Challenges in using Cryptography - End-user and Developer Perspectives

Von der Fakultät für Elektrotechnik und Informatik

der Gottfried Wilhelm Leibniz Universität Hannover

zur Erlangung des akademischen Grades

Doktor rerum naturalium

(abgekürzt: Dr. rer. nat.)

genehmigte Dissertation

von Herrn

M. Sc.

Christian Stransky

2022

1. Referent: Prof. Dr. Sascha Fahl
2. Referent: Prof. Dr. Ralph Ewerth

Tag der Promotion: 14.02.2022

Erklärung

Ich versichere, dass ich meine Dissertation

„Challenges in using Cryptography - End-user and Developer Perspectives“

selbstständig und ohne fremde Hilfe angefertigt, mich dabei keinen anderen als den von mir ausdrücklich bezeichneten Quellen und Hilfen bedient und alle vollständig oder sinngemäß übernommenen Zitate als solche gekennzeichnet habe. Die Dissertation wurde in der vorliegenden oder einer ähnlichen Form noch bei keiner anderen in- oder ausländischen Hochschule anlässlich eines Promotionsgesuchs eingereicht und hat noch keinen anderen Prüfungszwecken gedient.

(Ort/Datum)	(Unterschrift mit Vor- und Zuname)

Summary

"Encryption is hard for everyone" is a prominent result of the security and privacy research to date. Email users struggle to encrypt their email, and institutions fail to roll out secure communication via email. Messaging users fail to understand through which most secure channel to send their most sensitive messages, and developers struggle with implementing cryptography securely.

To better understand how to support actors along the pipeline of developing, implementing, deploying, and using cryptography effectively, I leverage the human factor to understand their challenges and needs, as well as opportunities for support.

To support research in better understanding developers, I created a tool to remotely conduct developer studies, specifically with the goal of better understanding the implementation of cryptography. The tool was successfully used for several published developers studies.

To understand the institutional rollout of cryptography, I analyzed the email history of the past 27 years at Leibniz University Hannover and measured the usage of email encryption, finding that email encryption and signing is hardly used even in an institution with its own certificate authority. Furthermore, the usage of multiple email clients posed a significant challenge for users when using S/MIME and PGP.

To better understand and support end users, I conducted several studies with different text disclosures, icons, and animations to find out if users can be convinced to communicate via their secure messengers instead of switching to insecure alternatives. I found that users notice texts and animations, but their security perception did not change much between texts and visuals, as long as any information about encryption is shown.

In this dissertation, I investigated how to support researchers in conducting research with developers; I established that usability is one of the major factors in allowing developers to implement the functions of cryptographic libraries securely; I conducted the first large scale analysis of encrypted email, finding that, again, usability challenges can hamper adoption; finally, I established that the encryption of a channel can be effectively communicated to end users. In order to roll out secure use of cryptography to the masses, adoption needs to be usable on many levels. Developers need to be able to securely implement cryptography, and user communication needs to be either encrypted by default, and users need to be able to easily understand which communication' encryption protects them from whom. I hope that, with this dissertation, I show that, with supporting humans along the pipeline of cryptography, better security can be achieved for all.

Keywords: Usable Security, End User, Developer

Zusammenfassung

"Verschlüsselung ist für alle schwierig" ist ein wichtiges Ergebnis der bisherigen Sicherheits- und Datenschutzforschung. E-Mail-Benutzer:innen tun sich schwer, ihre E-Mails zu verschlüsseln, und Institutionen scheitern bei der Einführung einer sicheren E-Mail-Kommunikation, Messaging-Benutzer:innen wissen nicht, über welchen sicheren Kanal sie ihre sensiblen Nachrichten versenden sollen, und Entwickler:innen tun sich schwer mit der sicheren Implementierung von Kryptografie.

Um besser zu verstehen, wie man die Akteure bei der Entwicklung, der Implementierung, des Einsatzes und der Nutzung von Kryptografie effektiv unterstützen kann, nutze ich den menschlichen Faktor, um ihre Herausforderungen und Bedürfnisse sowie die Unterstützungsmöglichkeiten zu verstehen.

Um Entwickler:innen besser zu verstehen und Forscher:innen zu unterstützen, habe ich eine Plattform entwickelt, die die Durchführung von Entwicklerstudien aus der Ferne unterstützt, insbesondere mit dem Ziel, die Implementierung von Kryptografie besser zu verstehen. Die Plattform wurde erfolgreich für mehrere veröffentlichte Entwicklerstudien eingesetzt.

Um die institutionelle Einführung von Kryptografie zu verstehen, analysierte ich die E-Mailhistorie der letzten 27 Jahre an der Leibniz Universität Hannover und untersuchte die Nutzung von E-Mailverschlüsselung und fand heraus, dass E-Mailverschlüsselung und -signierung selbst in einer Umgebung mit einer eigenen Zertifizierungsstelle kaum genutzt wird. Darüber hinaus stellt die Nutzung mehrerer E-Mail-Clients eine große Herausforderung für die Anwender:innen bei der Verwendung von S/MIME und PGP dar.

Um die Endnutzer:innen besser zu verstehen und zu unterstützen, führte ich mehrere Studien mit unterschiedlichen Texten, Symbolen und Animationen durch, um herauszufinden, ob die Benutzer:innen davon überzeugt werden können, über ihren sicheren Messenger zu kommunizieren, anstatt zu unsicheren Alternativen zu wechseln. Ich fand heraus, dass die Nutzer:innen Texte und Animationen zwar wahrnehmen, dass sich die Sicherheitswahrnehmung aber nicht wesentlich ändert, solange den Nutzer:innen zumindest Informationen über die Verschlüsselung angezeigt werden.

In dieser Arbeit habe ich beispielhaft untersucht, wie Forscher:innen bei der Durchführung von Experimenten mit Entwickler:innen unterstützt werden können; ich habe festgestellt, dass die Benutzerfreundlichkeit einer der wichtigsten Faktoren ist, wenn es darum geht, Entwickler:innen die sichere Implementierung der Funktionen kryptografischer Bibliotheken zu ermöglichen; ich habe die erste groß angelegte Analyse verschlüsselter E-Mails durchgeführt und dabei festgestellt, dass auch hier Probleme bei der Benutzerfreundlichkeit die Akzeptanz behindern können; und schließlich habe ich festgestellt, dass die Verschlüsselung eines Kommunikationskanals den Endbenutzer:innen wirksam vermittelt werden kann. Um die sichere Nutzung von Kryptografie in der breiten Masse zu verbessern, muss diese auf vielen Ebenen benutzbar sein. Entwickler:innen müssen in der Lage sein, Kryptografie sicher zu implementieren, und die Kommunikation der Benutzer:innen muss entweder standardmäßig verschlüsselt sein, oder die Benutzer:innen müssen leicht verstehen können, welche Verschlüsselung der Kommunikation sie vor wem schützt. Ich hoffe, mit dieser Dissertation gezeigt zu haben, dass mit der Unterstützung von Menschen bei der Nutzung von Kryptografie eine bessere Sicherheit für alle erreicht werden kann.

Schlagwörter: Benutzbare IT-Sicherheit, Endanwender, Entwickler

Foreword

The research presented in this dissertation would not have been possible without the support of my advisor, co-authors, colleagues, research assistants, friends and my family.
My parents encouraged me from the beginning to follow my dreams and interests, I will be forever thankful for the support. I am grateful to Katharina Meister for all the love and support she provided during the writing of this dissertation.
I would also like to thank my advisor Sascha Fahl. He introduced me to the research area during my bachelor's thesis and has since then constantly supported me and made it possible for me to work on any research ideas I had.
I am also grateful to Yasemin Acar, who convinced me in the first place to pursue a doctoral degree together with her. I enjoyed all our discussions and night shifts during our paper deadlines.
I would further like to thank my colleagues Dominik Wermke, Nicolas Huaman, Marten Oltrogge, Alexander Krause, Sabrina Amft, Niklas Busch and Harjot Kaur for an inspiring work environment and all the discussions we had; my co-authors Michelle L. Mazurek, Blase Ur, Oliver Wiese, Arne Dreißigacker, Peter Leo Gorski, Sven Bugiel, Erik Derr, and all others for the great collaborations; Michael Brenner and Karsten Heiken for the support in our email research project; Yasemin Acar, Dominik Wermke and Nicolas Huaman for proofreading this dissertation.
Last but not least I would like to thank the research assistants Johanna Schrader, Marcel Jahnke, Sebastian Roth, Lukas Löhle who helped in my research projects and the students who wrote their bachelor's and master's theses under my supervision: Marcel Jahnke, Jascha Schütte, Jannik Meineke, Hannah Deters, Jakob Kühl, Till Deeke and Noah Wöhler.

List of Abbreviations

AIC	**A**kaike **I**nformation **C**riterion
AJAX	**A**synchronous **J**avaScript **a**nd **X**ML
API	**A**pplication **P**rogramming **I**nterface
AWS	**A**mazon **W**eb **S**ervices
CA	**C**ertificate **A**uthority
DFN	**D**eutsches **F**orschungs**n**etz (German National Research and Education Network)
DSA	**D**igital **S**ignature **A**lgorithm
E2E	**E**nd-**to**-**E**nd
EC2	Amazon **E**lastic **C**ompute **C**loud
ECDSA	**E**lliptic **C**urve **D**igital **S**ignature **A**lgorithm
HTTP	**H**yper **T**ext **T**ransfer **P**rotocol
HTTPS	**H**yper **T**ext **T**ransfer **P**rotocol **S**ecure
IDE	**I**ntegrated **D**evelopment **E**nvironment
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
IMAP	**I**nternet **M**essage **A**ccess **P**rotocol
JSON	**J**ava**S**cript **O**bject **N**otation
MITMA	**M**an **I**n **T**he **M**iddle **A**ttack
MDA	**M**ail **D**elivery **A**gent / **M**essage **D**elivery **A**gent
MTA	**M**ail **T**ransfer **A**gent / **M**essage **T**ransfer **A**gent
MUA	**M**ail **U**ser **A**gent / **M**essage **U**ser **A**gent
NIST	**N**ational **I**nstitute of **S**tandards and **T**echnology
NSA	**N**ational **S**ecurity **A**gency
PGP	**P**retty **G**ood **P**rivacy
PKI	**P**ublic **K**ey **I**nfrastructure
POP3	**P**ost **O**ffice **P**rotocol
RSA	**R**ivest–**S**hamir–**A**dleman (A public-key cryptosystem)
Q&A	**Q**uestion **&** **A**nswer
RFC	**R**equest **f**or **C**omments
SMTP	**S**imple **M**ail **T**ransfer **P**rotocol
SO	**S**tack **O**verflow
SQL	**S**tructured **Q**uery **L**anguage
S/MIME	**S**ecure/**M**ultipurpose **I**nternet **M**ail **E**xtensions
SSL	**S**secure **S**ocket **L**ayer
TLS	**T**ransport **L**ayer **S**ecurity
URL	**U**niform **R**esource **L**ocator

VM	**V**irtual **M**achine

Contents

Chapter 1

Introduction

This dissertation is based on published work that I conducted as the main research student. Chapters 2, 3, and 4 directly include this previously conducted research; previous work from the corresponding research papers has been slightly edited and adapted for this introduction. This introduction also includes developments presented at IEEE Security and Privacy in 2016 and 2017, previously published as conference papers ("You get where you're looking for: The Impact of Information Sources on Code Security" and "Comparing the Usability of Cryptographic APIs": [6, 7]). These papers were co-first authored with Yasemin Acar. Doowon Kim and supervising authors Michelle L. Mazurek, Sascha Fahl, Simson L. Garfinkel, and Michael Backes contributed to this research. I co-lead the study design, study execution, and evaluation of the studies in these papers; therefore, throughout this chapter, the academic "we" is used.

1.1 Motivation

"Encryption is hard for everyone" is a prominent result of more than two decades of research in the usable security and privacy community. Email users struggle to properly encrypt their emails and have for a long time [17, 20, 21, 62, 63, 64, 65, 79, 83, 111, 115, 116, 119, 120, 160]. Users of messaging apps fail to properly authenticate their contacts, and fail to recognize when a conversation is encrypted [151, 152]. Developers struggle to implement cryptography securely across all kinds of programming endeavors [6, 41, 48, 56, 67, 68, 90, 95, 96, 97]. Even though prominent messengers encrypt communication by default, previous research has shown that users will switch to less secure channels to transmit sensitive data [3, 4, 12, 32, 66]. Their mental models of encryption are incorrect and insufficient [3, 12, 32, 66], and they may make security decisions based on anecdotal advice [103]. Developers struggle with the secure use of encryption: multiple data breaches have shown that even large corporations leave user data unencrypted [85, 113, 146]. Website owners struggled to appropriately deploy HTTPS [47]; only a push from both main browser vendors and Let's Encrypt succeeded in making HTTPS the web standard with over 80% of pages loaded with HTTPS in September 2021 compared to 48.7% in Oktober 2015 [74, 124]. Especially application developers struggle with use of HTTPS, secure storage of user

data etc. [5, 7, 54, 92]. This persistent failure on all sides to successfully employ encryption to secure data at rest and in transit worries security researchers [7, 10, 78] and consumer protection agencies [51, 52], and does not only leave users at risk of identity theft and surveillance, but can also put those who rely on encryption to survive under heightened risk [106, 132]. It is therefore important to make widespread adoption of encrypted communication standard.

The case of email encryption is a prominent example for the wide gap between theoretical and practical use of encryption: Asymmetric encryption was proposed in the 1970s [36, 108], and PGP was introduced in 1991 [164]. To date, however, few people encrypt their emails, and few email clients offer a straightforward way to encrypt emails end-to-end [120]. Conversely, messaging apps that have been created to offer secure encryption and other privacy-preserving features are rarely adopted widely, as other factors are more important for users' choices of messengers [3, 31]. Only when a popular messaging application started implementing end-to-end encrypted messages by default, a large percentage of message traffic was encrypted [13, 157]. However, users still struggle to comprehend and execute messengers' authentication ceremonies [150, 151, 152].

Not only users struggle with encryption; webmasters and developers struggle with securely implementing encryption, or with components that are required for encryption, too: webmasters fail to appropriately use TLS certificates [47], Android developers struggle with client-side TLS validation [48, 95, 97]. In our past work, we have shown that developers are not solely responsible for their failure to securely implement encryption; unhelpful tooling, unusable documentation, insecure defaults, the complicated nature of the task, and insecure resources contribute to the problem. For example, we surveyed 295 Android developers about their IDE use and online research behavior, finding that the majority uses Android Studio, which by default supports turning off TLS certificate validation [7]. We also found that Stack Overflow, an online Q&A forum, the official Android documentation, and Google searches are popular ways to find information for security-relevant tasks, such as encryption or storing user credentials. In a controlled laboratory experiment with 54 students and professional Android developers from Germany and the U.S., we asked participants to write code relevant for security and privacy under time constraints. We controlled resource access by condition: participants either had free choice of resources or access to either Stack Overflow, the official Android documentation, or books only. Participants who were constrained to use Stack Overflow produced significantly more functional, and less secure code than those using the official Android documentation. Based on these poor security results for Stack Overflow use, we assessed the quality of Stack Overflow posts. We surveyed the 139 Q&A threads our participants had accessed during the lab study, and found that while only a quarter of them were helpful in solving the assigned tasks, only 17% of the helpful ones contained secure code snippets. For external validity, we randomly sampled 200,000 apps from Google Play, statically analyzed them, and found that 93.6% of those apps used at least one of the security-relevant API calls that our participants had also used in the lab study. Likewise, many of the security problems created under lab conditions also appear in the Google Play sample, possibly caused by Stack Overflow as well. Our results showcase that API documentation is secure but unusable, while peer-created informal resources like Stack Overflow is more usable, but can lead to insecurity. Since developers may be subject to time pressure, we can assume that they will continue to use the resource that quickly gets them functional results.

Laboratory studies are time-intensive to conduct, and also limit participants to the researchers' geolocation. Therefore, we developed a remote programming study tool called Developer Observatory [136], which we used in two remote security programming research studies and which is presented in Chapter 2. In the first study, we used Developer Observatory to conduct an online programming study to compare the usability of cryptographic Application Programming Interfaces (APIs) [6]. We started from the conventional wisdom that many dangerous cryptographic errors may be caused by APIs that are overly complicated, have insecure defaults or come with poor documentation. To counter this, researchers have developed cryptographic APIs that were designed with usability in mind. However, prior to our study, none of these had been evaluated for their ability to contribute to more secure development. In our research, we were the first to investigate how and why design and usability of cryptographic libraries impact how secure the code written with them can be. This research helped understand how to build effective future libraries, and improve existing ones. Using Developer Observatory, we conducted a 256-participant study with Python developers recruited from GitHub, in which we asked them to complete common security-relevant tasks involving symmetric and asymmetric cryptography while restricted to use one of five different APIs. We assess their results for functional correctness and security, comparing their results to their own impressions of whether they succeeded to write secure code with their assigned library. Our results suggested that designing for simplicity does not necessarily lead to usability, and that poor documentation, lack of code examples, and missing auxilary features such as secure key storage prevented participants from writing functional and secure code. Comprehensive documentation with easy-to-use code examples seems to make up for complex APIs in terms of functionally correct code and participant perceptions, but not code security. Across a fifth of insecure but functionally correct tasks, participants believed that their code was secure, which hints at bad communication from the API. We concluded that usability is not limited to a simple and convenient interface, but also includes broad support and accessible documentation with secure code examples.

In a second study using Developer Observatory, we aimed to better understand the GitHub convenience sample used by us and others for security development studies [9]. We conducted an experiment with 307 active GitHub users, asking them

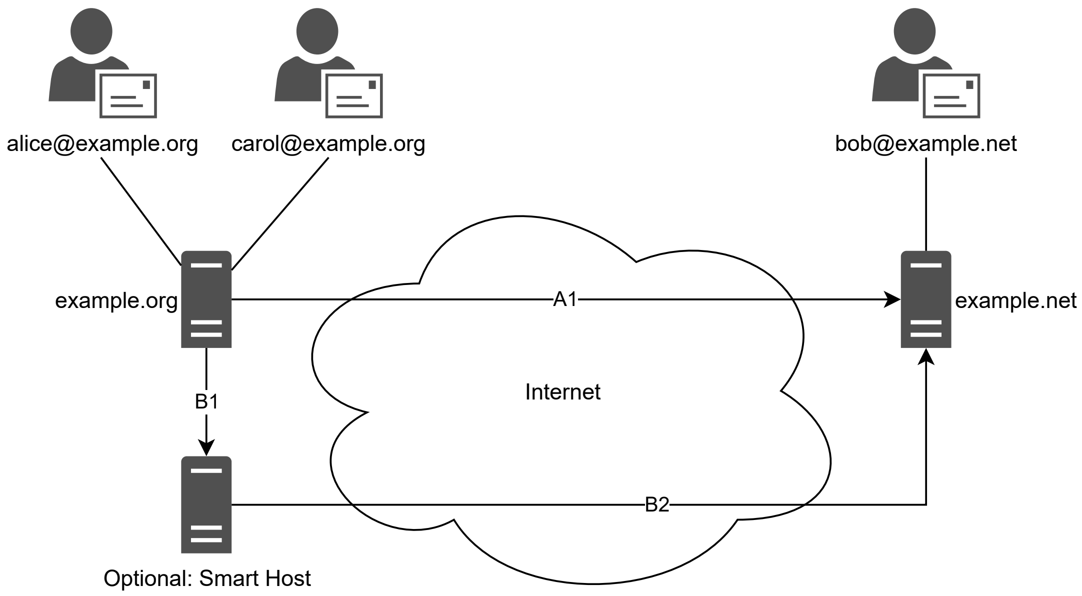


FIGURE 1.1: The general structure of email servers.

to complete tasks relevant to secure programming. We found differences in functional correctness and security related to self-reported years of experience, but no differences related to whether participants were students or worked professionally as developers, or whether they reported a security background. This study contributed to the discourse about validity when recruiting convenience samples to standin for the less accessible professional developers in secure programming studies.

1.2 Technical Background

This section will explain some required technical details to understand further chapters of this dissertation. A reader may skip this section if the technical background is already known, or they might return when reading the appropriate chapter.

1.2.1 The Email System - Why is Encryption important?

This section will give a short overview about the email system and how the emails are delivered through email servers. While there were different tools and systems in place, the general system has mostly been the same since the 1970s [145].

We first consider two examples based on the structure shown in Figure 1.1. In our first example, Alice wants to send an email to Carol. Both are using the email servers at `example.org`. The email is usually sent via the Simple Mail Transfer Protocol (SMTP) to the email server at `example.org`. This transfer can happen either unencrypted or using TLS/StartTLS[1], depending on the settings of the message user

[1]StartTLS initiates the TLS connection through an unencrypted channel and can be disabled by an active MITM attacker.

agent (MUA) and the available options on the server. Nowadays, most servers and apps will default to using TLS-encrypted connections, which means that emails cannot be read during transit from the MUA of Alice to the server at `example.org`. The MUA from Carol can now connect to `example.org` and retrieve emails through the IMAP or POP3 protocol. This transfer again can be encrypted with TLS. The email stored on the server at `example.org` however is always stored in plain text and the user has to trust that `example.org` does not accesses the email. If the receiver changes from `carol@example.org` to `bob@example.net`, several more servers are involved. The mail transfer agent (MTA) at `example.org` will identify that the target domain is outside of the local domain and depending on the configuration directly send it to the MTA at example.net (A1), or send it to an upstream smart host (B1) which then forwards the email to the MTA at example.net (B2). The traffic for each of these connections can again be sent unencrypted or encrypted with TLS. However, the email itself will always be available as plaintext on each involved server and can easily be intercepted by malicious entities. Smart hosts and relays are usually used for easier email traffic handling. They are often provided by a hosting company and deliver and/or receive emails for certain domains. The hosting company will attempt to make sure that their host does not end up on spam lists to ensure that emails can be delivered. Without such smart hosts, this work would have to be performed by the administrators of every MTA. Smart hosts can also exist for incoming emails, which handle spam and malware detection and ensure availability. An example for this is the DFN-MailSupport service,[2] a service that the DFN provides to all its members. DFN-MailSupport supports a variety of checks for incoming email, for example virus scans, spam discovery, black and white lists, realtime blackhole lists (RBL), and the sender policy framework (SPF).

1.2.2 End-to-End Encryption with PGP and S/MIME vs Transport Encryption

As described in the previous Section (1.2.1), email is often sent either unencrypted or using TLS and stored in plain text on email servers and relays. It is often advertised to the end users as encrypted and secure, who then mistakenly think that the whole process is encrypted [105]. However, TLS is only a transport encryption, which means that the email servers could still read the communication. A solution to this is the use of end-to-end encryption between sending and receiving users. For email, the two most commonly known solutions are OpenPGP[3] and Secure/Multipurpose Internet Mail Extensions (S/MIME). Both provide options to sign an email to cryptographically prove that the email was sent by the user and not modified in-between, and to additionally encrypt the email, so that only the receiver can decrypt and receive the email. However, the trust system used to verify the user's identity and the key exchange is fundamentally different between PGP and S/MIME.

[2]`https://www.mailsupport.dfn.de/der-dienst` (German)

[3]OpenPGP is an IETF standard for the PGP format. For simplicity, we abbreviate OpenPGP as PGP.

PGP was first standardized in 1996 and underwent changes to the standard in the following years, adding more and new algorithms. Public keys are usually distributed via public key servers, as attachment in an email or through Autocrypt[4], which was introduced in 2016. To ensure that a key actually belongs to a person X, the Web of Trust is used, that means other users will vouch that key x belongs to person X. If you trust the vouching person, then you also trust key x. S/MIME on the other hand uses the public key infrastructure (PKI) with certificate authorities (CA) that issue certificates to users. This is more commonly used in corporate environments, where it is easier to create an intermediate CA that only signs certificates for members of the company. As long as the root CA is trusted, any certificate derived from that root CA and its intermediate CAs will be trusted as well.

Configuring or using S/MIME or PGP is often challenging for users [63, 64, 160]. For S/MIME, a certificate has to be requested and deployed on the devices that the user uses, and the user's identity has to be verified by a certificate authority. The deployment has often been seen as challenging for users. PGP is in a similar situation, however, the tool support is less mature compared to S/MIME. While S/MIME has been widely supported in commercial tools, starting with Outlook 98 and early Thunderbird versions, PGP requires additional tooling. One example is Enigmail, which needs to be installed and configured in addition to Thunderbird to send and receive PGP emails. This poses another hurdle to lay users who might only want to send a simple email. Only recently, native support for PGP was added to Thunderbird[5], which improves the situation, but still requires users to actively create PGP keys and to verify the keys of peers to ensure that they have the correct key. The usability challenges that concern S/MIME and PGP will be discussed in more detail in the related work Sections 1.3 and 3.3.

1.2.3 End-to-End Encryption outside of Email: Mobile Messengers

Most users currently use multiple tools to communicate with their peers via texts, videos, images, and emoji. This can be in one-to-one chats or group chats. Most of these tools are available for different platforms to allow communication on every device. All of them have different privacy and security properties [148]. While most of these tools do not offer end-to-end encryption by default, most of them at least use transport encryption through TLS. A few tools offer dedicated so-called *private* or *secure* chats, which provide optional end-to-end encrypted channels. Examples for these tools are Telegram or the Facebook Messenger [46, 144]. Other tools like Signal, iMessage, or WhatsApp provide end-to-end encryption by default [16, 131, 158]. In contrast to email encryption with S/MIME or PGP, where the user has to manually configure the tool, create the keys, and potentially distribute their key, for most modern messengers, key exchange and generation occurs in the background

[4]`https://autocrypt.org`

[5]Starting with Thunderbird 78 (Summer 2020)

without user interaction [120]. This allows lay users to encrypt their communication and protect it from man-in-the-middle attacks without effort and without first learning how to. While this is great from a usability and security perspective, it creates new problems. Users are often unaware that they are using an encrypted channel and switch to a less secure tool like SMS/text message or email if they want to send sensitive information, as we will see in Chapter 4 of this thesis. Additionally, users rarely verify their contacts' keys, which makes them susceptible to man-in-the-middle attacks if an attacker manages to replace a contact's key [150, 151, 152].

