. O'Neill, E. Vaziripour, J. Wu, D. Zappala, and K. Seamons, ""We're on the same page": A usability study of secure email using pairs of novice users," in *CHI*, 2016.

[105] S. Ruoti, T. Monson, J. Wu, D. Zappala, and K. Seamons, "Weighing context and trade-offs: How suburban adults selected their online security posture," in *SOUPS*, 2017.

[106] S. Farrell, "Why don't we encrypt our email?" *IEEE Internet Computing*, vol. 13, no. 1, 2009.

[107] S. Gaw, E. W. Felten, and P. Fernandez-Kelly, "Secrecy, flagging, and paranoia: Adoption criteria in encrypted email," in *CHI*, 2006.

[108] C. Herley, "So long, and no thanks for the externalities: The rational rejection of security advice by users," in *NSPW*, 2009.

[109] A. Sasse, "Scaring and bullying people into security won't work," *IEEE S&P Magazine*, vol. 13, no. 3, 2015.

[110] R. Wash, "Folk models of home computer security," in *SOUPS*, 2010.

[111] M. Fagan and M. M. H. Khan, "Why do they do what they do?: A study of what motivates users to (not) follow computer security advice," in *SOUPS*, 2016.

[112] R. Kang, L. Dabbish, N. Fruchter, and S. Kiesler, ""My data just goes everywhere:" User mental models of the Internet and implications for privacy and security," in *SOUPS*, 2015.

[113] J. Wu and D. Zappala, "When is a tree really a truck? exploring mental models of encryption," in *SOUPS*, 2018.

[114] G. Stewart and D. Lacey, "Death by a thousand facts: Criticising the technocratic approach to information security awareness," *Information Management & Computer Security*, vol. 20, no. 1, 2012.

[115] J. Wu, C. Gatrell, D. Howard, J. Tyler, E. Vaziripour, K. Seamons, and D. Zappala, ""Something isn't secure, but I'm not sure how that translates into a problem": Promoting autonomy by designing for understanding in Signal," in *SOUPS*, 2019.

[116] W. Diffie and S. Landau, *Privacy on the Line: The Politics of Wiretapping and Encryption (2/e)*. The MIT Press, 2007, second edition 2007 (472 pages), first edition 1998 (352 pages).

[117] U. Gasser, N. Gertner, J. L. Goldsmith, S. Landau, J. S. Nye, D. O'Brien, M. G. Olsen, D. Renan, J. Sanchez, B. Schneider *et al.*, "Don't panic: Making progress on the "going dark" debate," Berkman Center for Internet & Society at Harvard Law School, 2016.

[118] H. Abelson, R. Anderson, S. M. Bellovin, J. Benaloh, M. Blaze, W. Diffie, J. Gilmore, M. Green, S. Landau, P. G. Neumann, R. L. Rivest, J. I. Schiller, B. Schneier, M. Specter, and D. J. Weitzner, "Keys under doormats: Mandating insecurity by requiring government access to all data and communications," *Journal of Cybersecurity*, vol. 1, no. 1, 2015.

[119] H. Abelson, R. J. Anderson, S. M. Bellovin, J. Benaloh, M. Blaze, W. Diffie, J. Gilmore, P. G. Neumann, R. L. Rivest, J. I. Schiller, and B. Schneier, "The risks of key recovery, key escrow, and trusted third-party encryption." *World Wide Web Journal*, vol. 2, no. 3, 1997.

[120] Virtru, "The simple guide to encryption key management: Understanding common data privacy methods and misconceptions," 2019, https://www.virtru.com/wp-content/themes/virtru/files/pdf/The%20Simple%20Guide%20to%20Encryption%20Key%20Management.pdf.

[121] K. Tian, S. T. Jan, H. Hu, D. Yao, and G. Wang, "Needle in a haystack: Tracking down elite phishing domains in the wild," in *IMC*, 2018.

[122] Q. Scheitle, O. Gasser, T. Nolte, J. Amann, L. Brent, G. Carle, R. Holz, T. C. Schmidt, and M. Wählisch, "The rise of certificate transparency and its implications on the internet ecosystem," in *IMC*, 2018.

[123] J. J. Roberts, "Exclusive: Facebook and Google were victims of $100m payment scam," Fortune Magazine, 27 April 2017. [Online]. Available: http://fortune.com/2017/04/27/facebook-google-rimasauskas/

[124] E. Nakashima and S. Harris, "How the Russians hacked the DNC and passed its emails to WikiLeaks," Washington Post, 13 July 2018. [Online]. Available: http://web.archive.org/web/20110413224418/http://blogs.rsa.com:80/rivner/anatomy-of-an-attack/

[125] P. Romera and C. S. Gallego, "How ICIJ deals with massive data leaks like the Panama Papers and Paradise Papers," 3 July 2018. [Online]. Available: https://www.icij.org/blog/2018/07/how-icij-deals-with-massive-data-leaks-like-the-panama-papers-and-paradise-papers/

[126] E. Vaziripour, J. Wu, R. Farahbakhsh, K. Seamons, M. O'Neill, and D. Zappala, "A survey of the privacy preferences and practices of iranian users of telegram," in *Workshop on Usable Security (USEC)*, 2018.