Protocols The "SoK: Secure Messaging" by Unger et al. gives a useful overview over the different protocols used by messengers and their security properties [148]. The most adopted one is the Signal (formerly TextSecure) protocol. It is used by Signal, WhatsApp, the Skype Private Chat, and the Facebook Secret Chat. In general, it will use the TOFU (Trust-On-First-Use) principle, and trust a key received when starting a new conversation. If a man-in-the-middle attacker tries to replace the key, the user will receive a notification that the contact's key has changed. Depending on the implementation, the user then first has to accept the changed key or will simply be notified with explanatory information. An interested user can then make informed decisions, while the usability for novice users is still retained.

1.3 Related Research

This dissertation explores how human factors play a role for all actors involved in implementing, deploying and using cryptography, and how research can support actors in the secure implementation and use of cryptography.

The research in this dissertation builds on research in three core areas. Email encryption research, secure messaging research, and human factors research with developers. In order to roll out secure use of cryptography to the masses, adoption needs to be usable on many levels. Developers need to be able to securely implement cryptography, and user communication needs to be either encrypted by default, or users need to be able to easily understand which communication' encryption protects them from whom.

This section will give a short overview over related work that is groundwork for the research area in general. Furthermore, each chapter contains a related work section with the appropriate related work needed to understand the chapter.

1.3.1 Usable Security and Privacy for End-Users

The research field of usable security in general was started with three important papers and articles. In 1996, Mary Ellen Zurko and Richard T. Simon published the paper "User-centered security" at the 1996 Workshop on New Security Paradigms (NSPW'96) introducing the term "user-centered security" [165]. They assumed that "considering user needs as a primary design goal at the start of secure system development" would be a promising approach for the future [165], something that turned out to be true. The article "Users are not the enemy" by Anne Adams and Angela Sasse from 1999 is also groundbreaking. They conducted a user study with 30 interviews and found that users may compromise security mechanisms to achieve their primary task [11]. They identified the root cause as the complicated way that many security mechanisms are implemented, and that users are often not aware of security implications. They concluded that security departments need to adopt a user-centric approach to improve the situation [11]. Another seminal work is the paper "Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0" by Alma Whitten and J. D. Tygar from 1999. They conducted a usability evaluation of PGP 5.0 with cognitive walkthroughs combined with a lab study and identified challenges for end users with using PGP [160]. Only four of the 12 participants were able to successfully send a signed and encrypted email within 90 minutes. A quarter of the participants even accidentally exposed the secret by sending an unencrypted email. Two participants noticed this immediately afterwards while one participant thought the encryption was transparent to him. Participants furthermore had problems to handle the key exchange with other participants. One participant also forgot the password for their key pair and had to generate a new key [160]. This work by Whitten and Tygar was the beginning of a research direction (called "Johnny papers") that evaluated the usability of different communication systems. It is also an important motivation for Chapters 3 and 4 of this thesis.

Since the publication of these papers and articles, there have been many more peer-reviewed publications in different venues on the topic of usable security and privacy. The Symposium On Usable Privacy and Security (SOUPS), which was first held in 2005, is dedicated to usable security and privacy research and has been held annually since then. Usable security and privacy research is also published at other conferences and workshops. For example, the top-tier security conferences "IEEE Symposium on Security and Privacy (IEEE S&P)", "USENIX Security Symposium", and "ACM Conference on Computer and Communications Security (CCS)" regularly accept such papers and even dedicate tracks to this field. The history of usable security and privacy has been described in more detail in the book "Usable Security: History, Themes, and Challenges" by Simson Garfinkel and Heather Richter Lipford [61].

1.3.2 Usable Security and Privacy for Developers

In early usable security and privacy research, the research focus was mainly on end users and how to improve their understanding, and on making security systems easier to use. Recently, another subfield emerged, with a focus on developers. In 2012, Fahl et al. published the paper "Why Eve and Mallory Love Android: An Analysis of Android SSL (In)Security", where they evaluated apps on the Android Play

Store and found that many apps were vulnerable to man-in-the-middle attacks due to faulty TLS implementations [48]. In their follow-up paper "Rethinking SSL Development in an Appified World", they investigated possible causes and found threads on Stack Overflow where insecure examples for TLS were posted, and that were viewed over 30,000 times [50]. They also conducted interviews with 14 Android developers to explore the reasons for the faulty TLS implementations. They found that developers often implemented certificate validation insecurely for testing purposes with self-signed certificates, because no valid certificates were available for testing, and later on forgot about it. In other cases, customers requested it or developers were unaware of the introduced issues. Even after an explanation, one developer said that he checked the traffic with Wireshark, could not see plaintext data, and did not understand the issue.

Since then, research has started to focus on developers as users in need for usable security as well. In 2016 Green et al. published the paper "Developers Are Not The Enemy! The need for usable security APIs" in response to the paper "Users are not the enemy". They propose that API designers and library developers should design their libraries with usability in mind and not assume that every software developer is an expert in cryptography. They give a list of ten recommendations to follow when designing an usable and secure API, such as "Easy to learn, even without cryptographic expertise." ,"Easy to use, even without documentation", "Hard to misuse. Incorrect use should lead to visible errors", and "Defaults should be safe and never ambiguous" [68]. Since then, there have been several libraries that claim to be designed for usability and research papers that examined the usability of different cryptographic libraries [6, 88, 90, 91, 96, 163]. One example is our paper "Comparing the Usability of Cryptographic APIs" by Acar et al. which reports on a developer programming study with GitHub developers using various cryptographic APIs to implement small programming tasks. We found that around 20% of the functional code that was created by participants had security problems, while participants thought it was secure [6]. We found that simplicity of APIs is not enough for developers to produce secure and functional code. Instead, good documentation with code examples and a complete coverage of features is required to produce functional and secure code [6]. The study within that paper was conducted remotely using the platform described in Chapter 2.

1.4 Thesis Statement

The purpose of this work is to explore the problem space of challenges in using cryptography from a human factor perspective, and provide guidance on how to increase secure adoption of cryptography. Accordingly, the central thesis of this dissertation is:

In order to roll out secure use of cryptography to the masses, adoption needs to be usable on many levels. Developers need to be able to securely implement cryptography, and user communication needs to be either encrypted by default, or users need to be able to easily understand whether and how to encrypt their communication. For end-user email encryption, I expand on the large body of previous research by introducing a first-of-its-kind large-scale field study to measure the adoption of end-to-end encrypted and/or signed emails, identifying expectedly low adoption, as well as quantifying usability obstacles that may hinder adoption. For end-user encrypted messaging, I design encryption disclosures based on previous work and quantify their effect. To improve research on secure use of cryptography in software development, I contribute a research platform, and evaluate it by supporting studies that measure the usability of cryptographic APIs.

1.5 Contributions

Part of the research that contributed to this thesis has been published as conference or workshop papers. This section serves as a summary of the individual papers that contributed to this dissertation, listing the publication venue, and each author's contributions. As usual for collaborative research, this work would have been impossible without the significant contributions of the co-authors. An * (asterisk) denotes the main author(s), a † (dagger) denotes an author with significant contribution, and a $^+$ (plus sign) denotes a supervising author with significant contribution. For this dissertation, the papers have been mildly edited, updated and contextualized as needed for a coherent dissertation. The papers that contributed to this dissertation are:

You get where you're looking for: The Impact of Information Sources on Code Security

This paper contributes to the motivation of this dissertation. We report on a developer study with Android developers, in which we compared how developer documentation impacts code security, showing that the usability, as well as the presence of secure code examples, are important for code security. This paper contributed to Yasemin Acar's dissertation.

Authors: Yasemin Acar*, Michael Backes, Sascha Fahl$^+$, Doowon Kim†, Michelle L. Mazurek$^+$, Christian Stransky*.
Published at the 2016 IEEE Symposium on Security and Privacy.
Contributions to the paper: *The online survey was designed and evaluated by Yasemin Acar, Sascha Fahl, Michelle L. Mazurek, and me. The lab study was designed, conducted and evaluated by Yasemin Acar, Sascha Fahl, Michelle L. Mazurek and me. Doowon Kim helped in conducting the lab study. The API call confirmation study was designed by*

Sascha Fahl and Yasemin Acar. It was conducted and evaluated by me and Sascha Fahl. Yasemin Acar, Doowon Kim, Michelle L. Mazurek, Sascha Fahl, and I co-wrote the paper. Additionally Sven Bugiel assisted in the lab study and Marten Oltrogge assisted in the API call search.

Comparing the Usability of Cryptographic APIs

This paper contributes to the motivation of this dissertation. We researched how the usability of cryptographic APIs impacts code security in a developer study with Python developers, finding that designing simplistic APIs is not sufficient for secure code results, while safe defaults and supportive documentation are necessary. This paper contributed to Yasemin Acar's dissertation.

Authors: Yasemin Acar*, Michael Backes, Sascha Fahl$^+$, Simson L. Garfinkel$^+$, Doowon Kim†, Michelle L. Mazurek$^+$, Christian Stransky*.
Published at the 2017 IEEE Symposium on Security and Privacy.
Contributions to the paper: *The preliminary selection of libraries was done by Doowon Kim, Sascha Fahl, and me. The final selection of libraries was based on criteria discussed by all authors. The study was designed by Yasemin Acar, Michelle L. Mazurek, Sascha Fahl, and me. The literature review was conducted by Simson L. Garfinkel and Yasemin Acar. The surveys questionnaire was designed by Michelle L. Mazurek and Yasemin Acar. Sascha Fahl and I conducted the study. The evaluation was conducted jointly by Yasemin Acar, Doowon Kim, Michelle L. Mazurek, Sascha Fahl, and me. The paper was co-written by Yasemin Acar, Doowon Kim, Michelle L. Mazurek, Sascha Fahl, and me. Simson L. Garfinkel, Michelle L. Mazurek, and Sascha Fahl provided feedback and supervised during the whole process.*

Security Developer Studies with GitHub Users: Exploring a Convenience Sample

This paper contributes partly to the introduction of this dissertation. We report on a developer study with GitHub programmers, finding that they represent a diverse developer population with high variations in software development experience, as well as their security expertise. This paper contributed to Yasemin Acar's dissertation.

Authors: Yasemin Acar*, Christian Stransky*, Dominik Wermke†, Michelle L. Mazurek$^+$, Sascha Fahl$^+$.
Published at the Symposium on Usable Privacy and Security 2017.
Contributions to the paper: *The study was designed by Yasemin Acar, Michelle L.*

Mazurek, and me. The study was conducted by Yasemin Acar, Sascha Fahl, and me. All authors evaluated the study and co-wrote the paper.

Lessons Learned from Using an Online Platform to Conduct Large-Scale, Online Controlled Security Experiments with Software Developers

This paper contributes to Chapter 2 of this dissertation.
It reports on the design of a platform for conducting large scale online security developer studies, as well as the success of using the platform to conduct two insightful programming studies with developers recruited from GitHub.

Authors: Christian Stransky*, Yasemin Acar†, Duc Cuong Nguyen, Dominik Wermke, Elissa M. Redmiles, Doowon Kim, Michael Backes, Simson L. Garfinkel, Michelle L. Mazurek+, Sascha Fahl+
Published at 10th USENIX Workshop on Cyber Security Experimentation and Test (CSET'17).
Contributions to the paper: *The idea for the tool presented in this paper was developed by Yasemin Acar, Doowon Kim and me. It was designed and implemented by me and Doowon Kim. The developer studies were designed by Yasemin Acar, Michelle L. Mazurek, Sascha Fahl, and me; they were conducted by me and Sascha Fahl; and evaluated jointly with me, Yasemin Acar, Doowon Kim, Michelle L. Mazurek, and Sascha Fahl. The paper was written jointly with me, Yasemin Acar, Elissa Redmiles, Simson L. Garfinkel, Michelle L. Mazurek, Sascha Fahl, supported by Duc-Cuong Nguyen and Dominik Wermke. Simson L. Garfinkel, Michelle L.Mazurek, and Sascha Fahl supervised throughout. The libraries and version of Python used throughout the studies reflect the state of 2016 and 2017, respectively.*

Paper summary To better understand how developers implement, configure, and use cryptography, security and privacy researchers are increasingly conducting controlled experiments focusing on IT professionals, such as software developers and system administrators. These professionals are typically more difficult to recruit than general end-users. In order to allow for distributed recruitment of IT professionals for security user studies, we designed Developer Observatory, a browser-based virtual laboratory platform that enables controlled programming experiments while retaining most of the observational power of lab studies. Developer Observatory can be used to conduct large-scale, reliable online programming studies with reasonable external validity. We report on our experiences and lessons learned from two controlled programming experiments (n>200) conducted using Developer Observatory. In both studies, we aimed to better understand how developers interact with cryptographic APIs.

Developer Observatory is available on GitHub, and as of October 2021, it is actively being used; others have continued to develop it. It is also actively being used for scientific research: For example, it was modified[6] and used for a study on the API usability of stateful signature schemes by Zeier et al. [163]. Developer Observatory was also used for a study on compile-time crypto library warnings by Gorski et al. [67].

27 Years and 81 Million Opportunities Later: Investigating the Use of Email Encryption for an Entire University

This paper contributes to Chapter 3 of this dissertation.
In this research, we analyzed a whole university's data set of emails sent and received over 27 years, finding that a small minority of emails are end-to-end encrypted. We identify usability challenges posed by key rollovers, multiple devices and key exchange.

Authors: Christian Stransky*, Oliver Wiese†, Volker Roth+, Yasemin Acar†, Sascha Fahl+
Accepted for publication and to appear in the IEEE Symposium on Security & Privacy, May 2022.
Contributions to the paper: *The idea for this research was conceived by me and Oliver Wiese. I coordinated access and processing of the data with the data protection officer, as well as the data center. The code for the evaluation script was co-written by me and Oliver Wiese; we jointly conducted the data analysis. All authors jointly wrote the paper. Yasemin Acar, Volker Roth, and Sascha Fahl supervised throughout.*

Paper summary Email is one of the main communication tools and has seen significant adoption in the past decades. However, emails are sent in plain text by default and allow attackers easy access. Users can protect their emails by end-to-end encrypting them using tools such as S/MIME or PGP.

Although PGP had already been introduced in 1991, it is a commonly held belief that email encryption is a niche tool that has not seen widespread adoption to date. Previous user studies identified ample usability issues with email encryption such as key management and user interface challenges, which likely contribute to the limited success of email encryption.

However, so far ground truth based on longitudinal field data is missing in the literature. Towards filling this gap, we measure the use of email encryption based on 27 years of data for 37,089 users at a large university. While attending to ethical and data privacy concerns, we were able to analyze the use of S/MIME and PGP in 81,612,595 emails.

We found that only 5.46% of all users ever used S/MIME or PGP. This led to 0.06% encrypted and 2.8% signed emails. Users were more likely to use S/MIME than PGP by a factor of six. We saw that using multiple email clients had a negative

[6]`https://github.com/leddy-gee/developer-observatory-fork`

impact on signing as well as encrypting emails and that only 3.36% of all emails between S/MIME users who had previously exchanged certificates were encrypted on average.

Our results imply that the adoption of email encryption is indeed very low and that key management challenges negatively impact even users who have set up S/MIME or PGP previously.

On the Limited Impact of Visualizing Encryption: Perceptions of E2E Messaging Security

This paper contributes to Chapter 4 of this dissertation.
We showed that the presence of a visual or textual cue can successfully communicate encryption properties to end-users. A prominently displayed textual disclosure can meaningfully communicate encryption; additional visual indicators may not be needed.

Authors: Christian Stransky[*], Dominik Wermke[†], Johanna Schrader[†], Nicolas Huaman[†], Yasemin Acar[†], Anna Lena Fehlhaber, Miranda Wei, Blase Ur[+], Sascha Fahl[+]
Published at the Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021).
Contributions to the paper: *I conceived of the idea for this paper and implemented the changes to Signal, supported by Marcel Jahnke. I co-designed the studies with Dominik Wermke, Yasemin Acar, Blase Ur, and Sascha Fahl. Johanna Schrader and Miranda Wei co-designed all but the fifth study. The studies were conducted by me; qualitative results were coded by me, Dominik Wermke, Johanna Schrader, and Anna Lena Fehlhaber. Dominik Wermke and I conducted the quantitative evaluation. The paper was written jointly with Dominik Wermke, Nicolas Huaman, Yasemin Acar, Blase Ur, Sascha Fahl, and me. Yasemin Acar, Blase Ur, and Sascha Fahl supervised throughout. The tool was based on a 2019 Signal version and the 2019 WhatsApp design.*

Paper summary

Communication tools with end-to-end (E2E) encryption help users maintain their privacy. Although messengers like WhatsApp and Signal bring end-to-end encryption to a broad audience, past work has documented misconceptions of their security and privacy properties. Through a series of five online studies with 683 total participants, we investigated whether making an app's end-to-end encryption more visible improves perceptions of trust, security, and privacy. We first investigated why participants use particular messaging tools, validating a prior finding that many users mistakenly think SMS and email are more secure than end-to-end encrypted messengers. We then studied the effect of making end-to-end encryption more visible in a messaging app. We compared six different text disclosures, three different icons, and three different animations of the encryption process. We found that simple text disclosures that messages are "encrypted" are sufficient. Surprisingly, the icons negatively impacted perceptions. While qualitative responses to the animations showed they successfully conveyed and emphasized "security" and "encryption," the animations did not significantly impact participants' quantitative perceptions of the overall trustworthiness, security, and privacy of end-to-end encrypted messaging. We confirmed and unpacked this result through a validation study, finding that user perceptions depend more on preconceived expectations and an app's reputation than visualizations of security mechanisms.

Recognitions

- *The paper "You get where you're looking for: The Impact of Information Sources on Code Security", which contributed to this dissertation's introduction and to which I contributed as a main author ([7]), won the National Security Agency's 2016 Best Scientific Cybersecurity Paper Competition.*[7]
- *The paper "Comparing the Usability of Cryptographic APIs" was invited for presentation at Real World Crypto 2018 after publication.*

[7]`https://cps-vo.org/node/39262`

Chapter 2

Lessons Learned from Using an Online Platform to Conduct Large-Scale, Online Controlled Security Experiments with Software Developers

Disclaimer: *The contents of this chapter were previously published as part of the conference paper "Lessons Learned from Using an Online Platform to Conduct Large-Scale, Online Controlled Security Experiments with Software Developers", presented at the 10th USENIX Workshop on Cyber Security Experimentation and Test (CSET'17). This research was conducted as a team with my co-authors Yasemin Acar, Duc-Duong Nguyen, Dominik Wermke, Doowon Kim, Elissa Redmiles, Michael Backes, Simson L. Garfinkel, Michelle L. Mazurek, and Sascha Fahl; this chapter therefore uses the academic "we". The idea for the tool presented in this paper was developed by Yasemin Acar, Doowon Kim and me. It was designed and implemented by me and Doowon Kim. The developer studies were designed by Yasemin Acar, Michelle L. Mazurek, Sascha Fahl, and me; they were conducted by me and Sascha Fahl; and evaluated jointly with me, Yasemin Acar, Doowon Kim, Michelle L. Mazurek, and Sascha Fahl. The paper was written jointly with me, Yasemin Acar, Elissa Redmiles, Simson L. Garfinkel, Michelle L. Mazurek, Sascha Fahl, supported by Duc-Cuong Nguyen and Dominik Wermke. Simson L. Garfinkel, Michelle L. Mazurek, and Sascha Fahl supervised throughout. The libraries and version of Python used throughout the studies reflect the state of 2016 and 2017, respectively.*

Abstract

Security and privacy researchers are increasingly conducting controlled experiments focusing on IT professionals, such as software developers and system administrators. These professionals are typically more difficult to recruit than general end-users. In order to allow for distributed recruitment of IT professionals for security user studies, we designed Developer Observatory, a browser-based virtual laboratory platform that enables controlled programming experiments while retaining most of the observational power of lab studies. The Developer Observatory can be used to conduct large-scale, reliable online programming studies with reasonable external validity. We report on our experiences and lessons learned from two controlled programming experiments (n>200) conducted using Developer Observatory.

2.1 Introduction

While software developers have specialized technical skills, they are rarely security experts [6, 7]. Indeed, developers frequently make errors for many of the same reasons that end-users do: because they are uninformed, because they prioritize other requirements over security, and because tools for achieving security are difficult to use correctly. Consequently, identifying and solving security problems faced by software developers and other professionals promises to have a high impact on the overall security of software ecosystems [8].

Prior work observing or surveying software developers and other IT professionals is often limited by small sample sizes, as dictated by the high effort and cost necessary to recruit these professionals. To remedy the cost and difficulty of recruiting developers for security studies, we developed a browser-based virtual laboratory platform, Developer Observatory, to allow for controlled, geographically distributed experiments. In doing so, we wanted to retain as many as possible of the benefits of a lab study: controlling the programming environment, collecting in-depth data about when developers struggle and the strategies they use to solve problems, and administering exit questionnaires.

In this work, we describe a framework that allows researchers to conduct online secure-programming studies with remote developer participants. This framework includes mechanisms for randomly assigning participants to conditions, as well as the ability to administer surveys during and after study tasks. Our design also takes into account security (from misbehaving participants), participant privacy, and adaptability for a broad range of similar studies. Here, we report on our experiences instantiating and conducting experiments using our implementation of this framework: Developer Observatory.

Developer Observatory is instantiated using Jupyter Notebooks (for in-browser Python interpretation), Amazon EC2 (to isolate participants from each other and scale with demand), and the Qualtrics online survey platform. Developer Observatory allows for in-browser code editing so that participants can write and test code without installing anything, while also enabling researchers to collect detailed information about participant's activities. We successfully deployed Developer Observatory as part of two large-scale controlled experiments [6, 9], in which 2 396 developers interacted with our system and 563 completed an entire study run.

We make the following concrete contributions:

- We describe requirements for a scalable online secure-programming research tool.
- We develop a framework to meet these requirements and implement this framework in a complete study infrastructure (Developer Observatory), which we publicly release to the research community.
- We report on the benefits, challenges, and lessons learned when we used Developer Observatory for two large-scale studies of software developers.

2.2 Related Work

In order to make empirical studies in security easier to conduct and reproduce, a number of researchers have developed measurement systems and environments. The DETER project, for example, was created to support testing cybersecurity technology at scale, mimicking real-world deployments [23]. Other network-security testbeds support distributed networking research; Siaterlis and Masera provide a survey [128].

Relatively few such measurement environments, however, exist for human subjects research. A number of researchers have proposed guidelines for conducting empirical human-subjects experiments in computer science generally [35, 129], including an emphasis on choosing representative samples. Phonelab supports mobile-computing field studies, including collection of end-user data [93]. The University of Maryland's Build It, Break It, Fix It competition platform allows researchers to observe how participants develop and attack security-relevant systems [114]. The Security Behavior Observatory, recent work by Wash et al. examining the security behavior of undergraduates, and Levecque and Lalonde's work on cybersecurity clinical trials all apply client-side infrastructure to collect field data on security behavior, but do not enable direct, controlled experimentation [59, 81, 154].

Our work is thus distinctively different, offering a scalable tool for conducting online programming experiments with software developers. While we developed Developer Observatory for the purpose of conducting security experiments, its use could extend beyond security into software development and other computer science fields, enabling evaluation of Application Programmer Interfaces (APIs) and/or observation of the learning process of software developers.

2.3 Requirements

We distill requirements for online secure-programming experiments targeting software developers.

2.3.1 Validity Requirements

Validity requirements are designed to ensure that a study successfully meets its scientific goals for measuring useful information about participant behavior.

Internal Validity The framework should prevent duplicate participation to the extent possible. The framework should support randomly assigning participants to conditions (with or without weighting or blocking). The framework should also support showing tasks to participants one at a time, and should allow researchers to specify a randomized or rotated order for tasks. We also want the framework to promote a study environment that is as similar as possible across all participants: e.g. using the same software versions and pre-installed libraries.

Ecological Validity. In any laboratory study, it is important to maximize ecological validity, or the ways in which the study resembles real-life situations, to the extent possible. The framework should promote ecological validity by allowing participants to use their own equipment and software setup whenever possible, and to participate from their own home or work environments.

Data Collection. The framework should support automated collection of as much data as possible. This starts with, but is not limited to, collecting the code participants write and self-reported survey questions about their demographics and their experience with the study. For example, we would also like to capture when code is copy-and-pasted (and potentially ask the participant where it came from), how many times the participant tests their code, and whether it produces execution errors. We would also like to capture timestamps for all relevant events. The data collection mechanism should be extensible to support varying needs across studies.

2.3.2 Interaction Requirements

We want the framework to be easy to use for both participants and researchers.

Participants. We do not want interaction difficulties to drive away otherwise-willing participants. Participants should be able to participate with the equipment and software they have, without having to install libraries, virtual environments, or programming languages. We want each participant's experience to be seamless, from clicking a link in a recruitment advertisement through completing all programming and survey tasks, without having to take explicit actions like entering an identification (ID) value, creating an account, or stopping to download or install. Participants who give up on a task should (where reasonable within the study design) be able to skip it, provide information about why, and continue to the next task, rather than quitting the study entirely.

Researchers. The framework should make it easy for researchers to develop a new study; this means it should be modular and extensible, and in particular that items that are expected to change from study to study should be isolated from longer-term infrastructure components. It should be easy for researchers to monitor an ongoing study, in order to manage the rate of participation and to quickly identify and correct any problems that occur. The collected study data should be easy to export, and data collected in different parts of the study should be easily linkable via a consistent participant pseudonym. Finally, the infrastructure should be deployable within most researchers' budgets.

2.3.3 Technical Requirements

We define five technical requirements, as follows.

Reliability. Participants, who are recruited by email or web advertisement, may follow the provided link and begin participation at any time. As such, it is critical that the study infrastructure is robust and reliable, without requiring researchers to constantly monitor that it is operating properly while the study is running.

Scalability. The scale of desired participation will vary based on requirements of individual studies, but participant counts in the order of hundreds of participants seem likely to be needed. While it may not be necessary for all participants to run at one time, the infrastructure must expand to allow sufficient simultaneous participants, and data storage and retrieval must be designed for the total participant load.

Participant Isolation. Participants must not be able to see or alter others' code or survey responses. Further, participants should not be able to access information about conditions to which they have not been assigned (to protect the integrity of the experimental design).