[127] R. Abu-Salma, K. Krol, S. Parkin, V. Koh, K. Kwan, J. Mahboob, Z. Traboulsi, and M. A. Sasse, "The security blanket of the chat world: An analytic evaluation and a user study of Telegram," in *European Workshop on Usable Security (EuroUSEC 2017)*. Internet Society, 2017.

[128] E. Vaziripour, J. Wu, M. O'Neill, R. Clinton, J. Whitehead, S. Heidbrink, K. Seamons, and D. Zappala, "Is that you, Alice? A usability study of the authentication ceremony of secure messaging applications," in *SOUPS*, 2017.

[129] F. Valsorda, "Op-ed: I'm throwing in the towel in PGP, and I work in security," Ars Technica, December 2016.

[130] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *IEEE S&P*, 2000.

[131] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[132] S. Kamara, "Encrypted search," *XRDS*, vol. 21, no. 3, pp. 30–34, Mar. 2015. [Online]. Available: http://doi.acm.org/10.1145/2730908

[133] N. Santos, K. P. Gummadi, and R. Rodrigues, "Towards trusted cloud computing." *HotCloud*, vol. 9, no. 9, p. 3, 2009.

[134] T. F.-M. Pasquier, J. Singh, D. Eyers, and J. Bacon, "Camflow: Managed data-sharing for cloud services," *IEEE Transactions on Cloud Computing*, vol. 5, no. 3, pp. 472–484, 2017.

[135] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 1996.

[136] S. Ruoti and K. Seamons, "Johnny's journey toward usable secure email," *IEEE Security & Privacy*, vol. 17, no. 6, pp. 72–76, 2019.

Appendix

The appendix contains definitions of properties and justificatons for ratings of the secure email systems.

1. Property Definitions

This section defines the properties that we use to compare stakeholder priorities and rate secure email systems.

Security Properties (S1–S8)

S1 ***Protection from eavesdropping*****:** A rating of full support indicates that the body of an email is kept confidential from all parties other than the sender and recipients. A rating of partial support indicates that the sender's and recipient's mail servers are able to read the body of the email, but not other relaying servers or network middleboxes.

S2 ***Protection from tampering and injection*****:** Full support indicates that email is authored by the claimed sender and has not been modified by other parties. Partial support indicates undetected modification or injection is possible by the sender's or recipient's mail servers.

S3 ***Private keys only accessible to user*****:** Full support indicates that no third-party has access to a user's private signing and/or decryption private keys. Partial support indicates password-encrypted private keys are stored by a third-party. Note, many user-chosen passwords may be unlikely to resist an offline guessing attack.

S4 ***Prevents exceptional access*****:** Full support indicates exceptional access by corporations or law enforcement to the plaintext contents of messages in exceptional circumstances is not possible. Partial support indicates exceptional access is possible, but requires the cooperation or mutual-coercion of multiple parties (*e.g.,* email provider and key server). Regular access to plaintext (*e.g.,* to scan for malware) implies exceptional access.

S5 ***Responsive public key revocation*****:**

Full support indicates an architecture allows for immediate and automatic checking of up-to-date revocation information.

S6 ***Provides a public key audit trail*****:** Full support indicates an audit log of public keys is available that provides non-equivocation [61], so that impersonation attacks can be detected.

S7 ***Supports sender pseudonymity*****:** Full support indicates that it is generally infeasible for the recipient to learn the sender's IP address, the sender's email address, and the sender's mail server. Note, because the contents of a message could potentially de-anonymize the sender, end-to-end encryption of the email body (S1) and the subject should also be used when desiring pseudonymity, though these are not included in our rating.

S8 ***Easy to detect phishing*****:** We award full support to systems that make it easy for a user to identify when an email is a phishing email. We award partial support if easy identification of phishing email is only possible for long-standing contacts—for example, using key continuity [49].

Utility Properties (T1–T4)

T1 ***Supports user choice of email providers*****:** An email provider is the company that a person uses to obtain an email address and arrange for email delivery, along with sometimes providing an email client (as in a webmail or mobile system). Full Support indicates that a user can use any email provider with the system.

T2 ***Supports user choice of identity provider*****:** An identity provider is the party that is responsible for binding a person's subject name or email address to their public key, such as a Certification Authority or trusted key server. Full Support indicates that a user can use any identity provider with the system.