Security. More generally, participants should not be able to break or exploit the study infrastructure. Specifically, participants should not be able to access the underlying operating system to perform operations outside the scope of the study, should not be able to disrupt the study infrastructure, and should not be able to, e.g., use

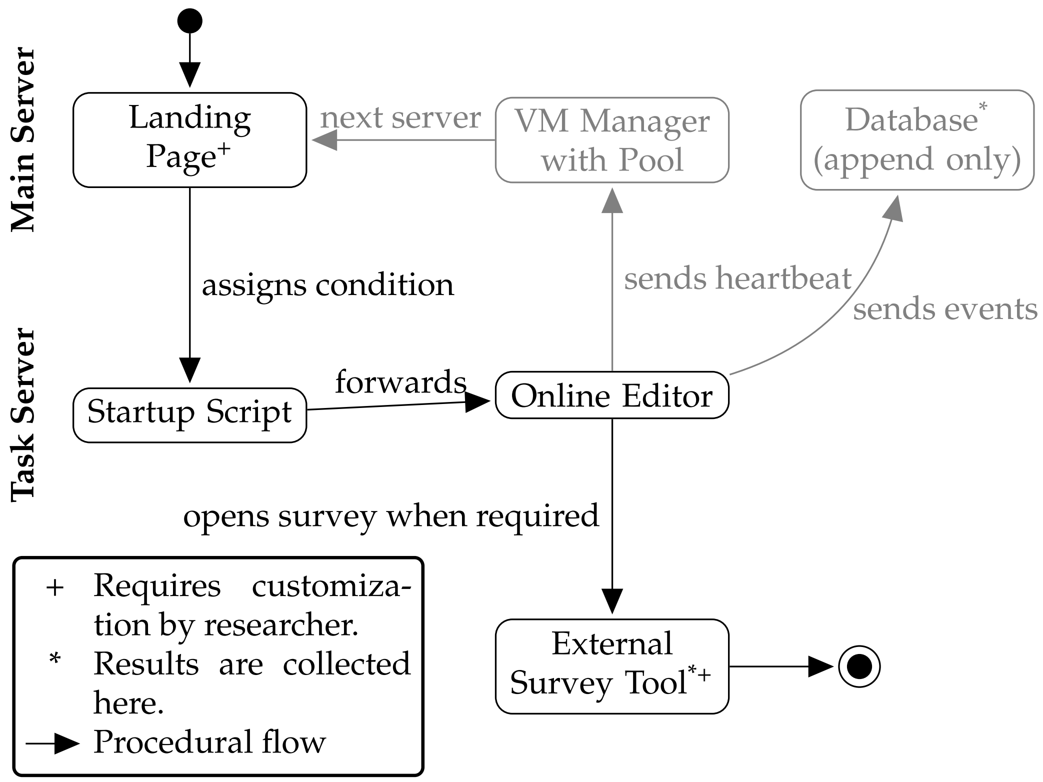


FIGURE 2.1: The different components of the Developer Observatory. The black boxes and arrows illustrate the navigation path of a participant (note that the grey components do not).

the study infrastructure as a platform to send spam or cause other harm. If a participant does disrupt the infrastructure (for example by writing code that creates an infinite loop), this disruption should affect only that participant and not extend to other participants. This requirement is complicated by the fact that participants will by definition be writing and executing code (possibly under a privileged user account).

Participant Privacy. Participant privacy is crucial, both as an ethical matter and as a practical reassurance to participants that their cooperation is low-risk. The framework should allow either anonymous or pseudonymous participation. In the pseudonymous case, participants should be identified by a coded ID which is stored separately from the email address (or other contact data) used for invitations. When desired, researchers can map secondary information (such as the channel via which a participant was recruited) to the coded ID.

2.4 Developer Observatory Design

To meet these requirements, we designed a framework (cf. Figure 2.1) with five main logical components (many of which can be co-located): a landing page, a VM manager, a task server with an online code editor, a database, and a survey tool.

A link in an invitation email or advertisement directs participant to the study's landing page. Here, information about the study is offered: The researchers and their institutions are identified so that participants can contact them in case of questions, the purpose of the study is explained in as much detail as possible without priming the participants and introducing bias, and a downloadable consent form is presented that details how information obtained in the study is obtained, processed, stored and published in an ethical and privacy-preserving way.

After consenting to the study, participants are assigned a condition and a VM instance. Conditions can be assigned round-robin, or with weighted probabilities (e.g., to account for conditions with higher dropout rates). The task order can be randomized when appropriate.

The VM manager creates instances of the task server and provides the next available instance for the correct condition to the participant. It is also responsible for maintaining a small queue of ready VMs, so that participants do not have to wait for a new instance to boot up, and for shutting down VMs that are no longer considered active according to the study policy.

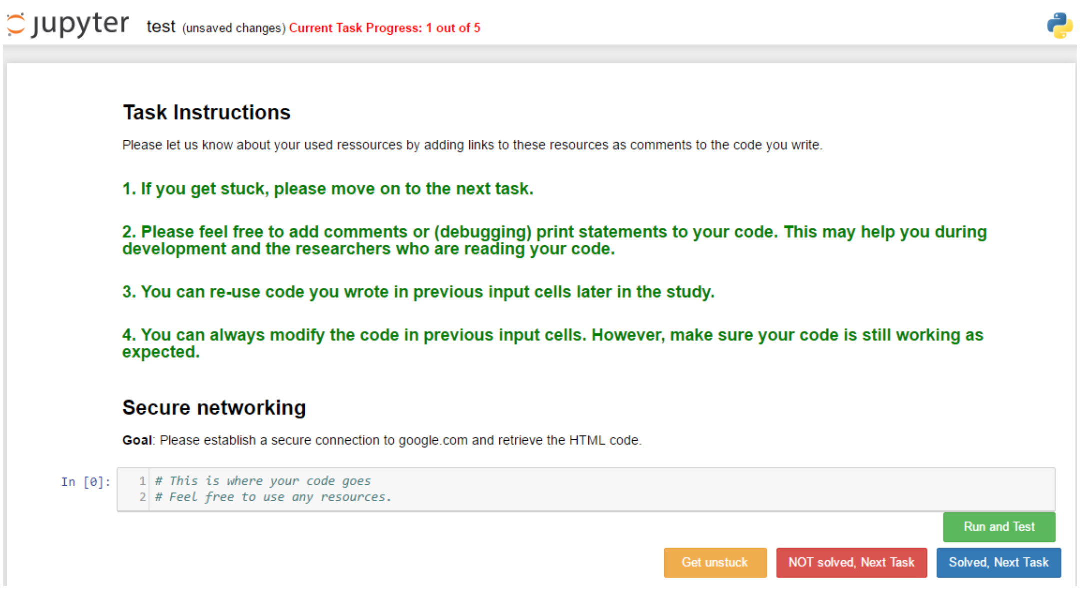
Participants are then forwarded to the task server, which hosts an online code editor. The editor supports writing and executing source code, as well as showing results and errors to the participant. Task instructions are provided inline, so that participants can read the instructions and work on their code in the same interface. Participants are also able to test their code and view output, as well as recover from infinite loops and kernel crashes. Once participants are ready to move on, they can indicate whether they are completing or skipping the task. We expected that offering the option to skip a task would decrease the number of participants who quit the study in frustration because they could not solve a particular task.

The task server has several important features that address the requirements described in Section 2.3. These include data collection features, such as detection of copy-paste events, test runs, task completions, and the collection of snapshots of the code at each important event. The task server also enables popover messages and/or short pop-up surveys triggered by particular events; for example, a copy-and-paste event can trigger a reminder to document the source of any pasted code.

Once a participant has completed all programming tasks, they can be forwarded to an exit survey. This exit survey might include questions about the tasks they worked on, their demographics and background, and any other relevant information. Information from the programming tasks and the exit survey is linked via a participant's unique pseudonym ID so that participant's code can be imported to help the participant remember their work while answering.

2.5 Implementation

In this section, we discuss our implementation of the Developer Observatory. Our implementation relies on Amazon EC2, which provides reliability and scalability.



Task Instructions

Please let us know about your used ressources by adding links to these resources as comments to the code you write.

1. If you get stuck, please move on to the next task.

2. Please feel free to add comments or (debugging) print statements to your code. This may help you during development and the researchers who are reading your code.

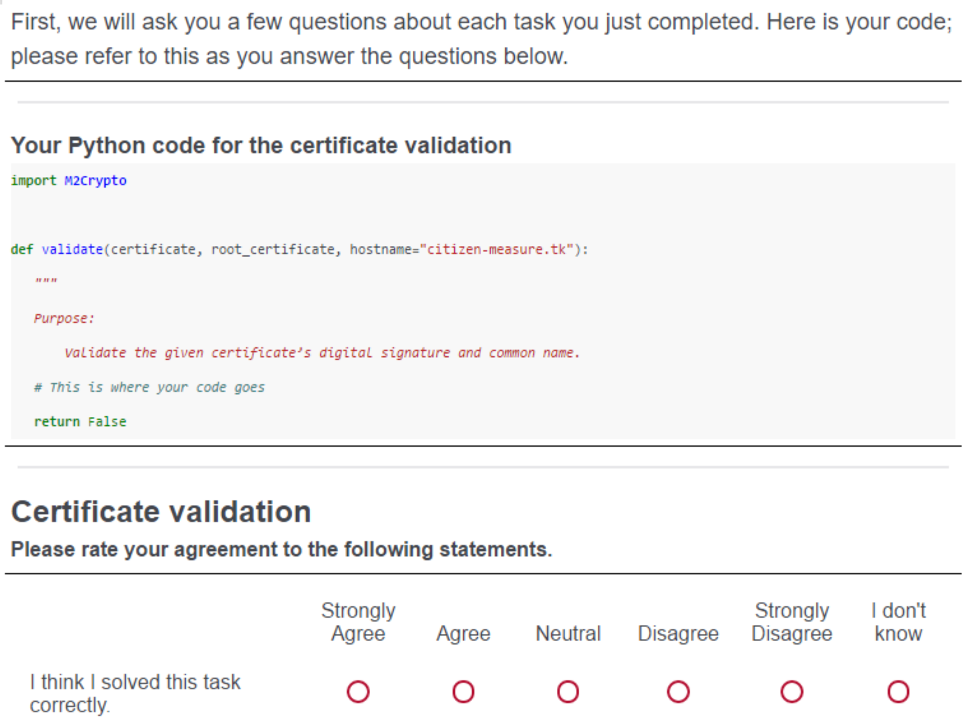
3. You can re-use code you wrote in previous input cells later in the study.

4. You can always modify the code in previous input cells. However, make sure your code is still working as expected.

Secure networking

Goal: Please establish a secure connection to google.com and retrieve the HTML code.

```
In [0]:
# This is where your code goes
# Feel free to use any resources.
```



First, we will ask you a few questions about each task you just completed. Here is your code; please refer to this as you answer the questions below.

Your Python code for the certificate validation

```
import M2Crypto


def validate(certificate, root_certificate, hostname="citizen-measure.tk"):
   """

   Purpose:

       Validate the given certificate's digital signature and common name.

   # This is where your code goes

   return False
```

Certificate validation

Please rate your agreement to the following statements.

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree	I don't know
I think I solved this task correctly.	○	○	○	○	○	○

FIGURE 2.2: Screenshots from one study conducted using Developer Observatory. Left: Participants completed tasks using a Jupyter Notebook as an online code editor. Task instructions and skeleton code were provided. Right: We used the Qualtrics web service feature to show each participant their code for the task being asked about.

We use one EC2 instance to host the landing page, VM manager, and database. The VM manager creates and assigns participants to EC2 *micro* instances: these are cheap instances with limited resources, but they are sufficient to run the online code editor. The database is stored on a separate PostgreSQL server.

The VM manager keeps a small pool of already-prepared instances for assignment to new participants. A VM instance requires about one minute to boot, so without this pool, participants would be forced to wait unacceptably before beginning the study. As one instance in the pool is assigned to a participant, a new instance is started in order to maintain the pool size. For example, in our first study using Developer Observatory [7], we found we needed a pool of 10 instances to keep up with demand. For a subsequent, smaller study, we used an initial pool size of five, which we reduced to three as participation slowed. This generally proved sufficient to prevent potential participants from waiting. [9]

To manage outstanding instances, we use a heartbeat signal. An assigned instance will send a heartbeat to the VM manager every 60 seconds, starting when that instance is assigned to a participant, as long as the browser window is open and the online editor is operational. An assigned instance automatically shuts itself down after a participant finishes all tasks; the VM manager can also apply a shutdown policy to kill the instance if it has been open too long. In our cryptographic API study, we experimented with a few shutdown policies before finding that killing an instance after four hours with no heartbeat provided a good tradeoff between wasting instances and allowing participants to complete tasks in their own time. We applied this policy for our recruitment study as well. We note that Amazon limited us to a maximum of 50 concurrent instances, so managing shutdown became important during times of high demand.

As our online code editor, we used Jupyter Notebook 1.0, an in-browser editor for writing and executing Python code.[1] Each programming task consists of two Jupyter *cells*: one *markdown* cell that provides instructions for the task, and one *code* cell that provides skeleton code for the task itself or to test it. Participants then add and edit code directly in the code cell and run their code with the run button. A screenshot from the cryptographic API study is shown in Figure 2.2, left.

When a participant is assigned to an instance, the VM manager provides the assigned condition to the instance as a parameter. The appropriate task file(s) for the assigned condition are then copied into place, with a generic name, so that the filename does not provide extra information about the study goals and conditions.

After all tasks have been completed or skipped, the participant is redirected to the exit questionnaire. Thus far, we have used Qualtrics and LimeSurvey to host exit questions. To help the participant remember their work, Developer Observatory can show participants' code for a given task next to questions about that task (cf.

[1] Jupyter supports kernels for most major programming languages: `github.com/ipython/ipython/wiki/IPython-kernels-for-other-languages`

Figure 2.2, right). To achieve this, we used the Qualtrics Web Service module and a JavaScript solution for LimeSurvey.

Data Collection. We configured Developer Observatory to capture the current state of the participant's code at each click of the run button. This reflects the fact that many developers add functionality incrementally, test, and add and remove debugging output as they work. By capturing snapshots of this process, we can examine how the code was developed and how many test runs were associated with different tasks. We store the result of each run, including return codes, error codes, exception messages, and stack traces.

We also gather copy-and-paste events to measure the fraction of code our participants write themselves versus copy-and-paste from information sources such as official API documentation, Stack Overflow, and blog posts. Additionally, participants are encouraged to report the URL code was copied from. The text associated with copy-and-paste events is available for later analysis in order to distinguish code copied from information resources from code being moved around within the participant's work.

All events that we capture are logged with corresponding timestamps; in addition, we track timestamps when each task begins and when the participant chooses to end the task (by indicating success or by skipping the task).

Developer Observatory is designed to be easily extensible for features that may be important to other researchers. For example, any event detected by JavaScript can be recorded and/or used to trigger a code snapshot. For example, a researcher interested in how developers scroll up and down within their code could add a listener for scroll events that triggers an AJAX request.

Security. Allowing unknown users to execute arbitrary code poses a severe security risk: Malicious users could try to overallocate resources, crashing the infrastructure; try to read other participants' data from the database or delete existing study data; use the infrastructure as a platform to send spam; or other problems.

We manage these risks in several ways. We use EC2 to ensure that damage caused by an attacker to the server hosting the online code editor is limited only to that participant's instance, rather than affecting other participants or the infrastructure more broadly. Additionally, we strictly limit the software installed on the task server instance, minimizing the harm that an attacker who breaks out of the Jupyter notebook can achieve. The only common resource for all instances is the interface to the database, which limits the data it accepts.

We implement public-key pinning for certificate validation for the communication between instances and the database server. Using EC2 security groups, we ensure that the task servers can only create outgoing connections to our database server, nowhere else. Unfortunately, the Jupyter notebook on the task servers is only reachable for participants via HTTP because the hostnames are assigned by Amazon's EC2 and we cannot retrieve a valid certificate for Amazon's EC2 subdomains.

Data transfer from the task to the database server is size-limited and sanitized at both ends.

Finally, we use Google's reCAPTCHA service[2] at the landing page to make it harder for an attacker to create many instances and exhaust the EC2 allocation.

Configuration. All configuration options for our tool are maintained in a single configuration file on the landing page server. The configuration file controls the VM pool size and identifies the type of instances that should be created for the pool (in our case, micro instances). The configuration file also provides an upper bound for total number of participants, so the experiment will stop if a maximum limit (such as one specified by an ethics committee) is reached.

The configuration file also manages several security items, including Amazon EC2 keys and security-group ID, a registration key for reCAPTCHA, and database login credentials. It also supports the use of a one-time token system to limit each invited participant to only starting the study once.

Meeting our Requirements. In Table 2.1, we summarize how Developer Observatory achieves the requirements described in Section 2.3.

2.6 Using Developer Observatory

We successfully used Developer Observatory to run controlled experiments for two studies of Python developers. Below, we provide an overview of these studies, an evaluation of the efficacy of our tool for running them, and a discussion of our lessons learned. Both studies were approved by all our institutions' review boards.

Overview of Studies. We first used Developer Observatory for an online, between-subjects study comparing how effectively Python developers could write correct, secure code using different cryptographic libraries (the *API study*) [6]. We recruited developers from GitHub and assigned them programming tasks using one of five conditions and one of two sets of encryption tasks. Assignment to conditions and task sets was initially random, with counterbalancing to ensure roughly equal numbers of participants started in each condition. However, when it became clear that dropout rates (starting but not completing the study) varied widely among conditions, we used Developer Observatory to weight the assignment to favor under-populated conditions. Within each condition, task order varied using Latin-Square assignment. After finishing the programming tasks, participants completed a brief exit questionnaire with questions about the code they had just written, the assigned library, their prior programming and security experience, and demographics.

[2]cf. `https://www.google.com/recaptcha`

TABLE 2.1: The requirements we specified, and how they are met by Developer Observatory.

Validity	
Internal	Our framework serves all participants the same VM image, which comes with the prescribed development environment (i.e., pre-installed Python version and libraries) and the same online editor based on Jupyter. We allow for the random assignment of participants to conditions, the prevention of duplicate participation via a token system and reCAPTCHA, and we enable the consecutive display of randomized tasks.
Ecological	Our framework allows participants to solve programming tasks from home, their place of work, or their favorite coffee shop, using their own computer.
Data collection	Each code execution triggered by a participant is stored on the database server, along with the execution result and a time stamp. Additionally, copy-and-paste events are stored. Qualtrics stores questionnaire data.
Interaction	
Participants	Participation in a study using our framework does not require participants to download programming tasks or install any software such as an editor or libraries. Instead, they can participate using their web browser. Participants are directed through study tasks without needing to follow complicated directions.
Researchers	Setting up a new study is straightforward. Once the main server (hosting the landing page, VM manager, and database) has been installed and configured, very little must change to run a new study. Only the consent form, tasks and associated condition assignment plan, and exit questionnaire must be specified, and some configuration options (such as VM pool size) can be updated. Data can be retrieved from the database and Qualtrics in CSV or SQL format; the two sets of data are easily linked via participant ID.
Technical	
Reliability	We use Amazon's EC2 infrastructure to ensure a high level of reliability for virtual machines.
Scalability	Amazon's EC2 infrastructure can scale up well beyond our requirements.
Participant Isolation	Participants are each assigned their own virtual machine, guaranteeing strong isolation and preventing interference between participants.
Security	VMs that serve tasks are firewalled and only allow incoming network traffic via HTTP. Outgoing traffic is limited to the HTTPS port of the database server. Data storage requires strong authentication using a token system, and the database account is limited to only inserting new results.
Privacy	Participants are assigned a random ID after agreeing to the consent form. The mapping between participant email addresses and their random ID is stored on a different server that is not accessible from the tool.

Later, we used Developer Observatory to compare correctness and security results for professional and student developers (the *recruitment study*), again recruiting participants from GitHub [9]. Here, no conditions were assigned (other than self-reported student/professional status post-hoc). Participants completed three randomly ordered security-relevant tasks and a brief exit questionnaire similar to the questionnaire from the API study.

Overall, 2396 participants accepted the consent form and started one of the studies (API: 1571, recruitment: 825). Of these, 563 (API: 256, recruitment: 307) completed all tasks. We note that the dropout rate was significantly higher for the crypto API study (84%) than the recruitment study (43.6%) ($X^2 = 130.5$, $p < 0.0001$). We believe this difference is primarily attributable to the tasks in the API study being harder: in the API study, participants achieved functional solutions for only 64% of all tasks, compared to 88% in the recruitment study.

Participants who did not drop out spent median 56 minutes (Q1 = 35, Q3 = 106) on the programming tasks in the API study and 53 minutes (Q1 = 30, Q3 = 85) on the programming tasks in the recruitment study.

Participant Experience. In both studies, participants provided feedback in free-text questions within the exit questionnaire, by leaving their opinions commented into their task code, and by emailing us. Overall, most feedback was positive. A few participants wrote us when they were denied access to the study during an overload incident (described below) or an AWS outage. Nine participants complained about using the online editor; most reported falling back to a local editor, then pasting their code into the Notebook. Seven API-study participants reported annoyance with a popover message that appeared after each copy-and-paste event reminding the participant to document the source of any reused code; we reworked the popover to not show after copy-and-pasting within the editor field and received no further complaints. A few participants mentioned that Python libraries we included in the Notebook were insufficient or out of date (easily fixable for the next study), and two had trouble with LimeSurvey. A few potential participants were disappointed the platform didn't work in a mobile browser.

Researcher Experience. From our perspective as researchers, the studies ran (mostly) smoothly. We easily retrieved and analyzed the code our participants wrote, their copy-paste events, and data about how frequently they pushed the "run" button. We were also able to quickly link participants' code to their exit questionnaire responses and analyze all facets of their results: whether their code was functional, whether it met our security criteria, and how their self-reported responses correlated with their coding results.

Compared to running a developer study in the lab [7], using Developer Observatory we found recruiting and running study participants to be easier and faster. Because we couldn't ask participants to think aloud, we missed out on some qualitative data. However, participants left many detailed and passionate responses both in the comments to the code they wrote and in free-text portions of the exit questionnaire, so we did collect interesting qualitative nuggets. Our measurements of participants' code were as good as in a lab study, but timing data was less precise, because participants stepped away from and then resumed tasks. However, we found that online participants were willing to commit as much or more total time as lab participants, despite not being compensated, perhaps because they could work on their own schedules from the comfort of their preferred environment. Overall, we successfully collected and analyzed detailed data about participants' behaviors.

Our biggest challenges using Developer Observatory were managing the pool of VM instances and calibrating how many recruitment emails to send to avoid overloading the system (see below). After the first week, however, we became proficient, and managing these resources required little additional effort.

Managing Participants and Resources. Initially, instead of the heartbeat feature, we expired an instance four hours after it was assigned to a participant. Many potential participants opened the online code editor before deciding not to participate; these instances remained idle for four hours. Given our initial (default for EC2) limit of 20 instances running simultaneously, our available instances were quickly exhausted, and some potential participants were rejected due to lack of resources. We solved this problem by implementing heartbeats, by requesting an increase from Amazon to 50 concurrent instances, and by manually emailing participants who were rejected to re-invite them. The original four-hour timeout also inconvenienced a small number of participants who were cut off while still working on their code. We manually restored these participants' sessions and emailed them a link to continue. Adding the heartbeat feature solved this issue.

We also made interesting observations about timing. For the most part, we periodically sent invitation emails and potential participants arrived at the landing page in a fairly even stream. However, we once sent emails periodically over a weekend. We saw almost no traffic during the weekend, but then received a huge influx of participants on Monday morning, presumably because invited participants had arrived back at work and were all checking their email at a similar time. This rush nearly overloaded our infrastructure: A few participants were unable to start the study. We recognized the issue, freed up unused instances, re-invited the participants who had experienced issues and paused invites until the rush had evened out. For future studies, we recommend not sending large numbers of invitations immediately before or during a weekend or holiday.

Future Work. In future work, we plan to expand and improve Developer Observatory. For researchers, we plan to add a web-based configuration and management tool that makes it easy to set up study parameters, and then monitor participants in progress, from one graphical interface. This interface should also allow researchers to retrieve data from both programming tasks and the exit questionnaire in one step. We would also like to add further instrumentation for collecting participants' behavioral patterns while writing code, and perhaps to allow a study design that switches back and forth between coding tasks and survey questions as many times as necessary. We would also like to add an explicit feature allowing participants to pause and resume sessions at their convenience, without worrying about closing their browser or having their instance killed by the VM manager as inactive. Furthermore, we plan to add an automatic invitation system, which invites new participants depending on the current workload as well as sending re-invites to participants who were rejected because no resources were available. We also plan to support other researchers who wish to extend Developer Observatory, and will open-source Developer Observatory as well as provide our contact information for support.[3] While we did not use debugging during the study, it is possible to extend Jupyter with debugging functionality using ipdb[4] to give developers an additional tool that they might normally use.

Takeaways. Security experiments in which developers are asked to write code are increasingly common. We developed Developer Observatory, a distributed, online platform for administering security programming studies to developers remotely. We found that Developer Observatory allowed us to recruit professional and hobbyist developers, from all over the world, faster and more easily than for a lab study. This allowed us to increase the power and generalizability of our experiments, with little to no loss of internal validity.

We designed Developer Observatory both to meet our immediate needs and to be extensible to other security experiments. Our experience suggests that interaction with Developer Observatory was satisfactory both for us as researchers and for most of our participants. We hope that by providing Developer Observatory as an open-source tool for the security community, we can help increase researchers' access to developers, lessen the inconvenience of study participation for developers, and increase the scale and validity of future research.

To understand how cryptography can meaningfully and securely be implemented in software, researchers need to understand how developers interact with cryptographic libraries. Studying expert populations is challenging in many ways: due to geographical, cost, and time restrictions, it can be hard to recruit developers into research laboratories. Therefore, in this chapter, we presented an online platform that allows for programming experiments with developers. This chapter demonstrated that online studies with developers can be conducted at scale and lead to meaningful, externally valid research results with professional developers as participants, with high internal validity. The study environment allowed us to understand

[3]`https://github.com/developer-observatory/developer-observatory`
[4]`https://pypi.python.org/pypi/ipdb`

many challenges developers face when implementing encryption. For example, we found usability problems with many common Python cryptographic libraries, specifically issues concerning feature completeness and documentation. We also found that student participants can contribute useful insights to this type of research, and that programming experience is an important correlate to secure use of cryptographic libraries.

With this understanding of how to support research on use of cryptography by developers, the remainder of this thesis focuses on how to support end-users. Proceeding in this research direction is based on the understanding that secure implementation of encryption is only as useful as it is widely used by end users. If end users fail to adopt encryption, for example because of usability issues, they fail to reap the benefits of potentially secure communication. For the purpose of understanding how end-users use encryption, we were able to analyze a university-wide data set of emails, to measure use of email encryption, as well as identify challenges. This exploration of real-world use of email encryption is detailed in Chapter 3.

Chapter 3

27 Years and 81 Million Opportunities Later: Investigating the Use of Email Encryption for an Entire University

Disclaimer: *The contents of this chapter were previously published as part of the conference paper "27 Years and 81 Million Opportunities Later: Investigating the Use of Email Encryption for an Entire University", accepted for publication and to appear in the IEEE Symposium on Security & Privacy, May 2022. This research was conducted as a team with my co-authors Oliver Wiese, Yasemin Acar, Volker Roth, and Sascha Fahl; this chapter therefore uses the academic "we". The idea for this research was conceived by me and Oliver Wiese. I coordinated access and processing of the data with the data protection officer, as well as the data center. The code for the evaluation script was co-written by me and Oliver Wiese; we jointly conducted the data analysis. All authors jointly wrote the paper. Yasemin Acar, Volker Roth, and Sascha Fahl supervised throughout.*

Abstract

Email is one of the main communication tools and has seen significant adoption in the past decades. However, emails are sent in plain text by default and allow attackers easy access. Users can protect their emails by end-to-end encrypting them using tools such as S/MIME or PGP.

Although PGP had already been introduced in 1991, it is a commonly held belief that email encryption is a niche tool that has not seen widespread adoption to date. Previous user studies identified ample usability issues with email encryption such as key management and user interface challenges, which likely contribute to the limited success of email encryption.

However, so far ground truth based on longitudinal field data is missing in the literature. Towards filling this gap, we measure the use of email encryption based on 27 years of data for 37,089 users at a large university. While attending to ethical and data privacy concerns, we were able to analyze the use of S/MIME and PGP in 81,612,595 emails.

We found that only 5.46% of all users ever used S/MIME or PGP. This led to 0.06% encrypted and 2.8% signed emails. Users were more likely to use S/MIME than PGP by a factor of six. We saw that using multiple email clients had a negative impact on signing as well as encrypting emails and that only 3.36% of all emails between S/MIME users who had previously exchanged certificates were encrypted on average.

Our results imply that the adoption of email encryption is indeed very low and that key management challenges negatively impact even users who have set up S/MIME or PGP previously.

3.1 Introduction

Email is one of the major online communication tools. As of February 2021, there are more than 4 billion email users worldwide sending and receiving over 300 billion emails per day [143]. While email is used for all kinds of information including the most sensitive kinds such as trade secrets, account credentials, and health data, regular email is not encrypted and allows network attackers and service providers unauthorized access. This is not for a lack of tools. Both S/MIME [40] and PGP [164] were introduced almost 30 years ago with the goal to provide end-to-end encryption for email. However, in contrast to modern messaging tools such as Signal [130] or WhatsApp [157] that implement end-to-end encryption by default, S/MIME and PGP require a complex manual setup by users. Consequently, previous work has shown that using email encryption correctly and securely is challenging for many users [63, 116, 118, 119, 127, 160]. They struggle with setting up and configuring encryption keys, distributing them, managing keys on multiple devices, and revoking them. These findings, already anticipated by Davis [30], are corroborated by public reports of failed PGP use. For example, it took Edward Snowden and the journalist Glenn Greenwald a few months and serious effort to set up PGP for email in order to communicate securely [101]. Hence, it is commonly believed in the security community that end-to-end encrypted email is not widely used, mostly because of lacking usability and awareness issues identified in a multitude of user studies in the past 22 years (cf. [27, 63, 64, 65, 118, 120, 127]). To the best of our knowledge, our work is the first scientific collection and evaluation of ground truth on the adoption of end-to-end email encryption. Our work is mainly motivated as follows:

Ground Truth We aim to confirm the security community's anecdotal knowledge about the low adoption of end-to-end email encryption and provide ground truth based on field data. Our longitudinal field data can help motivate future work to improve the adoption of end-to-end encryption for email.

Method Extension We extend the toolbox of the past 22 years of email encryption research that was initiated with the seminal paper "Why Johnny Can't Encrypt" [160] at USENIX Security'99 that is mostly based on laboratory experiments and self-reporting studies: In this work, we investigate a large dataset including millions of data points of thousands of users and years of their email data.

Validate Results from Previous Work and Investigate Underexplored Challenges We confirm findings from previous work (e.g. [3, 83, 84, 120]) obtained by other methods including smaller-scale interviews, surveys, and controlled experiments. Additionally, we also investigate further challenges that require large scale field data.

Motivated by the above, we make the following contributions in the course of this work:

Data Collection Pipeline In collaboration with our data protection officer, university staff council, and the technical staff of the university IT department, we developed and tested a reproducible and privacy friendly data collection pipeline that allows to analyze large amounts of email data with a focus on S/MIME and PGP usage (cf. Section 3.4.1). The data collection pipeline is part of our replication package. We aim to encourage other institutions to investigate their adoption of S/MIME and PGP to contribute to an even better understanding of the email encryption ecosystem.

Adoption of Email Encryption at a Large University We provide a detailed evaluation of the adoption of email encryption at our university in the past 27 years. In our evaluation, we focus on the use of S/MIME and PGP for 37,089 total email accounts. Our investigation of 81,612,595 emails found that 2.8% of them were digitally signed and 0.06% were encrypted. We found that only 5.46% of our users ever used S/MIME or PGP and that S/MIME was more widely used than PGP. However, PGP was the more popular email encryption tool amongst researchers.

Use of S/MIME and PGP We provide a detailed overview of S/MIME certificates and PGP keys in our dataset and find that RSA is the most widely used key algorithm, employing 2048 bits keys most often for S/MIME. PGP keys used 4096 bits most often, although newer PGP keys used less secure 2048 bits. We find that more than one third of all PGP keys did not have an expiration date set making revocation unnecessarily complicated and *Deutsche Telekom* to be the root CA for 64.95% of all S/MIME certificates.

User Interaction Challenges including Key Management We report on an investigation of user interaction challenges that previous work identified in user studies. Most interestingly, we focus on key management issues during key distribution, multi device use, and key rollover. We find that even after exchanging public keys, only 3.36% of all emails between S/MIME users were encrypted on average. The use of multiple email clients had a negative impact on the amount of signed and encrypted emails. While most S/MIME and PGP users renewed their keys in time, 11.5% of S/MIME key rollovers occurred after the keys' expiration.

Overall, our results confirm the pessimistic assessment of the security community: Although our university provides all researchers, staff, and students with free access to S/MIME certificates, only very few make use of them and only a negligible amount of emails was encrypted or signed. Our findings also support results from previous user studies and illustrate additional challenges. Management of email encryption keys is hard and distributing keys, using multiple email clients, or having to renew keys complicates matters.

The rest of this work[1] is organized as follows: In Section 3.2 we provide information on S/MIME, PGP, and our university's S/MIME certificate authority. We provide an overview of related work and contextualize our contributions in Section 3.3. In Section 3.4, we describe our methodology by providing details for our data collection pipeline, discussing ethical and data privacy implications of our work, illustrating limitations, and summarizing the replication package. In Section 3.5, we provide detailed results of our evaluation, discuss their implications in Section 3.6, and conclude this chapter in Section 3.7.

3.2 Background

In this section, we provide background information on OpenPGP, S/MIME, and the email ecosystem of our university including its S/MIME certificate authority.

[1]Find our companion website at: `https://publications.teamusec.de/2022-oakland-email/`

3.2.1 OpenPGP

OpenPGP[2] is an encryption standard (cf. [25, 43]) which is used for email encryption and digital signatures. PGP is an open source project and was first standardized in 1996. In the first standardization, PGP messages were added to the text body (named: PGP Inline) of an email. Later versions introduced a separate MIME type for PGP messages (named: PGP MIME). Over time, new algorithms have been added, including the Camellia and ECDSA cryptography algorithms. PGP supports the use of key servers for public key exchange. Users can search these servers for keys for given email addresses. However, keys may also be exchanged by attaching public keys to emails. Additionally, several email clients, like K9 on Android or Thunderbird using the Enigma plugin, support hidden key exchange by adding public keys to email headers. This feature has been standardized and further developed by the open source project Autocrypt[3] since 2016.

In contrast to centralised trust infrastructures known from the web PKI or S/MIME, PGP relies on the Web of Trust to verify identities. In the web of trust approach, users sign each other's key when meeting other PGP users in person. Therefore, users can trust a new key if another trusted key previously signed the new key, relying on a decentralized trust chain.

3.2.2 S/MIME

Secure/Multipurpose Internet Mail Extensions (S/MIME) is a standard to encrypt and sign emails. It was first introduced in 1998 (RFC2311 [43]) and has constantly been improved since then. S/MIME utilizes a Public Key Infrastructure to verify certificates and as such has mostly been used in corporate environments, where a certificate authority (CA) is deployed or third party CAs are utilized to issue certificates to employees. It has been widely supported out-of-the box without the need for third party plugins in commercially used email clients like Outlook 98 and higher or Thunderbird.

3.2.3 Email Ecosystem at Leibniz University Hannover

Email at our university is a centralized service. The university's computing center provides email accounts for all administrative staff members, for all students as well as faculties, departments, and research groups. Overall, our university offers 90 different study subjects reaching from engineering to humanities, and has about 30,000 students and 5,000 staff members. Faculties, departments and research groups are organized in decentralized units, i.e. each faculty, and most departments and research groups have their own subdomain (e.g., sec.uni-hannover.de for the information security research group) for their email. Users can access their email

[2]Abbreviated as PGP in the paper

[3]cf. `https://autocrypt.org`

accounts either through a web interface or dedicated email clients using the university's POP3, or IMAP and SMTP servers.

3.2.4 University Certificate Authority / Registration Authority

Our university is part of the public key infrastructure of the communications network for science and research in Germany (DFN).[4] The university's computing center provides a registration authority for the DFN CA to issue certificates for email end-to-end encryption and signing, server authentication for TLS, and document signing for its scientific and administrative staff and all students (DFN-PKI).[5] Certificate signing and revocation is processed through the DFN CA.

Certificate Policies All university employees and students are eligible to obtain S/MIME certificates. However, certificate use is neither officially endorsed, nor are issuances automatically triggered. Individual work groups may informally encourage certificate use. While the CA also provides server certificates (e.g., for TLS), our work focuses on user certificates for email encryption/signing.

User Certificates To apply for a certificate, eligible staff and students can apply online, receive a certificate signing request, make an appointment with the registration authority, show up in person, present a proof of identity, and then receive a valid certificate. This process is comparably complicated. In contrast, creation of a student ID that can be used to access free transport and student discounts, and is used as proof of identity in exams, does not require in-person interaction. The process is also not embedded in any other existing onboarding process at the university. New certificates for the same user can be issued without another identification if the last identification is not older than 39 months.

Certificate Signing Request Process Users can generate a certificate signing request (CSR) for email certificates using a web application entering their personal details and a revocation pin. The web application generates a CSR, saves it in the browser certificate store, and asks the user to enter a passphrase which protects the CSR's private key.

Certificate Revocation Process The CA provides a web interface for certificate revocation. To revoke certificates, users have to enter their certificate's serial number and a revocation PIN.

Certificate Expiration Once 30 days and a second time 15 days before the user certificate expires, the DFN sends users an expiration warning via email and encourages users to renew the certificate.

[4]https://www.dfn.de/en/

[5]`https://www.pki.dfn.de/ueberblick-dfn-pki/` (german)

History and Milestones Our university began to issue S/MIME certificates in 2004. The first root certificate used to sign the original CA (G1) through the DFN expired in 2019. Starting from 2017, certificates were issued using a new root CA (G2). Additionally, 479 certificates issued using the G1 CA had to be replaced with new certificates issued by the G2 root CA. Figure B.3 in Appendix B.2.1 illustrates the issued certificates per year. In the first years, the amount of server and user certificates is roughly equal. However, the number of S/MIME certificates is only slowly increasing compared to the number of server certificates. We can see a drop in requested certificates in 2020, probably due to the COVID-19 pandemic.

3.3 Related Work

Note: This section assumes that you have read the related work section in section 1.3. It is slightly shortened from the original version of the paper.

We discuss related work in two key areas and put our work into context: User studies for end-to-end encryption with end users and email field studies.

3.3.1 End-to-End Encryption Studies for End Users

The usability of end-to-end encryption has been a research focus for decades. Following Whitten and Tygar, Garfinkel and Miller [63] and Roth [111] hypothesized that some of the challenges could be overcome by simplifying the key verification process. Their solutions were based on omitting the verification of keys by third parties and instead using a trust on first use (TOFU) approach. Garfinkel and Miller tested their approach on lay users and found that color-coding messages depending on their signature status makes users significantly less susceptible to social engineering attacks overall. Garfinkel et al. surveyed 470 merchants who received digitally-signed VAT invoices from Amazon and found that merchants should send signed emails by default as the passive exposure seems to increase acceptance and trust [62, 64]. Ruoti et al. studied different self-designed and publicly available encryption tools [115, 116, 117, 119] to improve the usability of email encryption in their laboratory. Their research interests were key management, key distribution, and automatically vs. manually enabled encryption. In several laboratory studies by Ruoti et al. [115, 116, 119], participants rated at least one tool as usable and indicated interest in secure email, but did not know when or how they wanted to use it. In another recent journal article, Ruoti et al. came to the conclusion that secure and usable systems so far had only been tested in short-term studies and future research should investigate long term usability and adoptability of secure email systems [120]. Atwater et al. and Lerner et el. proposed clients for PGP similar to Keybase[6] to study how to simplify key distribution [17, 79]. They proposed to upload users' public keys to a website and confirm ownership such that they can retrieve emails from the

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

corresponding email address. In a lab study, Atwater et al. found that such a key distribution mechanism enabled participants to send more encrypted emails and had improved usability for (webmail) users. Lerner et al. compared their tool called Confidante with Mailvelope[7] and showed that their tool reduces the error potential [79]. In a lab study, all participants in the Mailvelope group struggled to import public keys or to share their own public key. In the Confidante group, three out of nine participants struggled to import public keys. However, all of them managed to share their public keys successfully. Bai et al. proposed encryption prototypes to study user flows of different key management approaches [20, 21]. In an interview study, participants preferred to register their public keys on a webpage and automatically retrieve public keys of communication partners from the webpage over manual key management. Fahl et al. examined different usability aspects for Facebook message encryption mechanisms and found that automatic key management and key recovery capabilities are important for adoption [49]. McGregor et al. reported that cooperating journalists used PGP to encrypt their emails when investigating the Panama Papers [84]. However, journalists also identified obstacles when using encryption with multiple devices. Consequently, they used secure messengers such as Signal instead of PGP on their smartphones. Gaw et al. interviewed nine employees from the same company and found that users flagged encrypted mails as urgent and found those to be annoying when used for all messages [65]. They argued that understanding of social factors is important for adoption. In a combined lab and field study, Mauriés et al. participants struggled with Enigmail for Thunderbird and the Mailvelope browser plugin [83]. Enigmail users needed help to setup the tool on their computer. The setup process in Mailevelope was unclear. One participant struggled to import a public key and send an encrypted email.

In addition to email, mobile messaging apps including Signal, Threema and WhatsApp, made end-to-end encryption available for the masses. WhatsApp, for example, introduced end-to-end encrypted messages for all its users by default in 2016 [157]. In an interview study, Abu-Salma et al. identified blockers and barriers for the adoption of end-to-end encryption including incompatible tooling and misconceptions of end-to-end encryption features [4]. They argue that usability is not the primary obstacle and that fragmented userbases or a lack of multi-device support significantly contribute to the non-adoption of end-to-end encryption. In a different study, Abu-Salma et al. further explored users' mental models and found misconceptions about security properties of messengers [3]. They argue that adoption is no longer the main challenge for end-to-end encryption tools but that people instead switch to non secure communication tools and need assistance in choosing the right one for sensitive information. Stransky et al. confirmed these findings in an online study with WhatsApp users. They found that security perceptions of end-to-end encryption in mobile messaging apps heavily depended on the reputation and expectations of an app provider, while visualizing encryption has only limited impact

[7]Mailvelope is a browser plugin for PGP: `https://www.mailvelope.com/`

on perceived security [137]. Similarily, Akgul et al. found that participants noticed educational messages and that they improved understanding of security concepts when they are used in isolation. However, when those messages were implemented in a realistic environment, they could not find significant improvements in the mental models of end-to-end encryption of users [12].

Overall, previous work primarily focused on identifying blockers and barriers to adopting end-to-end encryption for email or mobile messaging apps and studied alternatives to or extensions of existing approaches using laboratory and interview studies. In contrast, we aim to extend the toolbox of end-to-end encryption research and provide ground truth based on longitudinal field data by evaluating a large dataset including millions of data points of thousands of users and years of their email data.

3.3.2 Email Field Studies

In addition to the user studies discussed above, researchers performed multiple field studies on the use of email. In 1996, Whittaker and Sidner analyzed the mailboxes of 18 NotesMail users containing 2,482 emails. They postulated email overload and studied how users handle a mass of emails [159]. In 2006, Fisher et al. revisited this analysis with a sample of 600 mailboxes containing 28,660 emails [57]. They analyzed users' email sorting strategies, especially with respect to dealing with the increased volume of emails, and postulated that large folders would make email retrieval hard. Alrashed et al. studied a sample of anonymized email logs from Outlook Web Access over a four month period, containing about 800 million actions[14]. They aimed to understand how users handle incoming emails. They find that most emails have a short lifetime and that deleting email is the most common action on messages users interacted with once. Avigdor-Elgrabli et al. analyzed a sample of donated mailboxes of a major email service provider, containing about 5 million emails [19]. They used machine learning techniques to identify relationships between emails. Roth et al. performed a study with anonymized mailboxes from 17 voluntary users that contained approximately 139,000 mails to investigate which security mechanisms would be most appropriate for their communication patterns [111]. They argue that for individual non-commercial users, out-of-band verification of keys would be more feasible than relying on public key certificates issued by third parties.

In addition to the above field studies that focused on email usability more broadly, researchers investigated the adoption of security protocols for email. Foster et al. scanned the Simple Mail Transfer Protocol (SMTP) configurations of about 300,000 major email providers and email generators in March 2014 and February 2015, and investigated the behavior of known email providers [60]. They found that TLS is widely used and discovered a dramatically low adoption of effective TLS certificate validation. Durumeric et al. examined log data for SMTP handshakes of Google's Gmail service from January 2014 to April 2015 and compared it with a snapshot of SMTP configs of Alexa top million domains as of April 2015 [39]. The authors examined the distribution and use of TLS and other server-side security mechanisms. They found that the top mail providers proactively encrypted and authenticated messages. However, these practices had yet to reach widespread adoption in a long tail of over 700,000 SMTP servers with less secure configurations. Ulrich et al. evaluated the PGP key database (SKS-Keyserver) in December 2009, examining 2.7 million keys of which 400,000 were expired and 100,000 were revoked [147].

Overall, the above field studies investigate the adoption and usability of email in a broader context, measure email servers' security configurations, and conduct small-scale security analyses of email encryption. We extend previous field studies by focusing on adopting email encryption using a longitudinal large field dataset.

3.4 Analyzing 27 Years of Email Data

Below, we provide detailed information on the data collection and analysis process in our work.

We performed our longitudinal analysis of a large email dataset in coordination with the technical staff of the university's IT department, the data protection officer, and the university's staff council (see Section 3.4.3 for more details). We implemented a data collection pipeline to collect pseudonymized[8] metadata for all email accounts, including the use of S/MIME and PGP (cf. Figure 3.1) at our institution in the last 27 years. At no point did we collect email subject or body information to avoid the disclosure of personally identifiable information (PII) to the researchers. We also ensured that metadata included neither email account names nor the departments' names or subdomains. We aimed to keep the number of processing errors low and consistently tested the pipeline with our own mailboxes until no further processing errors occurred.

The IT department's technical staff reviewed the pipeline for functionality and data protection aspects, and then executed it on the university's standby backup email server. The backup server is a hot-standby of the primary mail server, and automatically takes over in case of a failure. Data between both servers is constantly synchronized and as a result, the backup is identical to the live data. The backup server retains all email data, dating back to early 1994.

Figure 3.1 provides an overview of the nine-step processing pipeline: We initially started with a local testing environment on a small sample emailbox created specifically for our study **(1)**; The technical staff reviewed the initial pipeline and iteratively tested it against the full set of emailboxes of the researchers and their own emailboxes **(2)**; We exported the emailboxes to JSON-formatted files **(3)**; We parsed and pseudonymized all emails **(4)**; We performed assertion checks on every email to ensure that neither the emailaddress nor the domain was present in any result fields

[8]The pseudonymization process is described in Sections 3.4.1 and 3.4.3.

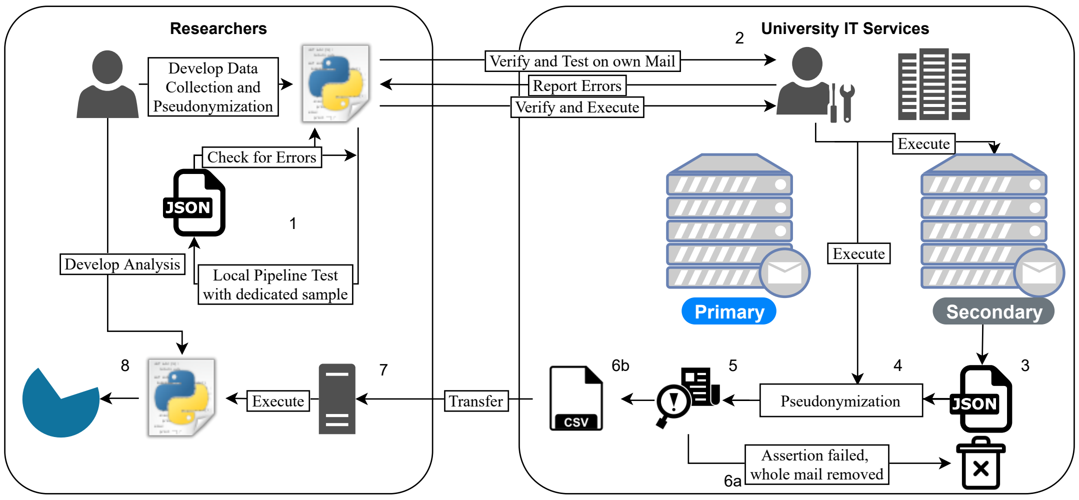


FIGURE 3.1: Illustration of our data collection and evaluation pipeline.

to account for emailclients writing private data to unexpected places **(5)**; In the case of a succeeding assertion check, we stored the resulting email metadata for further analysis on a secure server in the university's computing center **(6a)**; In the event of a failure, we dropped all email metadata to avoid the leakage of private information **(6b)**; The IT department's technical staff transferred pseudonymized results to the authors' secure cloud storage **(7)**; We analyzed the pseudonymized results **(8)**.

3.4.1 Data Collection Pipeline

The university uses Dovecot[9] for their email servers. Dovecot offers an export feature to extract all emails as a JSON-formatted file. We implemented our processing pipeline using a large JSON file per mailbox containing all exported emails as input. For parallel processing, we used the GNU Parallel [142] tool. On behalf of the researchers, the university's IT staff executed the pipeline on the backup email server to make sure raw emails were not exposed to the researchers. Below, we describe the metadata we collected. In the cases where we applied pseudonymization, we provide a description of the pseudonymization procedure. Table B.1 in Appendix B.2 gives an overview of both general and S/MIME and PGP specific information.

General Information For each email we collected the local user account, message ID, the sender, and the list of receiver email addresses. If present, we also collected the lists of carbon copy and blind carbon copy addresses for outgoing emails. For pseudonymization, we hashed all of these values using a secret salt[10] and the SHA-256 hash function. We grouped email users into: Student, Staff, Faculty, NX Unknown[11] and External. For data protection reasons, we did not collect the exact send

[9]`https://www.dovecot.org/`
[10]The secret salt was only accessible to the IT staff and not to the researchers
[11]Used if the email subdomain did not exist anymore, and the original purpose was unclear when we performed our experiments.

and receive dates as well as times of an email but only the corresponding week. If set, we collected the raw user agent string to identify email client software, operating system, and compute platform if possible. We grouped mailbox folders into: Inbox, Subfolder of Inbox, Outbox, Subfolder of Outbox, Junk, Trash, and Spam. For further cryptographic metadata analysis, we stored whether an email was signed and encrypted or contained Autocrypt headers. Below, we use the term *cryptographic emails* for all emails that contained cryptographic metadata.

Cryptographic Metadata For all cryptographic emails, we collected attached cryptographic metadata for S/MIME, PGP and Autocrypt. Table B.1 gives an overview of the collected information and storage formats including pseudonymization.

For S/MIME, we collected the pseudonymized serial number of the leaf certificate, validity start and end date (granularity by week), the signing hash algorithm, the key size and key type (e.g., 2048 and RSA). We collected the key usage and extended key usage options (e.g., email signing, certificate signing, code signing), and the number of valid email addresses for each certificate. Finally, we collected the complete certificate chain, including all metadata for all signing certificates. We did not pseudonymize the serial numbers for non-leaf certificates.

For PGP, we collected the key type (e.g., public key or a sub-key), the signature algorithm, and key length. For elliptic curve keys, we collected curve information as well as pseudonymized key IDs, and creation and expiration dates. For extended PGP keys, we collected update dates. If a key included subkeys we also stored the above data for the sub keys.

The pipeline dropped all data (like email subject, email content, non-key attachments) that we excluded from our analysis.

3.4.2 Data Cleaning

We included all 81,647,559 emails and 37,463 email user accounts at our university from January 1994 to July 2021 in the initial analysis (cf. Section 3.5.1). However, we excluded some emails and email user accounts based on the following procedure:

Processing Errors Parsing a dataset that spans millions of emails from over two decades that were generated and sent by many different email clients and versions poses a significant challenge. Hence, we were not able to successfully parse all emails. Our parser failed to parse 0.09% emails in the dataset. Due to privacy restrictions, we were not allowed to further investigate root causes of parser failures. However, Appendix B.1 provides more details on S/MIME and PGP related parsing errors.

Inactive Email Accounts Our initial data set included 37,463 total user accounts. However, we identified 18,302 inactive email accounts for which we did not find any sent emails. Of them, 17,928 email accounts received but did not send emails and 374 did neither receive nor send emails. Many students prefer to use their private email accounts instead of their automatically created university email accounts, leaving them inactive.