T3 ***Supports server-side content processing*****:** Full support indicates server-side content processing is possible, for example to provide spam and malware filtering, to identify high priority emails, or to automatically label or reply to messages. Note that research into computing on encrypted data is active and promises to enable the composition of message confidentiality and content processing [132], however we assume in our evaluation that only existing techniques are utilized.

T4 ***Provides persistent access to email*****:** Full support indicates the user has persistent access to their email—whether through private key recovery or some other mechanism—without the need to remember or store a secret value. Partial support indicates that persistent access to their email is possible, but only if the user remembers or has access to a secret value. We assume that access to the private key is what is important here – the actual emails can be stored in encrypted form in the cloud or on a local hard drive and backed up for resilience to hard drive failure.

Deployability Properties (D1–D3)

D1 ***No client software updates needed*****:** Full support indicates that there is no need to update existing email clients or adopt a new email client. Requiring installation of a browser extension earns a rating of no support.

D2 ***No email server updates needed*****:** Full support indicates that there is no need to update existing email servers or adopt new email servers in order to support the secure email system.

D3 ***No infrastructure updates needed*****:** Full support indicates that there is no need to update existing non-email infrastructure or adopt new non-email infrastructure.

Usability Properties (U1–U2)

U1 ***Effortless same system encryption key discovery*****:** We rate systems based on whether discovery of the recipient's encryption key is effortless for users of the same system, meaning the same deployment of a walled garden or the same centralized key server. For open systems we define the same system to encompass any software that implements the system. Full support indicates that an email client can automatically acquire any recipient's encryption key who is using the same system. Partial support indicates the system distributes encryption keys by automatically attaching them to outgoing emails. This requires a sender to first receive email from a recipient before they can send encrypted email to that contact.

U2 ***Effortless encryption/signing key validation*****:** We rate whether a system makes it easy for a user to validate that a recipient's public key is the intended, legitimate public key for that user. Full support indicates a system automates validation, with a public key audit trail (S6) and responsive public key revocation (S5). Partial support indicates a system automates validation using a trusted key server or Certification Authority, with manual key validation needed only when there is concern that the the trusted entity might be acting maliciously.

2. Systems

This section describes the secure email systems from Table 5 and justifies our ratings of each system for each of the properties. We base our evaluation on how these systems currently function and not on how they could theoretically function.

SYS1 Baseline email: This system refers to sending and receiving plaintext email over unsecured links. This provides no security but has full support on relevant utility ad deployability properties. This system and other MTA-based approaches do not offer end-to-end encryption so they are not evaluated on properties related to key management or identity providers.

SYS2 Email + TLS and DKIM: This system uses TLS where available to secure the links between users, email providers, and mail transfer agents (MTAs). Additionally, Domain Keys Identified Email (DKIM) is used to sign email messages.

While links are encrypted, this does not prevent intermediary MTAs from reading the contents of a message, leading to this scheme being rated as having no support for confidentiality (S1). The use of DKIM is rated as partial support for integrity (S2). Many servers do not enforce TLS or use DKIM [4], [5], [7], so this system requires changes to email servers (D2). DKIM requires the adoption of DNSSEC, resulting in a rating of no support for no new non-email infrastructure (D3).

SYS3 Mixminion remailer [85]: This system uses layered encryption and mixminion relays between the sender's and recipients' mail providers.

This system provides pseudonymity (S7), but since the recipients' mail providers can see the contents of the message, it provides only partial support for confidentiality (S1). This system requires custom clients (D1) and new mixminion servers (D3), leading to a rating of no support for both properties.

SYS4 Corporate S/MIME: This system uses S/MIME to encrypt and sign messages. Key pairs for each user are stored and distributed by an LDAP server. Public keys for a user are discoverable by other users of the enterprise or organization, but not by outside users.

While messages are encrypted end-to-end, the private key (S3) is accessible to the enterprise, resulting in partial support for confidentiality (S1) and (S2), as well as a rating of no support for preventing exceptional access (S4). Even though ownership of the private key would allow server-side content processing (T3), this is not done in practice. Escrow of the private key allows users to retrieve lost keys, providing persistent access to their email (T4). Sender and recipient email addresses are unencrypted and do not provide pseudonymity (S7). Currently, corporate S/MIME solutions do not support public key audit trails (S6), but do provide responsive revocation (S5). This system does nothing to make it easy to detect phishing (S8).

Users do not have choice of email provider (T1) or Certification Authority (T2). While many clients already support this system (e.g., Outlook, Apple Mail), support is not universal, and so we rate it as needing client software updates (D1). There are no changes needed for email servers (D2), but it is necessary to deploy servers that support key escrow (D3). Users of this system can effortlessly discover (U1) public keys for users in the same organization, but validation (U2) requires trusting the Certification Authority.

SYS5 Hosted S/MIME [48]: This system is a variant of S/MIME that has the email provider (in practice, Google) store each user's public and private keys to encrypt and decrypt messages for users. This removes the need for email clients to support end-to-end encryption.

Because email is encrypted and decrypted by the email server, it is possible for that server to perform content processing on that material (T3). Because it is the server, not the client, that handles encryption, it is also the server (D2), not the client (D1), that needs to be modified to make this system work. Because a user needs to receive an email first in order to discover the encryption key for the sender of that email, the system receives partial support for effortless encryption key discovery (U1).

SYS6 ProtonMail [95]: This is a secure webmail system that provides automatic encryption among users of the system. The email provider stores a user's password-protected private key so the user can access their email from multiple devices. If senders wish to email recipients from other email providers, messages are encrypted using a shared secret.

This system uses end-to-end encryption, so it receives full support for protection from eavesdropping (S1) and tampering (S2). Because the service stores the user's (password-protected) private key, we rate it as having only partial support for that property (S3) and for prevention of exceptional access (S4), as well as partial support for persistent access to email (T4). Because ProtonMail offers a new service, we rate them as having no support for no new infrastructure needed (D2). Using a centralized key server for their users provides full support for effortless public key discovery (U1), but only partial support for validation (U2) since the server is trusted.

SYS7 PGP: This system uses PGP, with keys distributed in person or using a key packet server.

This system uses end-to-end encryption, so it receives full support for protection from eavesdropping (S1) and tampering (S2). Private keys are accessible only to their respective users (S3), so it receives full support for preventing exceptional access (S4), but effectively prevents server-side content processing (T3) and persistent access to email in the case of private key loss (T4). Sender and recipient email addresses are unencrypted and do not provide pseudonymity (S7). Key packet servers do not currently provide a public key audit trail (S6), nor do they currently provide effective and responsive revocation (S5).

This system requires that users install new software (D1) and that key packet servers (D3) are deployed, but does not require changes to email servers (D2). This system allows users their choice of email provider (T1) and key server (T2). Users of this system can't effortlessly discover (U1) public keys for users since they must know which key packet server the recipient uses. Likewise, the difficulty of setting up trusted introducers leads to a rating of no support for effortless validation (U2).

SYS8 Autocrypt [65]: This project seeks to improve the adoption of end-to-end encrypted email. The goals of the project are to protect against passive eavesdropping, focus on incremental deployment, avoid asking users about keys, change only email clients, and use decentralized, in-band key discovery. Current specifications build on PGP and call for encryption keys to be exchanged automatically over email, with the first key received from a particular email address being bound to that email address (trust-on-first-use). Key continuity could be used to help users identify malicious key changes, but has not currently been developed.

Because users must email each other to share keys, but otherwise discovery is automated, this system receives partial support for effortless public key discovery (U1). Because keys are not validated without some additional manual comparison it receives no support for effortless key validation (U2).

SYS9 Key continuity [49]: This system has users obtain self-signed certificates and then exchange them using trust-on-first-use.

Ratings for key continuity are identical to Autocrypt, except that the system provides partial support for detecting phishing (S8) since key continuity provides a warning when a different certificate is seen.

SYS10 Enhanced CT [60]: This system uses a Certification Authority and auditing via Certificate Transparency. A user's email provider typically acts as the identity provider, creating a certificate and uploading it to transparency logs.

This system uses end-to-end encryption, so it receives full support for protection from eavesdropping (S1) and tampering (S2). Private keys are accessible only to their respective users (S3), so it receives full support for preventing exceptional access (S4), but prevents server-side content processing (T3) and persistent access to email in the case of private key loss (T4). The use of a Certification Authority provides responsive revocation (S5), and the use of Certificate Transparency provides an audit trail for the user's public key (S6). Key discovery is effortless (U1) because the email provider for a recipient is assumed to be the identity provider as well. Key validation is likewise effortless (U2) because of the CA system and Certificate Transparency.

SYS11 Virtru [120]: This proprietary system operates both a basic and advanced email encryption service. In basic operation, a client encrypts email using a symmetric key, then stores the key with a Virtru key escrow server. This server implements access control and delivers the symmetric key only to authorized recipients of an email. In the more advanced operation, a customer operates a trusted key server, which encrypts the symmetric key with a public key for the recipient. We rate Virtru with its basic mode because few details are publicly available about the key management used in the advanced mode.

Because the email server does not have access to users' private keys, and the key escrow server does not have access to users' messages, this scheme is rated as having full support for confidentiality (S1) and integrity (S2). However, both parties could be coerced into revealing the contents of a user's message, so it is rated as having partial support for exceptional access (S4). The use of centralized, symmetric encryption provides responsive revocation because a compromised encryption key can be easily deactivated (S5), and escrow provides persistent access to email (T4). The system makes it easy to discover the (symmetric) encryption key (U1), but validation (U2) is rated as no support since there is no mechanism to verify the authenticity of the encryption key, as could be done in a public key system.