Invalid Dates We excluded 307,680 (0.38% of our dataset) emails with obviously forged header dates, e.g., year <1994 or > 2021 (after data collection) and emails for which the date parser failed.

3.4.3 Ethical Concerns and Data Privacy

To conduct the large scale measurement study on email data, our institutions, and specifically the institution where the data was collected and evaluated, did not require formal ethical review for this type of study. Therefore, we did not involve an ethics review committee. However, we followed our institutions' guidelines for *good scientific practice*, which includes ethical guidelines. Here, the institutions specifically place the burden of determination of whether research is ethical on the respective researchers. We intensively discussed within and outside our research team to determine possible concerns with this research project, and whether this project would be feasible. We concluded that in addition to following laws and our institutions' ethics requirements, we should also follow the de facto ethics standards of the S&P community. The data used in this study can be described as pseudonymized data derived from human subjects, as mentioned in the Call for Papers.

In addition to ethics, we made sure to address all legal aspects of our research to adhere to strict German privacy protection laws and the European General Data Protection Regulation (GDPR). Therefore, we involved the data protection officer and the works committee of the institution where the data was collected and evaluated, as required by the German data protection regulations. We developed the data collection plan jointly with the data protection officer, with the goal to protect users' privacy and adhere to the strict German data protection regulations and the regulations in the state of Lower Saxony. After more than a year of multiple discussions and hearings, we agreed on the presented data collection plan (cf. details in Section 3.4.1). After the involved authorities had rigorously assessed the legal situation based on the GDPR, German data protection laws, and state law of the involved authorities, we were allowed to analyze pseudonymized metadata of all users at our institution without requiring user consent. Additionally, our legal counsel decreed that the benefit of our research to society outweighed the risks to individuals. We concur with the assessment that answering our research questions is beneficial for future end-to-end encryption research, which ultimately benefits society, and that there was no harm done to any participants based on possibly re-identifiable metadata. Importantly, we will not publish the metadata we collected and only an encrypted copy will be stored at the university data center for ten years without access by researchers to follow good scientific practice. As is common in research, we only publish aggregate data, and no email accounts can be re-identified through the publication of this paper. As part of the joint development of our data collection plan, we decided to take the following measures to protect users' (metadata) privacy and adhere to the GDPR, German, and state laws:

- The involved researchers never had access to raw data. The data collection pipeline was executed by the university's IT staff who operate the email servers and have access to the backup data. They transferred the pseudonymized data to the researchers to a secure server.
- We reduced the amount of data to the absolute minimum we required to investigate our research questions.
- We used cryptographically secure hash functions with salts unavailable to the researchers to pseudonymize user data.
- At all times pseudonymized data (cf. Table B.1 in the appendix) was only stored on secured university servers.
- We did not and will not share any data other than the aggregate numbers in the paper with anyone outside the team of involved researchers.
- We assured the data protection officer and the works committee that we would not take any actions to de-identify users.

The above data protection precautions ensured the necessary compromise between user privacy and data utility in order to perform the analyses on which we report in Section 3.5.

3.4.4 Limitations

Like every measurement study, our work comes with several limitations:

Data Set Our dataset includes email data from January 1994 to July 2021 covering 81,647,559 emails of 37,463 users from a large Germany University with more than 30,000 students and 5,000 employees. However, the dataset might not be complete. We could not include emails that users deleted from their accounts; similarly, we did not include emails sent to deleted accounts. Some research groups and departments have their own email servers; their emails were also not included in our dataset. Therefore, the dataset should not be assumed to include all emails that were sent or received from January 1994 to July 2021 at our university. Additionally, we do not assume that our dataset is representative for all email data in Germany or globally. Instead, we think our data set might overreport the use of email encryption since most email users in the dataset are highly educated and our university offers free S/MIME certificates to all email users.

However, we think our dataset is one of the largest and most valuable for the type of analysis we perform and our results are a valuable contribution to the security research community.

Data Analysis We could not analyse all cryptographic details, e.g., we could not verify digital signatures since we were not allowed to parse the body content of emails and we could not extract details for certificates or signatures that were used in encrypted emails, since this data is also encrypted.

We tested our pipeline thoroughly but still missed some edge cases that inevitably arise in mail software that evolves over a long time span. While our pipeline was able to process 99.91% of all emails, processing failed for the remaining 0.09%. Errors during S/MIME and PGP parsing were logged separately. We encountered 1,199 S/MIME and 23,168 PGP emails where parsing failed (cf. Appendix B.1 for more details). We deemed this margin of error tolerable compared to the high organizational costs of refining and repeating the entire process once more.

Distinction Between Send and Receive The dataset we evaluated did not contain information whether an email was sent or received. To still group emails into sent and received emails, one approach would be to group emails based on the folder they were stored in. However, this would introduce challenges such as email clients using different names for sent folders or users using their own folder names. Therefore, we decided to identify emails based on multiple parameters. Sent emails were not allowed to contain a `return_path` header, since it is added by outgoing mail servers and emails were only allowed to go through at most one mail relay[12]. Most incoming emails have more than five email relays.

Mail Client Behaviour The macOS and iOS clients AppleMail, iPhoneMail and iPadMail generally identify themselves using the *X-Mailer* header in mails, but the copy placed in the sent folder by these clients does not contain this header. As a result, these clients could only be correctly detected on received emails, their sent mails are included in the "No User Agent" group. The ticket system used by the university data center automatically deletes mails in its inbox and does not place a copy in the sent folder and as a result only the answered tickets are available in our dataset.

3.4.5 Replication Package

To improve the replicability of our work, we provide a replication package including the following material: (a) the complete processing pipeline consisting of multiple Python scripts to process and pseudonymize emails from Dovecot mail servers, (b) the analysis scripts to replicate our results on different datasets, and (c) the agreement with our data protection officer. Due to the sensitive nature of our measurement study, we cannot make raw email data available. We hope this replication package helps future studies to better compare and position themselves to our work;

[12]One mail relay header is added when the webmail client of the university is used for sending mails. Regular mail clients do not add this header.

and hope others replicate our work on different email datasets to improve our community's understanding of the use of email encryption. The replication package is available on our website at [138].

3.5 Results

We provide a detailed analysis of the email corpus we collected and the adoption of S/MIME and PGP from 1994 to July 2021 below.

3.5.1 Dataset Summary

In total, we analyzed metadata for 81,647,559 emails from 37,089 email accounts. Overall, the university's email users exchanged 40,540,140 (49.67%) emails internally.

Figure 3.2 illustrates the use of email at our university in the past 27 years. While we found only 350 emails in 1994, we detected an almost exponential growth and found 17,190,472 emails in 2020. This development reflects the enormous relevance of email as a communication tool and is in line with reports on the global use of email[13].

Use of Email Encryption and Signatures We found 2,334,042 (2.86%) emails that were either encrypted or signed using S/MIME or PGP. We identified a huge difference between the use of email encryption and signatures.

46,973 (0.06%) emails were encrypted. 26,105 (55.57%) emails were encrypted using S/MIME and 20,868 (44.43%) of them were encrypted using PGP. In contrast, 2,287,922 (2.8%) emails were signed. 2,040,794 (89.2%) of them were signed using S/MIME and only 247,128 (10.8%) were signed using PGP.

Figure 3.2 illustrates the use of S/MIME and PGP between 1994 and 2020. We found the first S/MIME signed email in 1998 and the first S/MIME encrypted email in 1999. The first PGP signed email appeared in 1994 and the first PGP encrypted email in 1997.

Key Insights: Dataset.

- We saw an exponential growth of the use of email between 1994 and 2020.
- Only 0.06% of emails were encrypted.
- 2.8% of emails were signed.
- S/MIME was more widely used than PGP.

[13] cf. `https://www.emailisnotdead.com/`

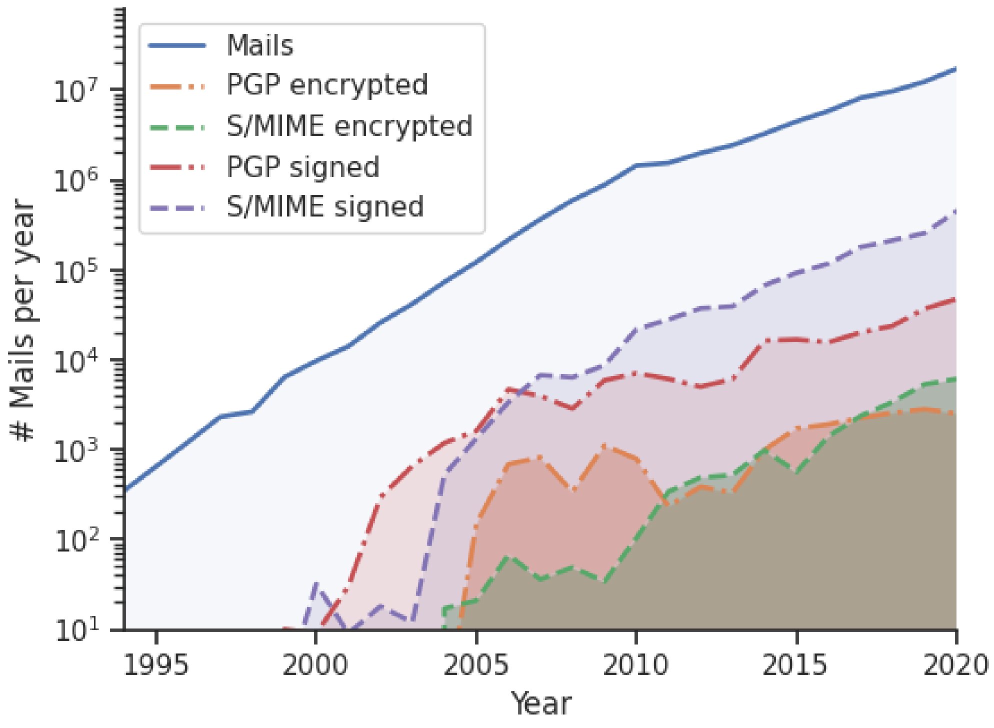


FIGURE 3.2: Rise of email, S/MIME and PGP over time at our university.

3.5.2 S/MIME Certificates and PGP Keys

Below, we give an overview of the S/MIME certificates and PGP keys we found. This includes certificates and keys from internal and external senders. Overall, we were able to collect 9,765 S/MIME certificates, 3,741 primary PGP keys and 3,840 sub keys (cf. Table 3.2 for details).

S/MIME Certificates All but one certificate that used an elliptic curve encryption algorithm supported the RSA encryption algorithm.

2048 bits was the most widely used RSA key size (91.58%); 5.54% of the RSA keys had 4096 bits. In 237 cases we saw 1024 bits RSA keys; 6 RSA keys had 512 bits. While the last 512 bits RSA key we found was created in 2010, we saw 2 1024 bits RSA keys created in 2020.

7,472 (76.52%) certificates supported the SHA-256 signature algorithm. However, we also found outdated signature algorithms including SHA-1 (2,028; 20.77%) or MD5 (148; 1.52%). Surprisingly, we found 11 certificates issued in 2020 using SHA-1. The last certificate using MD5 was generated in 2017.

5,194 (53.19%) of all certificates expired in 2020 or earlier. The mean validity period for S/MIME certificates was 3.13 years (sd= 2.70) ranging overall from a minimum of 4.00 weeks to a maximum of 99.99 years. 6,953 (71.20%) certificates

	S/MIME				**PGP**				**Total**	
	Sent		**Received**		**Sent**		**Received**		**Sent**	**Received**
	Signed	**Encrypted**	**Signed**	**Encrypted**	**Signed**	**Encrypted**	**Signed**	**Encrypted**		
Total	356,330	9,358	1,684,464	16,747	69,950	8,197	176,983	12,633	16,660,280	64,952,315
Client										
Thunderbird	231,475	6,311	508,423	7,103	46,407	5,775	85,904	6,825	8,206,215	11,875,969
Outlook	78,736	1,675	258,013	3,728	325	22	5,528	52	3,102,760	7,785,073
Ticketsystem	0	0	311,743	1	0	0	223	0	4	1,259,208
AppleMail	?[1]	?[1]	58,752	2,157	?[1]	?[1]	20,722	1,408	?[1]	2,268,757
Evolution	11,479	190	16,366	341	232	48	1,387	74	26,224	70,482
Mutt/NeoMutt	0	29	50	31	564	1	11,506	1	6,066	141,263
Outlook-Express	7	0	7,355	31	0	0	2,452	16	29,065	447,775
Claws Mail	0	0	145	0	1,654	207	2,918	161	2,573	21,640
iPhone/iPad-Mail	?[1]	?[1]	5,275	104	?[1]	?[1]	7	8	?[1]	723,482
MailMate	2,361	8	3,526	3	63	2	131	3	5,296	8,364
Other	1,270	3	13,397	128	2,804	221	8,658	2,167	2,572,631	9,715,649
No Useragent	31,002	1,142	501,419	3,120	17,901	1,921	37,547	1,918	2,708,758	30,634,653
Operating System[2]										
Windows	167,350	2,183	326,747	2,555	38,751	2,531	66,786	2,589	7,150,676	10,679,520
Linux	60,827	3,985	164,926	4,379	8,797	3,372	23,069	4,191	799,179	1,457,578
Mac	5,909	170	88,144	2,397	2,126	172	24,849	1,841	340,599	3,057,579
iOS	1	0	5,294	104	0	0	7	8	695	726,209
Android	81	1	66	0	34	101	167	110	84,744	120,115
Webmail	135	3	831	8	10	2	322	2	2,378,940	2,425,629
Unknown	91,025	1,874	597,037	4,184	2,331	98	24,236	1,974	3,196,689	15,851,032
No Useragent	31,002	1,142	501,419	3,120	17,901	1,921	37,547	1,918	2,708,758	30,634,653
Usergroup										
Scientific	237,579	3,992	808,539	8,376	66,483	6,708	115,388	7,796	11,776,198	39,504,464
Staff	106,434	5,318	684,791	8,155	2,122	1,134	49,111	3,864	3,265,602	17,560,401
NX Internal	9,903	20	121,243	129	77	39	7,153	436	970,160	4,960,051
Student	1,700	28	67,813	71	1,242	311	4,903	536	393,685	2,703,240
External	714	0	2,078	16	26	5	428	1	72,256	224,159
Affected emails	>= 5%	> 4%	> 3.5%	> 3%	> 2.5%	> 2%	> 1.5%	> 1%	> 0.5%	<= 0.5%

[1] AppleMail and the iOS mail client miss the *X-Mailer* header in the sent folder. Hence, we identify their client as "No Useragent".
[2] Not all clients store the operating system as part of the Useragent/X-Mailer field. If not available, we identify them as "Unknown".

TABLE 3.1: Top 10 clients of S/MIME and PGP users at our university.

were created between 2015 and 2020 with a peak of 1,654 certificates (16.94% of all S/MIME certificates) in 2019.

Overall, we found 671 different issuer names. However, 1,150 (11.78%) certificates had no issuer. The most prominent issuer was the DFN issuing 3,209 (32.86%) certificates. 622 (6.37%) were issued by our university. Another German university issued 563 (5.77%) of all S/MIME certificates we found. In 332 (3.40%) cases, a distinct issuer only signed a single certificate. 89 (0.91%) issuers signed only two certificates. 137 of them (32.54%) had no root CA. In total, we identified 495 root CAs. 274 of them were only the root of one certificate. The Deutsche Telekom was the root CA of 6,342 (64.95%) certificates. In contrast, 1,150 (11.78%) were self-signed and not linked to a CA.

PGP Keys PGP keys can be split into primary and sub keys. Each primary key can have multiple sub keys. Below, we focus on primary keys. We found 3,741 primary and 3,840 sub keys. Primary keys had on average 1.03 sub keys with a maximum of 10.

Most PGP keys used RSA (3,169; 84.71%) and 2048 bits (928; 24.81% keys) or 4096 bits (2,015; 53.86% keys) keys, followed by DSA (323; 8.63%) with mostly short key

			# S/MIME	# PGP Primary	# PGP Sub
			9,765	3,741	3,840
Cryptographic Algorithms[1]	RSA	512	6	-	-
		1024	237	6	6
		2048	8,942	928	950
		3072	32	208	226
		4096	541	2,015	2,078
		8192	5	5	2
	DSA	768	-	1	-
		1024	-	290	1
		2048	-	17	2
		3072	-	15	2
	ElGamal	768	-	-	1
		1024	-	-	61
		2048	-	-	186
		3072	-	-	13
		4096	-	-	53
	EC	521	1	-	-
	ECDH	256	-	-	247
	EdDSA	256	-	249	2
Validity Period	mean	(years)	3.13	4.67	4.62
	std. dev.	(years)	2.70	3.39	3.68
	median	(years)	2.99	4.99	4.99
	min	**(weeks)**	4.00	0.00	0.00
	max	(years)	99.99	85.82	85.82
	unlimited	-	-	1,234	1,240

[1] Key sizes in bits

TABLE 3.2: Details for S/MIME certificates and PGP keys in our dataset

sizes <= 1024 bit (290; 7.75%) for signatures and Elgamal sub keys for encryption.

PGP key expiration dates can be set and extended by the user themselves. We saw this in 219 (5.85%) cases with a maximum of 4 extensions. 1,234 (32.99%) keys had no expiration date, while 1,344 (35.93%) keys were expired. The average validity period for PGP keys was 4.67 years (sd 3.39) ranging from zero weeks to 85.82 years.

The majority (276; 85.45%) of DSA keys were created before or in 2010 and used 1024 bit keys. 278 (86.07%) had no expiration date set. The last two DSA keys were created in 2019 with 3072 bit keys.

928 (24.81%) of the RSA keys used 2048, 2,015 (53.86%) 4096 bits. Only 6 (0.16%) had 1024 bit keys. 2048 bit was the most widely used key size (60-70% per year) between 2010 and 2013. 4096 bit was most dominantly used from 2014 and peaking in 2017 with 86.18%. Interestingly, in 2018 the picture changes again and 2048 and 3072 bit keys appeared more frequently again with 2048 bits taking back the lead in

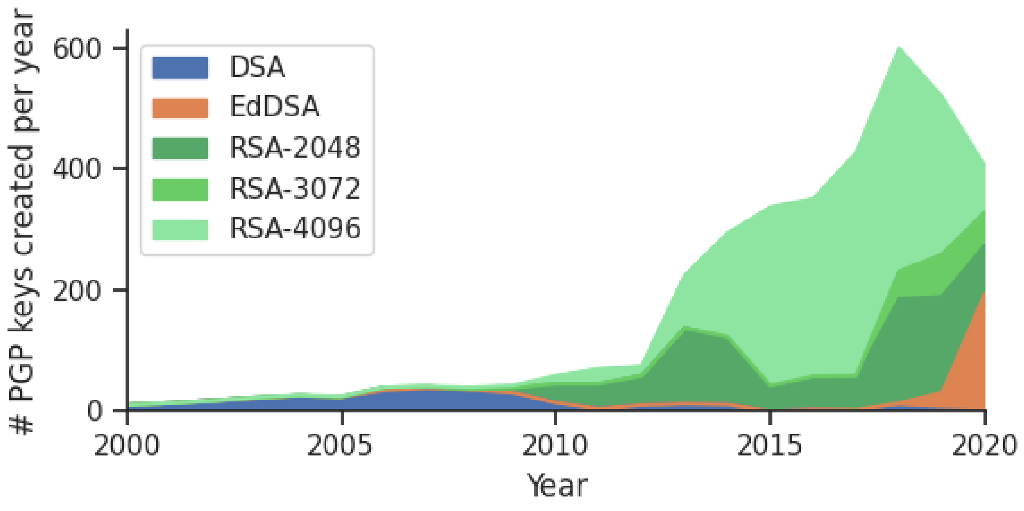


FIGURE 3.3: Distribution of primary PGP keys over time.

2020, although the amount of RSA keys is starting to decline in favor of EdDSA. We attribute this phenomenon to the release of Autocrypt in 2018 which uses RSA 2048 bit keys by default to avoid the 10kB header limit. [18].

We found the first four EdDSA keys in 2017, followed by nine in 2018, 31 in 2019 and 193 in 2020, almost reaching the number of RSA keys (207). 53 (21.29%) of the EdDSA keys did not have an expiration date set.

Until 2000, all 33 PGP mails were hashed with MD5. Between 2004 and 2015, SHA1 was most widely used in 62,273 (80.96%) PGP mails.

We found the first use of SHA2-256 in 2006 with a steady increase until 2016 when it became the most widely used algorithm with a total of 125,973 mails (68.54% since 2016; 50.22% overall). SHA2-512 first appeared in 2008, has constantly been growing since, and has been the second most used hash algorithm since 2016 (44,859; 25.94% since 2016; 20.26% overall). The remaining hash algorithms SHA-224 (7; <0.01%) and RIPE-MD160 (132; 0.05%) were almost non-existent in our dataset.

Key Insights: S/MIME and PGP.

- RSA is the most widely used key algorithm.
- A key size of 2048 bits was used most often with S/MIME.
- PGP keys used 4096 bits most often, although newer PGP keys used less secure 2048 bits.
- About one third of the PGP keys did not have an expiration date set.
- The *Deutsche Telekom* was root CA for 64.95% of all S/MIME certificates.

3.5.3 User Interaction with S/MIME and PGP

Overall, we identified 37,463 unique user accounts including 374 accounts with no emails at all and 18,302 users who never sent an email. Below, we focus our analysis on the remaining 19,161 active email users who have sent at least one email. Active users sent 688.93 emails (ranging from 1 to 182,379 emails) and received 3,249.81 (ranging from 0 to 2,881,904 emails) on average.

Based on the subdomain of user accounts' postboxes we assigned user accounts to the groups *scientific*, *staff*, *students*, *external* and *NX internal*. We assigned administrative staff of a research group to *scientific*, assigned subdomains that were no longer available when we collected and analyzed the data to *NX internal* and *external* to accounts that are hosted by the university but belong to external research projects. 9,053 (24.41%) of our users were in the scientific user group, 2,169 (5.85%) in staff and 19,002 (51.23%) were students (cf. Table 3.1 for a distribution of emails for all user groups).

Below, we focus on user behavior. We aim to understand real world implications of the common usability challenges (cf. [17, 63, 64, 84, 160]) with a focus on:

- General use of S/MIME and PGP.
- Use of S/MIME and PGP with multiple clients.
- Distributing S/MIME certificates and PGP keys.
- Long-term S/MIME certificate and PGP key management.
- Leakage of private keys.

General Use of S/MIME and PGP 94.54% of the active users never used S/MIME or PGP for sending email. In contrast, 62.59% of our active users received at least one email signed or encrypted using PGP or S/MIME.

375 users (1.96% of the active users) sent at least one encrypted email. While 167 users used PGP exclusively and 159 used S/MIME exclusively, 49 used both. 1,047 users (5.46% of all active users) signed at least one email. 446 users PGP exclusively, 455 users used S/MIME exclusively, and 146 users used both.

S/MIME users signed 33.58% of their emails on average after using S/MIME for the first time but encrypted only 1.26% of their emails on average. PGP users signed 4.90% of their emails on average but encrypted only 0.96% of their emails.

Breaking down S/MIME and PGP use into user groups, we observed that staff users used S/MIME to sign their emails more often than scientific users (3.26% vs 2.02%) and eight times more often than students (0.43%) on average. In contrast, the amount of PGP signed emails is smaller across all user groups, with the scientific user group being at the top with 0.56%, the staff users with 0.06%, and students with 0.32% on average.

For more details including the use of encryption, please refer to Table 3.1.

Use of Multiple Email Clients To identify email clients, we relied on user agent metadata in emails. We were able to detect email clients for 48,269,184 (59.14%) emails. Hence, we could detect the email client for 13,951,522 (83.74%) emails sent by our users, for 14,492 (82.55%) encrypted and 377,377 (88.53%) signed emails. However, 33,343,411 (40.86%) emails did not contain user agent information.

The most common client used for signing and encryption was Thunderbird in combination with Enigmail for PGP support[14]. It was used to send 289,968 (65.35%) S/MIME and PGP emails in our dataset. 80,758 (18.20%) of the S/MIME and PGP emails were sent using Outlook. Table 3.1 provides more details on the top 10 email clients.

Using multiple email clients posed additional burden on the widespread use of S/MIME and PGP since users have to synchronize their keys and certificates across all clients. For 8,828 (46.07%) out of 19,161 users, we could detect multi-client use. Table 3.3 provides an overview of multi-client use across all users and S/MIME and PGP users. Figures 3.5 and 3.6 compare how S/MIME and PGP users employ signatures and encryption between single- and multi-client users. Our data implies that single-client users sign emails more frequently than multi-client users. For S/MIME users, the number of signed emails decreases from median 62.25% for single-client users to median 1.58% for dual-client and to 1.03% for triple-client users. However, for power users with more than three clients the number of signed emails increases to median 17.48%. The majority (74.90%) of multi-client users tend to use one of their clients for email signatures exclusively and not employ signatures with other clients, while only 25.10% of the multi-client users used email signatures across different clients. In contrast, single-client PGP users only signed 0.36% of their emails on average. However, we did not find considerable differences between single and multi-client PGP users (cf. Table 3.3 for more details).

S/MIME Rendezvous In addition to multiple clients, the distribution of S/MIME certificates and PGP keys between users poses another significant challenge for the adoption of email encryption. We focus on S/MIME certificate distribution, since S/MIME clients automatically attach certificates including public keys to signed emails. However, plain PGP clients[15] do not automatically attach their public keys to signed emails but require users to manually attach public keys or look them up on PGP key servers.

Signed S/MIME emails were the most frequent cryptographic emails in our dataset. Therefore, we were able to investigate users' interactions with S/MIME certificates they received from their communication partners. Without additional effort for key exchange, S/MIME users who received S/MIME certificates from others could encrypt future emails to those senders. Below, we report on such behavior of users in our dataset. 601 users (3.14% of all internal active users) sent at least one signed email. Overall, we identified 374 rendezvous where both sender and recipient exchanged public S/MIME keys due to sending signed emails to each other.

[14]Native PGP support was added to Thunderbird 78 (June 2020)

[15]These clients do not have Autocrypt support

Most emails (64.08%) between S/MIME rendezvous partners were signed. However, only 3.36% of all emails between rendezvous partners were encrypted on average. Once one rendezvous partner had sent a first encrypted email, 13.95% of all following emails were encrypted on average. Figure 3.4 illustrates the distribution of signed and encrypted emails between S/MIME rendezvous partners.

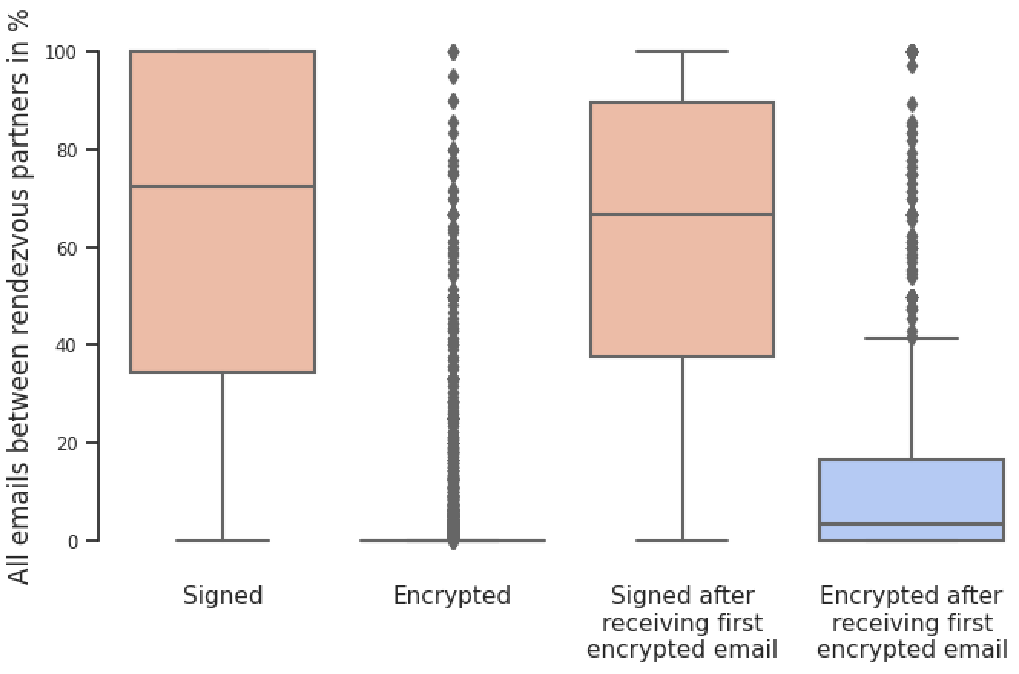


FIGURE 3.4: Distribution of signed and encrypted emails between S/MIME rendezvous partners. While they signed most of their emails, they encrypted only few. However, receiving an encrypted email had a positive impact on encrypting future emails between rendezvous partners.

Long-term S/MIME and PGP Key Management Below, we report on the long term use of S/MIME certificates in our dataset. In particular, we focus on the replacement of outdated certificates. Overall, we identified 680 university email addresses with at least one S/MIME certificate which was actively used. 496 (72.94%) of them were valid until December 2020. 203 (29.85%) of these email addresses had two or more certificates associated. On average, they used 2.86 certificates – one used up to 27 certificates.

Since certificates expire and need to be replaced with new ones, users need to create, set up, and distribute them ideally very close to the expiration date of the old certificate. Overall, we found 364 certificate rollovers. In 271 cases, the new certificates had a longer expiration period than the old certificates. 229 of the certificate rollover events we detected occurred in time before the old certificates expired. On average, they happened 55.05 weeks before the old ones expired. Figure 3.7 shows that the majority of users often create a new certificate shortly before the previous

	S/MIME								PGP							
	Emails signed[1]				Emails encrypted[1]				Emails signed[1]				Emails encrypted[1]			
clients[2]	**1**	**2**	**3**	**4+**	**1**	**2**	**3**	**4+**	**1**	**2**	**3**	**4+**	**1**	**2**	**3**	**4+**
mean	53.55	23.50	28.17	33.22	1.39	0.33	2.57	1.02	14.08	6.01	2.08	4.39	2.53	0.97	0.72	0.75
std. dev.	44.99	35.79	36.09	35.46	7.17	1.88	11.38	2.66	27.05	18.14	8.84	13.90	12.91	2.91	2.65	1.81
25%	1.59	0.14	0.13	0.10	0.00	0.00	0.00	0.00	0.08	0.04	0.03	0.03	0.00	0.00	0.00	0.00
50%	62.25	1.58	1.03	17.48	0.00	0.00	0.00	0.06	0.36	0.13	0.10	0.09	0.00	0.00	0.00	0.00
75%	100.00	45.03	57.92	67.91	0.00	0.00	0.24	0.79	10.00	0.57	0.32	0.42	0.00	0.00	0.00	0.48
max	100.00	100.00	99.48	98.61	67.59	20.00	85.85	20.17	100.00	93.75	72.61	91.23	100.00	20.53	22.58	13.52

[1] Mean, std. dev, percentiles and max values in %
[2] Number of clients used by single users.

TABLE 3.3: Distribution of signed and encrypted emails for multi-client users.

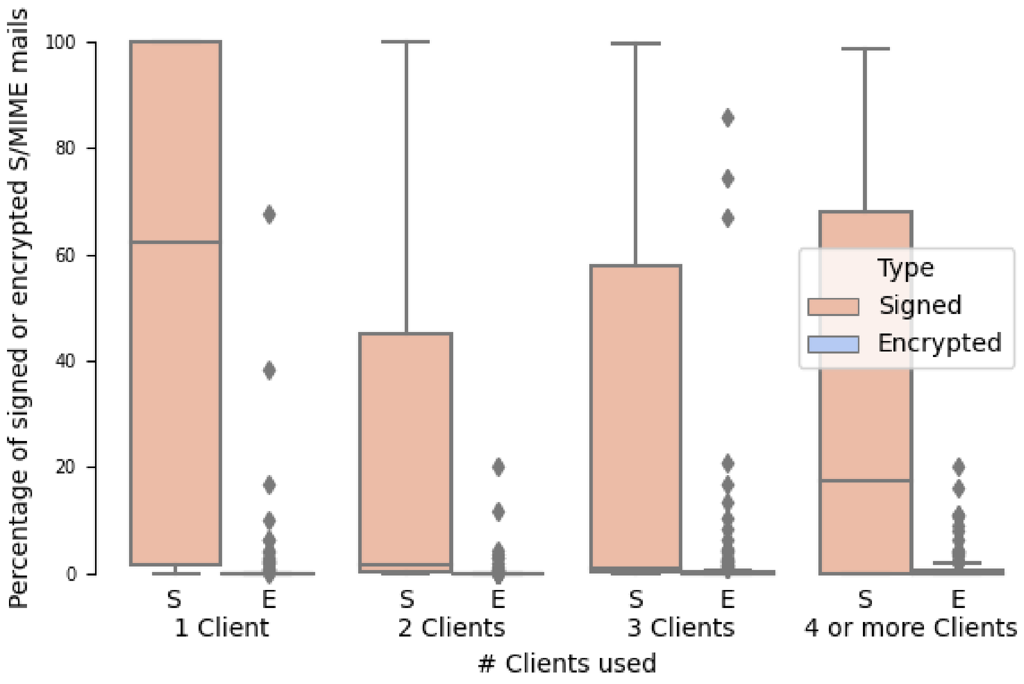


FIGURE 3.5: Distribution of S/MIME signatures and encryption for users with one or multiple clients.

certificate expires. However, we also detected 42 certificate rollover events that occurred after the old certificates expired. S/MIME could neither send signed nor receive encrypted emails in this time period. On average, the late rollover events occurred 70.64 weeks after the expiration dates.

Leakage of Private Keys Overall, we encountered three instances of private PGP keys (and their private sub keys) being sent via email in 2015, 2017, and 2018. All three keys were sent by the users to themselves. One of those was a freshly created PGP key (i.e., less than one week old). This is not necessarily a security issue as long as the passphrase to protect the key is adequate.

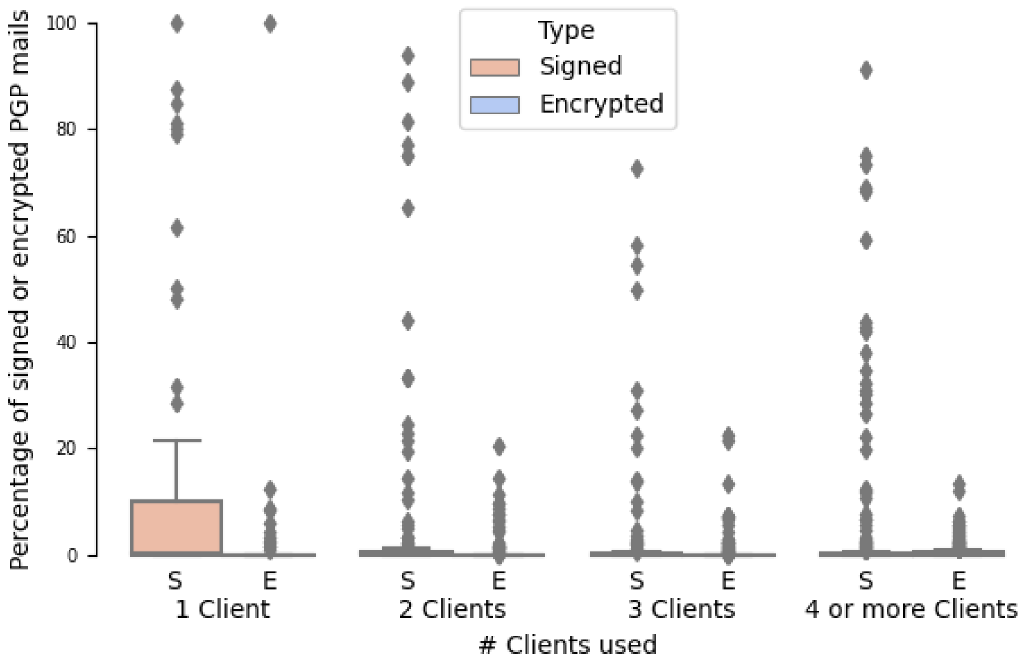


FIGURE 3.6: Distribution of PGP signatures and encryption for users with one or multiple clients.

Key Insights: S/MIME and PGP users.

- More than 94% of all active users never used S/MIME or PGP.
- S/MIME users signed six times more of their emails than PGP users on average.
- Using two to three different clients decreased the likelihood of signing emails by 51.76%.
- On average, less than 3% of all emails between users who had exchanged S/MIME certificates previously were encrypted.
- Leakage of private keys via email does not seem to pose an issue.

3.6 Discussion

In this section, we discuss the implications of our results and recommendations for the future development of email encryption in five key areas: Limited use of email encryption, use of insecure keys, impact of certain events, challenges of using multiple clients, and lack of opportunistic encryption.

Limited Use of Email Encryption As illustrated in Section 3.5.1, we observed that only a very small fraction of emails in our dataset was encrypted (0.06%) or signed (2.8%). While we saw an exponential growth of the use of email overall (cf. Figure 3.2), the fraction of encrypted emails remained consistently small. Our results also imply that S/MIME was more widely used than PGP for both signing and encrypting emails. Although these findings do not come unexpected in the light of

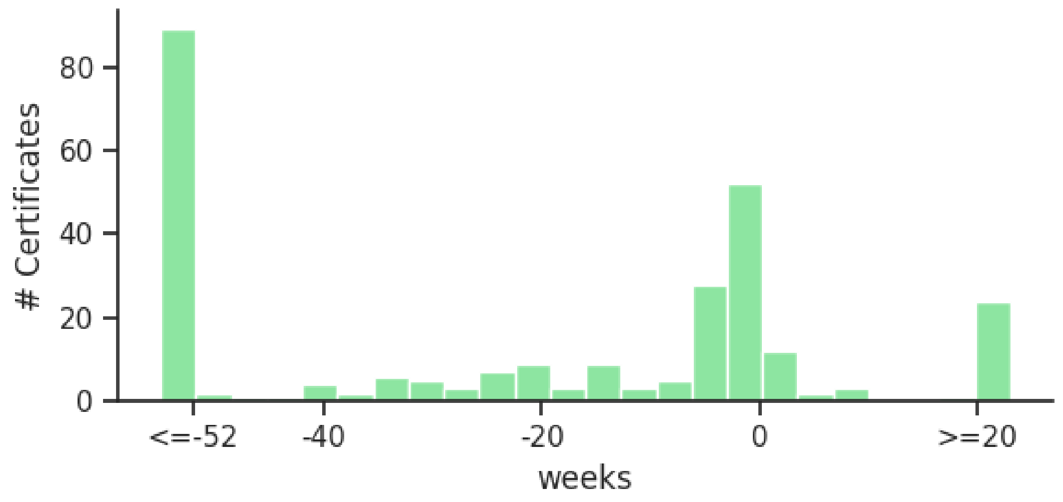


FIGURE 3.7: Distribution of the time around certificate expiration for certificate renewals in weeks. Most certificates were renewed around one year before they expired due to an expired root CA certificate. The second most certificates were renewed one week before they expired.

previous work [65, 83, 115, 116, 119], we think they can serve as ground truth to confirm previous user study results and have value for future development. The small fraction of email encryption and the fact that the use of email encryption did not grow with the overall use of emails suggest that both S/MIME and PGP are niche tools that are mostly used by a small number of security-aware users. However, the difference between the use of S/MIME and PGP seems to give grounds for optimism. Our institution provides an S/MIME infrastructure and encourages its use to improve security and privacy without advertising it aggressively. These findings are in line with results of previous works [3, 4] which find that many users do not use end-to-end encryption for security reasons but adopt it along with other features. Therefore, future development should look into adding more value to the use of email encryption on top of security and privacy. For example, more widely accepting digitally signed emails for administrative processes could increase the value of email encryption for a broader set of users.

Use of Insecure Keys For both S/MIME and PGP, we found only small numbers of insecure or outdated keys (cf. Table 3.2). The majority of keys used sufficient key sizes (>= 2048 bits) for RSA and Elgamal and >= 1024 bits for the DSA algorithm. However, we found a significant number of keys that had no expiration date set or were used after their expiration date. In particular, most PGP keys were affected (32.99% had no expiration date and 35.93% were expired). While the widespread adoption of secure algorithms and sufficient key sizes is positive, the prevalence of key expiration issues is a matter of concern. These findings suggest that initial key generation by email encryption clients and plugins works well and supports email users in setting up sufficiently strong encryption keys. However, they also illustrate that tool support to prevent the use of insecure and expired keys has limits. Therefore, future development should look into better prevention mechanisms that nudge users into only using secure and still valid keys.

Impact of Certain Events Our longitudinal field data allows us to shed light on the impact of specific events in time on the use of email encryption. In the following, we focus on two events: The Snowden revelations in 2013 and the COVID-19 pandemic in 2020. While this list is not exhaustive, both events had significant impact on the security community.

In 2013, Edward Snowden leaked the mass-surveillance program of the NSA and other security agencies [69, 70]. Users of modern digital communication tools were made aware of the significance of end-to-end encryption for information security. Compared to 2012 (0.035%), the use of email encryption doubled to 0.07% in the following years. While the impact of the revelations on absolute numbers is limited, the use of email encryption still significantly increased. This incident illustrates that awareness campaigns may positively affect the adoption of end-to-end encryption which is in line with an unprecedented growth of Signal users after a WhatsApp controversy in January 2021[16].

During the COVID-19 pandemic, we found that more emails were exchanged than before. This can be explained by the fact that most of the institution's staff worked from home and many administrative processes that had been paper-based before the pandemic were digitalized in 2020. Hence, for all emails we noted an increased amount of emails sent (39% compared to 2019). However, the number of S/MIME-signed emails also increased by almost 76% from 255,104 to 449,646 compared to 2019. In contrast, the amount of encrypted emails slightly dropped from 0.061% in 2019 to 0.050% and 0.051% in 2020 and 2021, respectively. Users might have had problems setting up and using email encryption from home and accessing the university's technical support during the pandemic. Similarly, about one-third fewer certificates were issued to users in 2020 compared to the previous year (cf. Figure B.3).

Key Management Challenges We identified multiple key management challenges. While we found limited use of email encryption in general, the use of multiple clients posed additional challenges for S/MIME users in our dataset (cf. Section 3.5.3) – single client S/MIME users signed more than 60% while users with two clients only signed 1.58% of their emails, which might be caused by a lack of tool support for transferring S/MIME setups between clients. Similarly, we identified challenges for long-term key and certificate management. 42 out of 364 certificate

[16]cf. `https://www.reuters.com/article/us-signal-users-idUSKBN29I27U`

rollovers occurred months after the certificates' expiration dates, which made it impossible to produce valid signatures in the meantime. Better tool support for easier key rollovers could contribute to earlier certificate updates for a broader set of users. However, we also identified a positive aspect of key management. Only three emails had private keys attached, all of which were sent to the users' own mailboxes. Hence, in no case was private key material leaked to unauthorized users. Overall, while our data implies that key management needs to be improved, private key leakage through email attachments was not an issue for our users.

Missed Opportunities In addition to the key management issues above, we identified the initial distribution and use of keys and certificates to be challenging for users. In particular, using already exchanged public keys between users for further email encryption was a surprising issue we found. While we identified mutual public S/MIME key exchanges between 374 users, they only encrypted 13.95% of their future emails. Overall, 318,214 emails in our dataset could have been encrypted between these users without any additional key exchange. Here, our data illustrates that an automated mechanism for more efficient opportunistic encryption similar to Autocrypt could potentially to help to increase the number of encrypted emails.

3.7 Conclusion

In this work, we presented the first analysis of a large corpus of longitudinal email data for thousands of users at a large German university. We were able to confirm common beliefs and results from previous work in the security community: Only few users used email encryption to secure only a small fraction of their emails. We identified key management to be challenging in particular in the context of multiple clients, key rollovers and key exchange. Based on our evaluation, we make suggestions for improving email encryption adoption. Overall, we hope our investigation provides a data driven motivation for future work to improve both the security and usability of email encryption solutions.

In order to understand the landscape of end-user use of encrypted communication, in this paper, we presented a large-scale longitudinal case study of the (non) use of encrypted email for our whole university. This chapter showed that, for a large university, encrypted email communications are extremely rare, with signed emails being slightly more common than encrypted emails. While use of email trends upward over time, adoption of signatures and encrypted email remains low. We identified possible usability challenges to encrypting email, specifically problems with key management, key rollover, key exchange, and multiple device use. Some of these may be addressed by existing measures like Autocrypt, but are subject to generally low adoption.

While emails are not end-to-end encrypted by default, modern messengers often use end-to-end encryption automatically. If users use these for all communication, all their data, including sensitive information, is end-to-end encrypted in transit. This beats the level of encryption that has been reached by the low number of adoptees of end-to-end encrypted email, as we have seen in this chapter. However, if users do not understand or trust encrypted messengers, and therefore fall back on unencrypted communication methods like plain email or SMS for sensitive information, they lose the benefit of secure messaging. Therefore, in a follow-up study, we explore how users understand and notice by-default end-to-end encryption in messengers. In Chapter 4, we explore user perceptions of encryption disclosures.

Chapter 4

On the Limited Impact of Visualizing Encryption: Perceptions of End-to-End Messaging Security

Disclaimer: *The contents of this chapter were previously published as part of the conference paper "On the Limited Impact of Visualizing Encryption: Perceptions of E2E Messaging Security", presented at the Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021). This research was conducted as a team with my co-authors Dominik Wermke, Johanna Schrader, Nicolas Huaman, Yasemin Acar, Anna Lena Fehlhaber, Miranda Wei, Blase Ur, Sascha Fahl, and Marcel Jahnke; this chapter therefore uses the academic "we". I conceived of the idea for this paper and implemented the changes to Signal, supported by Marcel Jahnke. I co-designed the studies with Dominik Wermke, Yasemin Acar, Blase Ur, and Sascha Fahl. Johanna Schrader and Miranda Wei co-designed all but the fifth study. The studies were conducted by me; qualitative results were coded by me, Dominik Wermke, Johanna Schrader, and Anna Lena Fehlhaber. Dominik Wermke and I conducted the quantitative evaluation. The paper was written jointly with Dominik Wermke, Nicolas Huaman, Yasemin Acar, Blase Ur, Sascha Fahl, and me. Yasemin Acar, Blase Ur, and Sascha Fahl supervised throughout. The tool was based on a 2019 Signal version and the 2019 WhatsApp design.*

Abstract

Communication tools with end-to-end (E2E) encryption help users maintain their privacy. Although messengers like WhatsApp and Signal bring end-to-end encryption to a broad audience, past work has documented misconceptions of their security and privacy properties. Through a series of five online studies with 683 total participants, we investigated whether making an app's end-to-end encryption more visible improves perceptions of trust, security, and privacy. We first investigated why participants use particular messaging tools, validating a prior finding that many users mistakenly think SMS and email are more secure than end-to-end encrypted messengers. We then studied the effect of making end-to-end encryption more visible in a messaging app. We compared six different text disclosures, three different icons, and three different animations of the encryption process. We found that simple text disclosures that messages are "encrypted" are sufficient. Surprisingly, the icons negatively impacted perceptions. While qualitative responses to the animations showed they successfully conveyed and emphasized "security" and "encryption," the animations did not significantly impact participants' quantitative perceptions of the overall trustworthiness, security, and privacy of end-to-end encrypted messaging. We confirmed and unpacked this result through a validation study, finding that user perceptions depend more on preconceived expectations and an app's reputation than visualizations of security mechanisms.

4.1 Introduction

The use of end-to-end encrypted communication tools for email (e.g., PGP [164], S/MIME [122]) or for mobile apps (e.g., WhatsApp [158], iMessage [16], Signal [131]) is an effective countermeasure against cybercriminals, nation-state attackers, and other adversaries [86]. Most end-to-end encrypted communication tools provide confidentiality, integrity, authenticity, and perfect forward secrecy [45] for message contents, but do not hide metadata like sender/receiver identities or when the message was sent [112]. Many previous studies have documented usability and adoption challenges for encryption tools [17, 28, 33, 141], especially for email encryption [62, 63, 115, 118] and modern end-to-end encrypted messaging apps [3, 4].

Of particular concern is that users often have flawed mental models of end-to-end encrypted tools' security and privacy properties. This can lead users to mistakenly use less secure alternatives like SMS or email for confidential conversations even when they already have access to end-to-end encryption through widely used tools like WhatsApp and iMessage [3].

Recent work has highlighted how increasing the visibility of typically invisible security mechanisms can improve user perceptions of trust and security. For example, a qualitative study on e-voting found that displaying security mechanisms improved both user experience and need fulfillment [38]. In the context of end-to-end encryption on Facebook, another study's qualitative results suggested that visibly transforming Facebook messages to and from ciphertext (an implementation artifact in that work) appeared to increase user trust and perceptions of security [49]. For email security, studies found that clearly labeling PGP-encrypted email differently from unencrypted email improved usability and perceived security, as well as reduced unintentional human error when interacting with PGP-encrypted emails [116, 119]. Our work tests these promising results in the space of mobile messaging apps. In an attempt to improve user comprehension and perceptions of security, privacy, and trust for end-to-end encrypted mobile messaging apps, we thus investigated visualizing encryption through various text descriptions, icons, and animations of the encryption process.

We conducted a series of five user studies on MTurk and Prolific to investigate the following three research questions:

RQ 1: Which messaging tools do people prefer for confidential communications, and why?

Of participants who had an end-to-end encrypted tool installed (80), 62.50% reported they would use a tool without end-to-end encryption for confidential conversations, echoing prior work [3]. This finding suggests that end-to-end encrypted communication tools can do more to discourage users from switching to less-secure tools in situations when security and privacy matter. Our root-cause analysis revealed factors like specific UI features, trust in companies, and security misconceptions contributed to participants' decisions.

RQ 2a: How does visualizing encryption through text, icons, or animations impact perceptions of end-to-end encrypted messaging tools' security, trust, and privacy?

RQ 2b: What external factors and expectations mediate encryption visualizations' impact on user perceptions?

In an attempt to highlight tools' end-to-end encryption, we investigated three types of visualizations: *text disclosures*, *icons*, and *animations*. In our remaining online studies, we investigated the impact of different variants of these visualizations. While some of these disclosures have been investigated previously, the animations are especially novel, as is our application of a consistent human-subjects protocol to study all three types.

We found that perceptions of a tool's security, trust, and privacy increased as soon as there was a simple indicator of encryption, such as a text statement that messages are encrypted (similar to WhatsApp's current interface). Contradicting the recent literature, additional emphasis did not appear to have much impact. More concretely, heavyweight animations and icons did not appear to provide much benefit beyond a lightweight text disclosure in emphasizing end-to-end encrypted messengers' security properties to users. While qualitative data suggested that rich visualizations like animations successfully emphasized security and encryption, they did not significantly impact quantitative measures of user perception. Notably, much of the recent literature relies on qualitative observations, whereas our dual use of both perspectives highlights limitations of visualizing security. Through a final study with additional questions, we validated the surprising lack of a quantitative effect and further unpacked users' expectations.

In this work, we make the following contributions:

- We detail which end-to-end encrypted communication tools participants use in different situations, and why.
- We investigate how visualizing encryption through text disclosures, icons, and animations impacts perceptions of security, privacy, and trust.
- We validate our findings and unpack the limitations of visualizing end-to-end encryption.

The rest of the chapter is structured as follows. Section 4.3 presents previous work relevant to this chapter and illustrates the novelty of our research. Section 4.4 provides detailed information on our methodology, including data quality and data analysis techniques we applied, as well as the ethical considerations and limitations of our work. In Section 4.5, we discuss the procedure and findings of our first study on the use of communication tools. Section 4.6 gives a detailed overview of the experiments we conducted on different visualizations of encryption, and Section 4.7 describes a validation study. Section 4.8 discusses our results, highlights their implications for secure messaging applications, and outlines possible future work. Finally, we conclude in Section 4.10.

4.2 Background on Secure Messaging

Modern digital communication tools come in different shapes and flavors. Many people use email, SMS, or messaging apps to communicate with one person or in groups. Communication tools are available on desktop and laptop computers and mobile devices and offer different sorts of security and privacy features [148]. While email, SMS and most messaging apps do not provide content encryption at all or only transport layer security using TLS [107] or other transport layer security [44], some messaging apps provide end-to-end encryption as default [16, 131, 153, 158] or an opt-in feature [46, 87, 122, 144, 164]. In contrast to no encryption or transport layer security, end-to-end encrypted communication tools allow their users to protect conversations against unauthorized access from service providers and other possible man-in-the-middle attackers. We provide an overview of the most popular communication tools in the U.S.[1] and less popular end-to-end encrypted tools in Table C.2. Table C.2 includes information about deployed end-to-end encryption protocols [2], implemented security indicators, platform availability and popularity on Android.

[1]We took this list from Statista [135] and extended it with email, SMS, iMessage and partly taken from Abu Salma et al.'s work [3]

[2]For a detailed discussion of end-to-end encryption protocols, we refer to [148].

4.3 Related Work

Note: This section assumes that you have read the related work section in section 1.3. It is slightly shortened from the original version of the paper.

We discuss related work on encrypted communication tools' usability, adoption, and perception, and previous attempts to visualize security, especially encryption.

Usability of End-to-End Encryption In two-person lab sessions, Ruoti et al. examined initial user experiences for three secure email systems (Pwm, Tutanota, Virtru) through role-play scenarios with 50 participants. They found that participants were interested in secure email in the abstract, but were unsure when and how they actually would use it. Only a few participants desired to use secure email regularly [115]. De Luca et al. conducted online studies and interviews to investigate the role of security and privacy in people's decisions to use secure messaging apps. They reported that peer influence primarily drove decisions to use a particular secure messaging app; security and privacy were minor factors [31].

A number of prior research studies utilized interviews [4, 12, 22, 66, 161] or surveys [3, 12] to investigate users' mental models of end-to-end encryption. Similar to the findings of our first of five studies, these works identified a number of misconceptions regarding the security properties of end-to-end encryption. We based some of our survey questions on this prior work in an attempt to gain deeper insight into the root causes of users' security misconceptions and to try to mitigate such misconceptions.

Visualizing Encryption We discuss literature on visualizing encryption in three areas: web, email, and messaging.

Visualizing and highlighting whether or not webpages are SSL/TLS-encrypted was historically a major focus of usable security research [134, 140]. In a lab setting, Whalen et al. conducted an eye-tracking study with 16 participants to test visual cues for SSL warnings, finding that icons provide prominent visual cues, yet they must be large and prominently placed [156]. Accordingly, we designed sufficiently large cues and placed them prominently in the center of our messaging app. Both Sobey et al. [133] and Maurer et al. [82] investigated alternative display methods, including full-browser themes, as security indicators of extended validity certificates. They found that additional indicators of the level of security improved user confidence, the ease of finding information, and user understanding. Based on their work, we tested a number of variations for each type of visual cue. Schechter et al. conducted a qualitative lab study with 67 participants about the effect of removing security indicators on a banking website [125], finding that users ignore security indicators and that study designs incorporating role-playing reduce participants' security behaviours. More recently, in 2016 Felt et al. conducted a large quantitative online survey with 1329 participants, testing multiple cryptography-related labels and icons.

They arrived at three indicators consisting of icons and labels for valid and invalid HTTPS and HTTP certificates to visualize the security level of the connection [55]. We built on this prior work by applying a similar but extended approach to the area of encrypted messaging apps, including the addition of qualitative elements and a validation study.

In the context of encrypting email, related work investigates how user errors can be prevented and perceptions of security can be improved using security indicators. Two recent connected studies from 2013 and 2015 by Ruoti et al. proposed a web interface to support PGP encryption [116, 119]. They found that visualizing encryption using labels and adding scrambled text as an indicator of encrypted text helped to reduce user error when using PGP and supports trust in email encryption. They proposed to further improve trust by letting users copy and paste email ciphertext, but in a followup study found that doing so had no measurable effect on usability or security perceptions. Garfinkel et al. found that Key Continuity Management (KCM) systems with color-coded messages could improve email security and effectively help novice users identify signed emails [63]. In 2015, Atwater et al. conducted a lab study investigating how a web interface can support email encryption [17]. They found that participants prefer PGP to be integrated into their existing tool (e.g., Gmail). Participants' trust perceptions were based not on the tool's design, but rather the tool's overall reputation. Based on this finding that encryption should integrate into existing and well-known tools, we chose to test our own indicators using a modified version of the highly popular, end-to-end encrypted WhatsApp Messenger.

Finally, we discuss related work regarding instant messaging and mobile apps. In 2012, Fahl et al. designed a tool for end-to-end encryption of private Facebook messages, evaluating the tool through lab and interview studies [49]. An artifact of their tool's implementation was that participants would see plaintext Facebook messages being translated to and from ciphertext. Their qualitative results implied that participants seeing the ciphertext upon sending or receiving messages was viewed positively and seemed to increase trust in the tool's security properties. In a lab study of the SELENE electronic voting protocol, Distler et al. [38] investigated how users reacted to seeing an explanation of encryption during the voting process. They found that overall perspicuity and users' perceptions of security increased due to the added waiting screen. In a followup online survey [37], they also tested different wordings of encryption in the scenarios of e-voting, online pharmacies, and online banking. They concluded that explanations of encryption should consist of short text without many elements, underpinning the design of the text disclosures we tested in Section 4.6.2.

In 2018, Demjaha et al. conducted an online study with 96 participants investigating metaphors to explain end-to-end encryption to users [34]. They concluded that wordings like "encryption" might be overloaded for end-users and alternative metaphors might better explain the strengths and weaknesses of end-to-end encryption. While we focus on differences in structural explanations, we implement some metaphorical approaches in our icons and animations, measuring their effects compared to more straightforward labels and icons. Schröder et al. investigated authenticity-related error messages for the Signal[131] Android app [126]. They conducted a mostly qualitative study with 28 participants, finding that Signal needs to improve the awareness and verification of authenticity in conversations, as well as to communicate risks more clearly (e.g., providing guidelines for handling potential MITM attacks). Their findings suggest that the security perceptions of Signal could be improved in general.

In a recent study Akgul et al. evaluated if in-workflow messages in a messenger could improved the mental models of end-to-end encryption and found that while participants noticed them, they did not pay much attention to it, which limited the effect [12].

4.4 Methodology

We conducted a series of online studies on MTurk and Prolific (cf. Figure 4.1). This section gives a high-level overview of our approach. Section 4.5 details our study investigating current use of communication tools. Section 4.6 describes our studies on how different designs of encryption visualizations impact user perceptions.

Overall, we conducted five different user studies with 683 participants. For the first four, we recruited on MTurk. For the fifth, which was our validation study, we recruited on Prolific. We required participants in studies 2–5 be experienced WhatsApp users, enforcing this requirement through a qualification task on MTurk (cf. Section 4.6) and Prolific's built-in participant filters. We decided to use WhatsApp for our studies, since it is the most commonly used messenger with end-to-end encryption enabled by default in the US that is available on multiple platforms [135]. We estimated required participant numbers for each survey using power analysis and were limited by the total number of available WhatsApp users.

Each study had a distinct purpose:

Study 1: Use of Communication Tools The purpose of this study was to gain insight into the selection of communication tools for both day-to-day and confidential conversations. We aimed to understand how and why users decide to use certain tools in particular circumstances. Table C.2 illustrates messengers that were considered in this chapter, and their features. Based on previous work [63, 119, 160], the results of Study 1, and the visual design of modern secure messaging apps, we then implemented potential encryption visualizations in a modern secure messaging app. Our goal was to investigate whether adding encryption visualizations to end-to-end encrypted messaging app's UI would increase perceptions of trust, security, and privacy without sacrificing usability. The results for this study can be found in section 4.5.

Study 2: Disclosures Current secure messaging apps use specific textual framing (disclosures) to inform their users that conversations are end-to-end encrypted. For example, WhatsApp displays *"Messages to this chat and calls are now secured with end-to-end encryption."*. However, prior studies on private browsing modes [162] and security warnings [53] have illustrated users' confusion about analogous disclosures. Therefore, we aimed to investigate whether a more detailed and technically correct ("end-to-end encrypted") disclosure had a different contribution to perceived security than more generous disclosures that are still connected to messaging security and comparatively tested six different versions. The results for this study can be found in section 4.6.2.

Study 3: Icons In addition to disclosures, a common approach is the use of security icons (e. g., lock symbols) [55] to indicate the presence of encryption or other security mechanisms. Similar to Study 2, we based our analysis on current secure messaging apps' security icons and icons discussed in previous usable security papers [55, 125]. We investigated three different icons, studying their impact on perceived trust, security and privacy, and usability. The results for this study can be found in section 4.6.3.

Study 4: Animations Additionally, we implemented and studied three animations of encryption. Prior work [38, 49] and the results of Study 1 implied that dynamic animations of the encryption process (e. g. disappearing messages or animations of plaintext turning into ciphertext) might increase perceptions of trust, security, and privacy. The results for this study can be found in section 4.6.4.

Study 5: Validation To validate and clarify the findings from studies 1–4, we performed a fifth study that addresses limitations of the previous four. One key challenge of studies 1–4 is the demographic bias of Amazon MTurk. Recent research identified generalizability and data quality issues on MTurk[75]. To account for this, we switched recruitment platforms, choosing Prolific[102]. Prolific provides strong tools to obtain a more diverse sample. Additionally, we performed the validation study to investigate root causes of particular results of Studies 2–4, so we also added an additional control condition and qualitative questions. To remove a potential confound suggested by the results of Studies 2–4, we also changed the messaging app from WhatsApp to a fictitious app we called Erebus. The results for this study can be found in section 4.7.

4.4.1 Study Procedure

We conducted all five studies sequentially to allow the findings of preceding studies to inform the design of later studies. For example, we used the most promising text disclosure from Study 2 in Studies 3–5.

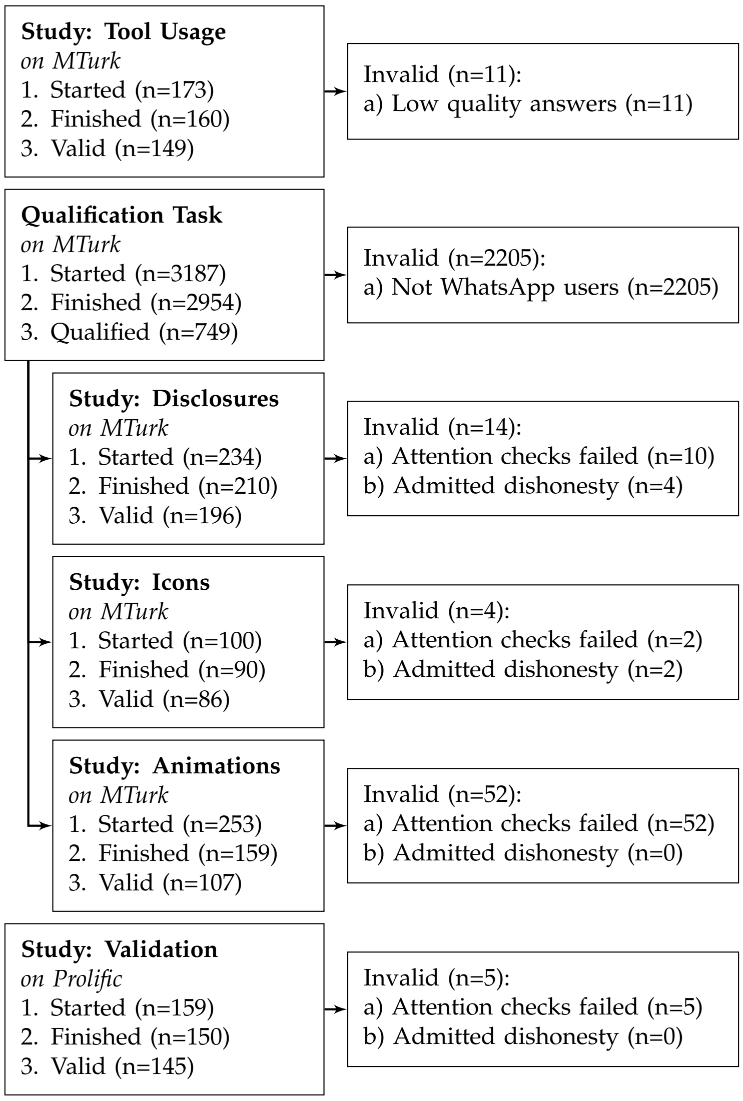


FIGURE 4.1: Illustration of our research procedure including *survey platform*, number of participants, and dropouts.

Lab vs. Online Study Across studies 2–5 we investigated six different text disclosures (cf. Section 4.6.2), three different icons (cf. Section 4.6.3) and three different dynamic animations (cf. Section 4.6.4). Consequently, we recruited a rather high number of participants ($n = 534$ total participants). This made a laboratory experiment infeasible. Hence, we decided to conduct our experiments online using Amazon Mechanical Turk and Prolific Academic. Both platforms are popular amongst usable security and privacy user studies [1, 72, 89, 149].

Mockups vs. Real App We aimed for high internal validity to ensure that font sizes, types, positions of icons, animations, and the content of conversations (cf. Figure C.4) remained consistent for all participants. Hence, we decided to use mockups instead of asking participants to install a real app on their devices. For the mockups, we created screencasts by forking the Signal Android app [131], since it implements

Factor	Description
Required	
Condition	Disclosures, icons, or animations (baseline: Control)
Optional	
CS Edu	Has CS education (self-reported, baseline: No)
CS Job	Has CS job (self-reported, baseline: No)
Age	Age in years (self-reported)

TABLE 4.1: Factors used in regression models. Model candidates were defined using all possible combinations of optional factors, with the required factors included in every candidate. Final models were selected by minimum AIC. Categorical factors are individually compared to the baseline.

the same encryption workflow that WhatsApp uses. We implemented the WhatsApp look and feel and all encryption visualizations. We recorded screencasts using the app and the conversation in Figure C.4. During the conversation, we presented the different visualizations in each condition.

Additionally, each study had an online survey questionnaire at the end. The survey questionnaire addressed the perceived usability, trust, security, privacy and satisfaction with the tool, tool preference for both day-to-day and confidential conversations and demographic information about our participants.

Because we expected significant learning effect across conditions, Studies 2–5 followed a between-groups design.

Pre-Testing Before we conducted the studies, we pre-tested our questionnaires and screencasts, following best practices for cognitive interviews [58]. To glean insights into how survey respondents might interpret and answer questions and how they perceive the screencasts, we asked participants to share their thoughts as they answered each survey question and watched the screencasts. We used the findings to iteratively revise and rewrite our survey questions to minimize bias and maximize validity and modify the screencasts based on the feedback. We conducted cognitive interviews with members of our research group and university students, and performed a pre-test on MTurk to evaluate our survey questions under realistic conditions and to calibrate compensation relative to the time required. Pilots took an average of 15 minutes, so we compensated participants \$2.50 (an hourly wage of \$10).

Data Analysis Prior to data analysis, we took measures to ensure data quality (cf. C.3.2).

We perform both quantitative and qualitative data analysis. Throughout the paper, we measure usability using the UMUX Lite questionnaire [80]. We compare responses to the UMUX Lite across conditions with Pearson's chi-squared test (χ^2).

We also collect net promoter scores, which are a quantitative measure of willingness to recommend a product. As these scores are continuous, we use the non-parametric Kruskal-Wallis H test (KW-H) for comparing conditions.

Because they might be influenced by multiple distinct factors, we analyze participants' perceptions of trust, security, and privacy by fitting linear regression models. For each regression analysis, we consider a set of candidate models and select the model with the lowest Akaike Information Criterion (AIC) [24]. We consider candidate models consisting of the "condition" (indicating the particular text disclosure, icon, or animation tested) plus every possible combination of optional factors. Required factors, optional factors, and corresponding baseline values are described in Table 4.1.

We present the outcomes of our regressions in tables where each row contains a factor and the corresponding change of the analyzed outcome in relation to the baseline of the given factor. Linear regression models measure change from baseline factors with a coefficient (*Coef.*) of zero for the value of the outcome. For each factor of a model, we also list a 95% confidence interval (*C.I.*) and a p-value indicating statistical significance. Also, we highlight p-values below $\alpha = 0.05$ with an asterisk (*).

We analyzed all free-text responses in an open-coding process [29, 139]. Two researchers iteratively developed a codebook [26], then used this initial codebook to code all free-text responses simultaneously, resolved coding conflicts, and incrementally updated the codebook until they were able to code open-ended questions without modifications to the codebook. The codebook remained stable once both researchers were satisfied that all important themes and concepts in the responses could be captured with the codes. Since the researchers resolved conflicts immediately as they emerged, we do not calculate inter-coder agreement [76].

4.4.2 Limitations

As with most self-reported online studies, our work has several limitations. In general, self-report studies may suffer from several biases, including over- and under-reporting, sampling bias, and social desirability bias. While we utilize self-report data, our central claims are not about the accuracy of respondents' answers to a given question, but rather about whether and how responses from different conditions differ from each other. Consequently, the threats to validity caused by those biases should apply equally across all conditions.

Conducting user studies on Amazon MTurk and Prolific is a widely used and accepted procedure for this type of research [98, 104]. However, MTurkers are known to be younger and more tech-savvy than the average population [104]. Additionally, our study focuses on the responses of U.S. Internet users, and thus, we can offer no insight into the generalizability of results for international participants.

Recently, the frequency of low data quality on MTurk has been increasing [75]. Therefore, we implemented a number of countermeasures (cf. Section 4.4.1). During data cleaning, we identified several participants who did not pass our quality measures (Figure 4.1) and excluded them from further analysis.

We cannot guarantee that no participants were both registered MTurk and Prolific users and took more than one study, since there is no way to track people across both services. However, this is unlikely, since MTurk and Prolific target different geographic regions, and we conducted only the validation study on Prolific. Hence, we can guarantee that no participant took the same study twice.

Studies 2–5 tested a small set of different text disclosures, icons, and animations. While we based our designs on previous work and the results of our first study, we cannot guarantee that there are not other variants that work even better. Individual studies transpired in a somewhat isolated context, potentially missing certain effects of long-time exposure. We deliberately focused on multiple shorter studies, instead of one single in-depth, long-term study, to gather wider insights with different elements.

We showed our participants short screencasts (videos) in studies 2–5 instead of letting them use a real messaging application on their own devices. We aimed for high internal validity, so we wanted to ensure all participants would receive the same treatment. Comparable related work also worked with mockups instead of real applications for the same reason [38, 49]. While this experimental design results in lower external validity, we consider this tradeoff acceptable.

We decided to use a widely-deployed tool instead of a fictitious app mockup to study the challenges of visualizing end-to-end encryption for an existing service provider and user base. We think our research provides valuable insights for a large set of users, although findings may not generalize to other end-to-end encrypted messaging tools.

4.4.3 Ethical Considerations

We designed our studies with privacy in mind and followed best practices concerning data collection to ensure that we adhere to the German data- and privacy-protection laws aswell as the European General Data Protection Regulation. Our institution does not require a formal IRB, but we designed the study protocol based on a previous IRB approved study. All surveys started with a consent form to inform participants about the purpose of the study and about the data we would collect and store. The consent form also contained contact information to reach the PI in case of questions or concerns.

4.5 Use of Communication Tools (Study 1)

The main goal of Study 1 was to learn which communication tools our participants used and preferred for everyday and confidential conversations, as well as to learn about the decisions they made when using specific communication tools for particular conversations. In particular, we were interested in how many participants already used tools that provide end-to-end encryption by default for everyday conversations. We were especially interested in what fraction of them preferred less secure alternatives to end-to-end encrypted messengers for confidential conversations, and why. The questionnaire consisted of both closed- and open-ended questions. We followed the methodology described in Section 4.4 and developed the survey questionnaire in an iterative process, using pre-tests to improve the questionnaire, data quality, and determine appropriate compensation. We recruited 149 U.S.-based participants on MTurk.

4.5.1 Questionnaire Structure

We asked our participants to answer questions about their current use of communication tools for day-to-day and confidential conversations, as well as decisions they make when they choose one of the tools they have available for communicating with a single person or with groups of people. We decided to ask for specific tools or tool providers to glean insights into real behaviors and decision processes. We administered demographic questions at the end of the questionnaire to minimize stereotype bias [77, 123].

Past Tool Usage We asked participants which communication tools they have used in the last six months. The list of tools included the ten most popular tools in the U.S. [135]. We added iMessage, email, and SMS to the list as popular messaging services that are pre-installed on many mobile devices by default. To better understand participants' choices and glean insights into their underlying mental models, we asked open-ended questions to explain their choices.

Security Assessments We asked participants to rate their perceived level of security when using personal email, Facebook Messenger, WhatsApp, Snapchat, and SMS in the presence of different attackers. We chose these tools based on their popularity [2] and security properties (cf. Table C.2 in the appendix).

Demographics We included several demographic questions about gender, age, ethnicity, education level, employment status, mobile device use, and the Security Behaviors Intentions Scale [42] for each participant. We aimed to assess whether demographic information would affect respondents' answers to the survey questionnaire. We also asked respondents for general feedback on the survey questionnaire.

4.5.2 Findings

We present both quantitative as well as qualitative results for the 149 valid respondents. The reporting of our findings focuses on actual tool usage in the past, insights into the perceptions, and decisions our participants made and their assessment of the security they think popular tools provide. Table C.1 provides an overview of demographic characteristics of the participants in all studies.

Tool Usage Of the 149 participants in this study, the majority used regular email (133; 89.26%), SMS/Text Messages (123; 82.55%) or the Facebook Messenger (114; 76.51%) that do not provide end-to-end encryption by default (cf. Figure C.3). Only a few participants (7) reported having used PGP, S/MIME, or a provider supporting end-to-end encryption to secure email conversations. Few participants (1) indicated prior use of Facebook's "Secret Conversation" feature. Overall, more than half of participants (80; 53.69%) reported use of an end-to-end encrypted communication tool, with WhatsApp (47; 31.54%) being the most popular by far.

Tools that support end-to-end encryption as an optional feature, such as Facebook Messenger, Skype and Telegram (cf. Table C.2), were also widely used (81.88%). However, only a few participants (9.02%) reported having used their end-to-end features.

While email, SMS/text message, Facebook, WhatsApp, iMessage, and Skype are the most popular tools for both day-to-day and confidential conversations (cf. Figure C.1 and Figure C.2), a minority of participants (32; 21.48%) preferred none of the given tools for confidential conversations[3]. They only trusted non-digital forms of communication.

Even though they were users of end-to-end encrypted communication tools for day-to-day conversations, many participants preferred email and SMS for confidential conversations. Of the 80 participants who used end-to-end encrypted tools for communication in general, the majority (50; 62.50%) preferred the use of insecure alternatives for confidential conversations. In particular, most (32; 68.09%) of the 47 WhatsApp users prefer less secure alternatives for confidential conversations.

Reasons for Using a Tool for Day-to-Day Conversations The main reason for people to use a certain communication tool for day-to-day conversation is ease of use (61.49%) followed by the availability of contacts in this tool (49.32%) and convenience (28.38%). One out of four (25.00%) participants also mentioned the delivery speed of text messages or instant messaging services and few (15.54%) mentioned the provided functionality. Some stated they are using a specific tool for a particular circle of people (11.49%) as mentioned by few participants: *"I belong to an online community for work and our main line of communication is through Facebook's messaging service."* (P157), *"My husband uses Google hangouts too, and since I talk to him the most,*

[3]None is an exclusive option and deselected the other fields.

this is the app I use most often." (P23), *"This is a group of family that has them, when I just need to relay info to that group I get on Telegram."* (P27).

Few participants mentioned that they like a tool for storing a conversation history (5.41%), group chats (4.05%), message read info (5.40%) and disappearing messages (1.35%).

Email was an outlier as a preferred communication tool in many ways. Some participants (17.86% of email users) prefer email over other tools because they did not feel forced to reply to emails immediately: *"It's more low key. There are no read receipts and you aren't expected to make a response immediately. You get to take your time."* (P151).

Email has a professional reputation as it is often used in the workplace, which 16.06% of email users noted. For 26.79% of email users, a key reason to use email is the support for large attachments and long text. This differs from all other tools, which are primarily instant-messaging services.

Reasons for Using a Tool for Confidential Conversations In two open-ended questions, we asked participants to elaborate on their preference for a specific tool for sensitive or confidential conversations and how they can tell that a specific tool keeps conversations confidential.

Almost half of participants (45.54%) mentioned a gut instinct that leads to a security belief as their main reason to prefer a specific tool for confidential conversations, e. g. *"I feel that it is safe."* (P36).

A quarter of our participants (25.00%) assumed a tool to be confidential when they send messages directly to their intended contact and had their own name and the name(s) of the communication partner(s) being shown in the user interface. 16.96% mentioned access control and strong passwords as reasons to prefer a particular tool as mentioned by one participant: *"I have a secure E-mail that is guarded by a good strong password."* (P123).

One out of four (26.79%) assumed a tool to be acceptable for confidential conversations because they thought it uses some form of encryption. However, 14.29% made wrong assumptions and thought encryption was being deployed on unencrypted channels (e. g., for SMS/text messages). Interestingly, only a few (8.04%) referenced "secret mode" or "secure chat" options in their decision.

6% of our participants also reported using SMS as a confidential channel because it is not an internet service: *"It is sent from me to another person, not on the internet."* (P31) and *"It feels off the grid, away from the dangers of the internet."* (P66)

For a few (3.57%), visual indicators like colors or icons earned trust even if they did not directly relate to security or privacy e. g. *"If the message is blue it should be encrypted."* (P148). In the iOS messenger, a blue message indicates that a message was sent via iMessage and a green message indicates that it was sent as a Text Message.

Few participants mentioned self-destructing and disappearing messages (4.46%), as in SnapChat, or the ability to delete messages manually (3.57%), as offered in WhatsApp, as influencing their preference:

"*I know that gmail for example encrypts messages and I trust google to be safe.*" (P77)

At the same time, half of the participants (50%) could not report specific reasons for their trust in a particular tool.

Key Insights: Tool Usage, Decisions and Security Beliefs.

- Email and SMS/text messages are the most popular tools for both day-to-day and confidential conversations.
- 53.69% of participants use a communication tool with end-to-end encryption enabled by default.
- 62.50% of participants who use end-to-end encrypted tools prefer less secure alternatives for confidential conversations.
- Participants reported a gut instinct that made them believe a tool to be secure.

4.6 Visualizing Encryption (Study 2–4)

Both previous work and the findings of our first study illustrate that the situation around end-to-end encrypted communication tools is complicated. Many users will avoid installing a new, more secure messaging tool [2] only because it provides better security [121, 160]. Instead, most users only consider messaging tools if their contacts (i.e. friends, family, and colleagues) also use the tools [31]. Additionally, previous work [31, 161], and our first study show many people suffer from misunderstandings and misconceptions of encryption.

Instead of propagating the more widespread use of such niche tools or working on correcting users' misunderstandings and misconceptions alone, we followed a different route. Depending on geographic region, between half of users (cf. Section 4.5) and 90% [2] of users *already* have tools that support end-to-end encryption by default, with WhatsApp being the most popular. However, our findings (cf. Section 4.5) suggest that many users are not aware of these security properties. More than half of our participants who use WhatsApp prefer less secure alternatives such as email or SMS/text messages for confidential conversations. Therefore, the remainder of our studies investigate how visualizing encryption impacts perceptions of end-to-end messaging security.

While the results of our first study (cf. Section 4.5) and previous work [3, 4, 31, 38, 49, 161] uncover a wide range of root causes for misconceptions about the security of messengers and insecure behaviour, only some of them can be addressed in the design of a communication tool. For example, we identified that trusting a company or decades of positive experiences were both root causes for misconceptions. However, these can hardly be addressed in the design of a communication tool. In contrast, there are promising candidates that can directly be implemented in the user interface of a communication tool (cf. Section 4.4). In this section, we describe multiple online studies we conducted with the goal to investigate the impact of different encryption disclosures, icons and animations on perceived trust, security, privacy, usability, satisfaction and self-reported likeliness to use the re-designed communication service for sensitive messages.

4.6.1 Experiment Design

To study visualizations of encryption using text disclosures, icons, and animations, we conducted four between-groups online experiments with WhatsApp users recruited on MTurk or Prolific. Each study follows the procedure we outline below.

Screencasts

To study the impact of different encryption visualizations on usability, perceived trust, security, privacy, satisfaction and tool preference, we decided to show participants a screencast of a ficticious WhatsApp update [4].

Using a screencast instead of static mockup images allowed us to study both static and dynamic encryption visualizations (cf. Section 4.4) and include a scripted conversation to provide more context for our participants[5]. We constructed the messages this way because it mimics a realistic personal conversation and credit card information is generally perceived as confidential and worth protecting.

To mimic WhatsApp as closely as possible, we forked the Android version of the Signal mobile app and adapted the user interface respectively by changing colors, typefaces, buttons and other user interface properties.

Questionnaire Structure

The survey questionnaire in this study was developed through an iterative process (cf. Section 4.5) and included the attention checks mentioned in Section 4.4. Completion of the survey took 10 minutes on average and we paid participants $1.7.

Usability, Trust, Security, Privacy and Satisfaction We asked participants to answer usability, perceived trust, security and privacy and satisfaction questions. For usability, we asked participants the two items UMUX lite scale [80]. Based on prior work [109], we built a 10-item scale of perceived trust, security and privacy (cf. Appendix C.2). Finally, we asked participants to fill out the net promoter score [71] to measure how much they liked the encryption visualization.

[4] cf. Appendix C.4 for the video introduction
[5] cf. Appendix C.3.1 for the conversation

Tool Preference We showed participants a list of the most popular communication tools from our first study (cf. Section 4.5) including the new ficticious WhatsApp version and asked them which tool they would prefer for both day-to-day and confidential conversations. To prevent lock-in obstacles as found in [31], we told all participants to assume that all communication partners have all tools installed. We aimed to assess whether our conditions had an effect on the participants' choice.

Demographics We asked our respondents the same demographic questions as in the questionnaire in Section 4.5. Table C.1 provides an overview of demographic characteristics of the participants in the studies in this section.

4.6.2 Text Disclosures (Study 2)

Based on the disclosures in current tools that support end-to-end encrypted communication (cf. Table C.2) and the results of our first study (cf. Section 4.5), we created six different disclosures out of the terms "secret", "private", "encrypted", "secure" and "end-to-end encrypted." In a between-groups design, we randomly assigned participants to one of the following conditions:

1. Control: *"blank"*
2. Encrypted: *"Messages to this chat are now **encrypted**."*
3. End-to-End Encrypted: *"Messages to this chat are now **end-to-end encrypted**."*
4. Private: *"Messages to this chat are now **private**."*
5. Secure: *"Messages to this chat are now **secure**."*
6. Secure & End-to-End Encrypted: *"Messages to this chat are now **secured** with **end-to-end encryption**."*
7. Secret: *"Messages to this chat are now **secret**."*

To make sure participants read the text of each disclosure, we showed them a screencast in fullscreen before entering the conversation for seven seconds including the respective disclosure. Overall, we recruited 196 valid participants on MTurk for whom we report findings below.

Findings We were specifically interested in the participants' opinions on usability and their perceptions of trust, security, and privacy.

As a usability metric we compared the distribution of UMUX Lite answer categories between our conditions. We found no apparent differences between the conditions (Q1: Pearson's $\chi^2 = 1.56$, p-value $= 1$; Q2: Pearson's $\chi^2 = 1.22$, p-value $= 1$), suggesting no observable effect (positive or negative) on the perceived usability of the different disclosures.

To better investigate how the different conditions affect participants' perception of trust, privacy, and security, we introduced a combined score based on their answers to our set of 10 likert-item questions. For each participant, the score consists of the average of all 10 likert-item questions mapped to numerical values, e.g., between

	Factor	Coef.	C.I.	p-value
Disclosure	*“Messages to this chat are now . . . ”*			
Disclosure	*“. . . private”*	0.27	[0.21, 1.07]	0.392
Disclosure	*“. . . secret”*	-0.49	[-1.09, 0.11]	0.111
Disclosure	*“. . . secure”*	0.06	[-0.53, 0.65]	0.845
Disclosure	*“. . . encrypted”*	0.68	[0.09, 1.28]	0.030*
Disclosure	*“. . . end-to-end encrypted”*	-0.09	[-0.69, 0.51]	0.768
Disclosure	*“. . . secured with end-to-end encryption”*	0.41	[-0.19, 1.01]	0.182
Icon	Icon (Baseline: Control):			
Icon	Envelope	-0.47	[0.92, 1.69]	0.105
Icon	Lock	-0.49	[-1.09, 0.11]	0.089
Icon	Shield	-0.70	[-1.27, -0.14]	0.014*
	CS Education	-0.51	[-0.97, -0.05]	0.029*
Animation	Animation (Baseline: Control):			
Animation	Disappearing Messages	0.08	[-0.34, 0.51]	0.707
Animation	Encryption/Decryption	-0.01	[-0.40, 0.38]	0.969
Animation	Progress Circle	0.25	[-0.14, 0.66]	0.210
	Age	-0.01	[-0.14, 0.65]	0.119
Validation	Animation (Baseline: Control without Disclosure):			
Validation	Control	0.45	[0.03, 0.86]	0.034*
Validation	Disappearing Messages	0.40	[0.01, 0.79]	0.043*
Validation	Encryption/Decryption	0.43	[0.04, 0.81]	0.030*
Validation	Progress Circle	0.71	[0.33, 1.10]	< 0.001*

TABLE 4.2: Results of the linear regression model examining whether different texts, icons and animations have an effect on the trust, security and privacy score in relation to a control baseline. Note the additional “Control” variable in the last study, due to “Control Without Disclosure” being the baseline. See Table 4.1 for further details.

-2 (Strongly Disagree) and +2 (Strongly Agree). For these scores, we considered a set of linear regression models consisting of the conditions as required factor and all combinations of optional factors listed in Table 4.1 and selected the model with the lowest AIC.

The final model (see Table 4.2) shows that the “encrypted” condition is significant with an overall positive coefficient of about 0.7 score points compared to the control baseline. This suggests significantly higher scores for the “*Messages to this chat are now encrypted*” disclosure compared to the blank control, which is in line with previous research by Distler et al. [37]. Participants seemed to prefer the encryption text, likely due to not fully understanding the term "end-to-end," or regarding it as a subset (i.e., less secure) of being "just" encrypted.

For the net promoter score, we found that no condition dominates any other (Kruskal-Wallis $H = 7.88$, p-value $= 0.24$). Based on these results, we proceeded with the “encrypted” text for our subsequent disclosure.

Key Insights: Disclosures.

- Participants felt most secure and private within the “encrypted” disclosure condition.

- The different disclosures did not have a significant impact on usability and satisfaction.

4.6.3 Icons (Study 3)

Next, we investigated three different icons. We chose a lock, a shield, and an envelope (cf. Figure 4.5) based on their typical usage in security and privacy contexts [15, 94], previous work in the field of security indicators [55, 125], and results from our first study (cf. Section 4.5). Together with the best-performing text disclosure from the previous study *"Messages to this chat are now encrypted,"* we showed participants one of the icons before the communication partners in the screencast entered the conversation.

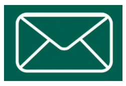


FIGURE 4.2: Envelope

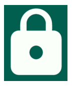


FIGURE 4.3: Lock

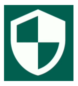


FIGURE 4.4: Shield

FIGURE 4.5: Designs used in the encryption icons study.

We showed participants a screencast including the "encrypted" disclosure from the previous study and the respective encryption icon. All screencasts lasted 94 seconds. A total of 86 WhatsApp users participated in this study. Findings reported below are limited to these valid participants.

Findings For the UMUX Lite questionnaire we found no significant differences across conditions (Q1: Pearson's $\chi^2 = 0.54$, p-value $= 1$; Q2: Pearson's $\chi^2 = 0.40$, p-value $= 1$).

Our set of linear regression models for the overall score included the icon condition as required factor and again all combinations of optional factors (cf. Table 4.1). To our surprise, the final model (See Table 4.2) shows that all three icon conditions are worse than the baseline by at least 0.47 score points. The shield condition is significantly worse by 0.7 score points. In addition, the optional computer science education factor is significant with a negative coefficient in the model. This is in line with previous work [55, 125] and additional evidence for the very limited effect of security icons on perceived trust, security and privacy.

For the net promoter score, we found that no condition dominates any other (Kruskal-Wallis $H = 5.68$, p-value $= 0.128$). We chose to proceed to the next study using the control (no icon) due to the negative coefficients of all other conditions.

Key Insights: Security Icons.

- We found a negative effect of security icons on perceived trust, security and privacy by at least 0.47 score points compared to the baseline.
- Participants with a computer science background particularly disliked the security icons we investigated, resulting in 0.51 less score points compared to participants without that background.
- The security icons had no impact on usability and satisfaction.

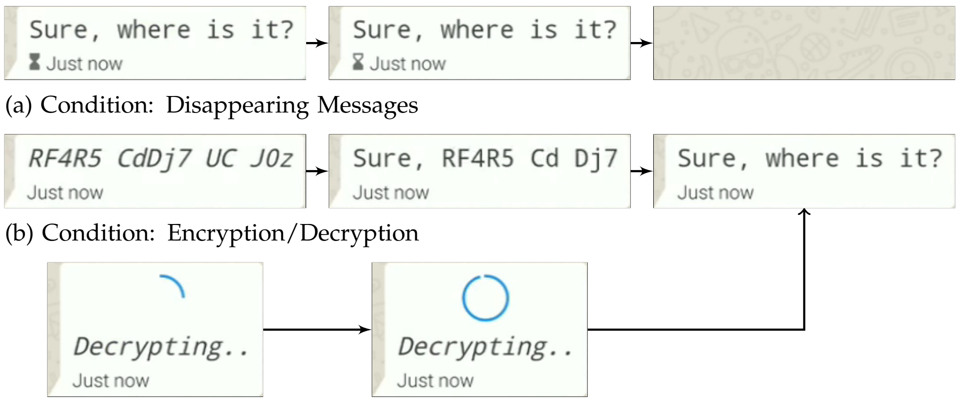
4.6.4 Animations (Study 4)

In addition to text disclosures and encryption icons, previous work [38, 49] and the results of our first study (cf. Section 4.5) suggest the use of animations of the encryption process to convey that a conversation is secure. We identified three different encryption animations: (i) Distler et al. [38] used a progress circle for an e-voting app; (ii) Fahl et al. [49] studied dynamic encryption and decryption animations to protect Facebook messages; and (iii) participants in Study 1 reported feeling particularly secure with disappearing messages on apps like Snapchat. Although disappearing messages are not technically connected to end-to-end encryption, we included them due to their contribution to perceived messaging security identified in previous work by Roesner et al. [110] and participants' comments in Study 1. One participant for example said it was security relevant *"Because the conversation deletes right after I read it."* (P9) and another said *"... once you open it, it's gone forever afterward."* (P14)

We implemented those three animations (cf. Figure 4.6). In contrast to Studies 2–3, we applied the dynamic encryption animations to the screencast conversation's messages, rather than as a fullscreen hint before entering a conversation. We pretested the animation duration with 20 MTurkers. Initially, we showed the animation for three seconds. Based on UMUX Lite and qualitative feedback, we gradually reduced the duration to one second. Based on the results from the previous studies, each condition included one of the encryption animations and a small text hint with the "Encrypted" disclosure. Based on our Study 3 results, we did not use an icon in this study.

Overall, we recruited 107 valid participants on MTurk for whom we report findings below.

Findings We found no impact of the different animation conditions on the UMUX Lite questionnaire (Q1: Pearson's $\chi^2 = 0.37$, p-value $= 1$; Q2: Pearson's $\chi^2 = 0.42$,



(a) Condition: Disappearing Messages

(b) Condition: Encryption/Decryption

(c) Condition: Progress Circle

FIGURE 4.6: Conditions in the security animations study.

p-value = 1). The final linear regression model includes a somewhat increased, but not significant, coefficient (0.25) for the "Progress Circle Animation" condition compared to the baseline, and almost non-existent positive and negative effects (0.08 and -0.01) for the other two animations (cf. Table 4.2). For the net promoter score of the animation conditions, we found again, as in the other two studies, that no condition dominates any other (Kruskal-Wallis $H = 1.63$, p-value $= 0.654$).

Key Insights: Security Animations.

- No factors were significant. The progress circle animation had a weak positive effect on perceived trust, security and privacy.
- Security animations did not impact usability or satisfaction.

4.7 Validation (Study 5)

Due to Study 4's inconclusive findings regarding the effect of animations (cf. Section 4.6.4), we validated our overall findings in a fifth study. Motivated by the lower data quality encountered in Study 4 compared to Studies 1–3, as well as the increasing difficulty recruiting WhatsApp users on MTurk, we switched to the Prolific recruitment platform for Study 5. Prolific provides fine-grained participant demographics, so we could directly target participants with messenger experience without requiring a qualification task. Since Prolific's pool of US workers who use WhatsApp was small (<500), we included UK participants, increasing the participant pool by 6,000. On Prolific, we recruited 145 participants from the UK and US, paying £2.70. In addition to changing recruitment platforms, we decided to delve into the root causes of why visualizing encryption appeared to have a limited impact on user perceptions. For this, we dropped both the UMUX Lite and net promoter score as we observed no significant differences for them in our previous studies. We also added open-ended followups to each Likert question to gain deeper insights into participants' opinions. Additionally, we added Likert questions that focused on what facets suggest that messages are being sent securely. To eliminate participants' perceptions of WhatsApp as a major factor guiding their perceptions, we also changed the messenger name to "Erebus." As this study was intended as validation, we especially focused on the "encryption" text from our first survey by including a new control that displayed no text at all ("Control without Text"). We retained the previous control condition to compare with the previous studies.

Findings As for the previous surveys, we generated a linear regression model listed in Table 4.2. The final regression models have significant non-zero coefficients for all included variables relative to our new control (not mentioning encryption at all). Going by coefficient, the Progress animation performs best compared to the baseline (0.71), followed by Encryption/Decryption (0.43) and Disappearing (0.40). Even the control condition from the previous surveys ("Control") shows a significant coefficient compared to the newly introduced baseline "Control without Text". This suggests a significant effect of the "encryption" text.

In addition to the regression analysis, we evaluated the open-ended questions to gain insight into the limited impact of encryption visualization. We report findings below.

Observing Animations In an open-ended question, we asked participants what they observed happening (if anything) when messages were sent or received, as well as what this indicated. Almost all participants described the animations we showed them ($>$ 90% in each condition). 60 participants (41.38%) wrote that the animation indicated an increased level of security. Hence, we can eliminate the possibility of participants ignoring the animations as the reason perceptions did not vary significantly across conditions.

Identifying Security In an open-ended question, we asked participants how they determine, in general, that a messaging app sends messages securely. The most prominent indicator for security was the *reputation* (48, 33.10%) of the service provider, followed by the mention of *encryption* (42, 28.96%).

"Honestly, I guess I just trust in the brand that it's safe. I do this through the popularity, good press and confidence in their service." (P18)

The relatively similar relevance of *encryption* and *reputation* for perceived security also explains the limited impact of the presence of encryption on perceptions of security.

Identifying Encryption Given that encryption is an important security mechanism, we asked participants to detail how they identify the presence of encryption in a messaging app. Most participants report relying on textual information in the form of *disclosures* (40, 27.59%) or an app's *feature list* (11, 7.59%) mentioning encryption.

However, 42 participants (28.97%) said they would not know how to recognize encryption's presence. Very few mentioned visual indicators. For example, 5 (3.44%) mentioned observing a *delay during sending*, 3 (2.07%) mentioned messages disappearing, and 2 (1.38%) mentioned seeing messages be *scrambled*. These explanations are consistent with our regression analyses (cf. Table 4.2), highlighting the limited effect of visualizations.

> Key Insights: Validation.
>
> - Study 5 confirmed the findings of Studies 2–4.
> - Visualizing encryption in any way, even a simple text disclosure, improves perceptions compared to not mentioning it at all.
> - Most participants saw the animations and felt they communicated "security", yet this did not change their perceptions any more than a text disclosure did.
> - An app's reputation greatly impacts perceptions.

4.8 Discussion

In our first of five studies, we investigated why participants use particular messaging tools, validating a prior finding [3] that many users mistakenly think SMS and email are more secure than end-to-end encrypted messengers. Based on these initial findings, we aimed to improve the visibility of end-to-end encryption in a messaging app. Across the four subsequent studies, we compared six different text disclosures, three different icons, and three different animations of the encryption process.

Impact of Encryption Visualization While investigating the impact of different visualizations of encryption, we were surprised to find that the simple "encrypt" disclosure outperformed most others (aside from the progress circle) in terms of perceived trust, security, and privacy. As expected, however, all disclosures performed better than the baseline of having no disclosure at all. We were also surprised to see that security icons had a negative effect, rather than increasing perceptions of trust, security, and privacy. This negative effect was particularly distinct for people with a CS background.

Previous work suggested that encryption visualizations might positively impact perceived trust, security, and privacy [38, 49, 116]. Those suggestions were based primarily on qualitative data. Our studies, which combined quantitative and qualitative data, reached somewhat different conclusions. Based only on the qualitative data we collected, one might have reached conclusions similar to those of prior work. For example, as reported in Section 4.7, nearly half of participants indicated that the animations of encryption indicated an increased level of security. In contrast, our quantitative analyses indicated that these different animations did not have a significantly different impact on perceptions of the trust, security, and privacy of end-to-end encrypted messaging tools than a straightforward text disclosure that the conversation in encrypted, which is what many secure messaging apps currently display. These findings call into question the magnitude and applicability of the effects reported in prior work.

Our findings suggest that highlighting the use of encryption in basic ways (e.g., "*Messages to this chat are now encrypted*") significantly increases perceived security, privacy and trust in messaging applications. That is, having *any* visualization of encryption outperformed the control in our validation study of not calling attention to the use of encryption at all. However, richer visualizations of encryption involving icons or animations seem to have only a limited additional effect. Although we did not observe these richer visualizations of encryption to significantly impact user perceptions and satisfaction in a positive direction compared to basic text disclosures, we also did not observe a negative effect.

4.9 Recommendations

Given the promise of rich visualizations of encryption reported in prior work, this finding is disappointing, as it suggests that simple modifications of messaging apps' UIs are unlikely to help users better assess apps' security and privacy. Despite the use of multiple design proposals from previous work, we could not find a significant improvement (Section 4.8).

Our qualitative results imply that instead of investing more effort into studying richer visualizations of encryption, focusing on the following aspects is potentially more promising. We make recommendations for both providers of end-to-end encrypted communication tools and usable security researchers.

4.9.1 Tool Providers

Trust in Company As we have seen in the qualitative answers in the tool usage and validation study (Section 4.5.2, 4.7), participants report that they trust the brand and that the company would keep their data secure. Tool providers could focus on generally improving trust in the brand.

Convenience Several participants mentioned using a specific app to communicate with their peer groups that decided on that app (Section 4.5.2). Introducing an app or feature that is not compatible with their peer groups leads to them switching back to another channel. Tool providers should make sure that end-to-end encryption features do not lead to inconveniences for their users.

Functionality Our participants also mentioned that they switched the tools when a messenger did not support a required feature, for example with large attachments that they send via mail (Section 4.5.2). That indicates that a full feature set is required to avoid people switching to insecure channels. Making end-to-end encryption available in communication tools should not limit existing functionality.

4.9.2 Usable Security Research

Correcting Mental Models Our participants showed a number of incorrect mental models, most strikingly: Around 25% of our participants assumed that their conversations are free of eavesdroppers if the user interface shows only the names of their intended communication partner(s) (Section 4.5.2) and show a lack of understanding of man-in-the-middle attacker capabilities. Also, 14.29% of participants falsely assumed channels that are generally not encrypted by default (e.g., SMS) to be encrypted (Section 4.5.2). These misconceptions likely impact the usage of secure and private messengers significantly. Addressing them better should be a major goal for our community.

Technical Background As seen in our regression for Study 3 (Table 4.2), participants with a technical background tended to rate trust in security indicators lower. Investigating factors that contribute to this perception and provide improvements for these factors (e.g., increase company transparency) could help address concerns unique to that demographic.

4.10 Conclusion

We studied whether making a messaging app's end-to-end encryption more visible improves perceptions of trust, security, and privacy. To that end, we conducted five online studies with 683 total participants, including a summative validation study.

While participants felt most secure and private within the "encrypted" text disclosure condition, the different text disclosures did not have a significant impact on usability and app satisfaction. We observed a surprising negative effect of security icons on perceived trust, security, and privacy. When focusing on animations, none of the factors was statistically significant, though we identified a weak positive effect for the progress circle animation on perceived trust, security, and privacy. The animations had no impact on usability and satisfaction. We confirmed these key findings in a final summative study, validating that visualizing encryption in any way, even a simple text disclosure, improves perceptions compared to not mentioning encryption at all. Most participants saw the animations, the richest and most novel aspect of our investigation, and reported qualitatively they communicated "security." However, quantitative perceptions of trust, security, and privacy did not differ significantly compared to a text disclosure.

In our first study, we replicated the finding of prior work that a non-trivial fraction of users mistakenly believes SMS and email to be more secure than end-to-end encrypted messengers. While we had hypothesized that richly visualizing the process of encryption would emphasize end-to-end encrypted messaging apps' security properties and combat this misconception, our results suggest that the existing practice of disclosing the use of encryption in a straightforward text disclosure may be sufficient if the text disclosure is displayed prominently.

Chapter 5

Conclusions and Future Work

This part of the dissertation builds on, and is partially taken verbatim from, conclusions and future work ideas from the previously published papers that contributed to this dissertation, namely Chapters 2, 3, and 4. As described in those chapters, this research was collaborative in nature, and this conclusion therefore uses the academic "we".

Throughout this dissertation, we have demonstrated that human factors are a major factor in secure use and implementation of cryptography. First, in collaborative work, we showed that documentation and API usability impact developers' ability to write secure code [6, 7, 9]. We demonstrated this by conducting laboratory and large scale online programming experiments with developers.

5.1 Lessons Learned from Using an Online Platform to Conduct Large-Scale, Online Controlled Security Experiments with Software Developers

Security studies, where developers are asked to solve a programming problem, are an important research method. Developer Observatory, which we developed and designed as an distributed online platform for administering security programming studies, helps to conduct studies with developers remotely. This is particularly useful in a pandemic like COVID-19, when laboratory studies cannot be conducted in-person. We were able to conduct studies with Developer Observatory and recruit professional and hobbyist developers from all over the world, something that would not have been possible in a laboratory study. As a result, we could increase the power and generalizability of our studies, with little to no loss of internal validity.

We designed Developer Observatory with extensibility in mind to allow adaptations for future security experiments. Our experiences during the studies conducted with Developer Observatory suggest that the experience was satisfactory for the users of the tool as well as the researchers. The tool is published as open-source and can be freely used by researchers all over the world. We hope that it aids further research and simplifies recruiting of developers while at the same time lessens the inconvenience of study participation for developers. As of October 2021, other researchers have used Developer Observatory [67, 163], and are contributing to it on GitHub [1].

Based on the experiences with experiments conducted with Developer Observatory it makes sense to expand and improve Developer Observatory for future research. A potential expansion for researchers could be a web-based configuration and management tool to set up study parameters, monitor participants in progress, and retrieve data from the study and surveys, from one graphical interface. Another feature could be to collect participants' behavioral patterns while they are writing code while also allowing participants to switch between single tasks and survey questions as often as they wish. Furthermore, the session handling could be improved to allow participants to pause and resume their sessions so that they are able to close their browser and return at a later point without having their instance killed by the VM manager. Another optional feature could be automatic invite handling to invite participants based on the available resources as well as reinviting participants if they had been rejected because of lacking resources. Debugging tools for Jupyter could also be introduced via ipdb[2] to give developers an option to debug their code.

In conclusion, we have demonstrated with Developer Observatory that it is possible to conduct online studies with developers at scale while keeping a high external validity of research results. The studies conducted with Developer Observatory so far gave meaningful insights into challenges faced by developers during the implementation of cryptographic programming tasks.

Based on the idea of Developer Observatory, we are currently working in our ongoing research on an expanded tool that allows for even more complex studies with developers and users by providing them a full desktop environment through their webbrowsers. That way, it is possible for researchers to provide participants a complete desktop environment with an IDE preinstalled.

Next, we explored the important use case of encryption for end-users. To understand for which purposes and for what end-users use encryption, we analyzed a university-wide data set of emails, which allowed us to measure the adoption of encrypted emails, as well as to investigate potential challenges.

5.2 27 Years and 81 Million Opportunities Later: Investigating the Use of Email Encryption for an Entire University

In our analysis of 81,612,595 emails, we observed that an unsurprisingly low fraction of emails are encrypted: 0.06% were encrypted and 2.8% were signed. While, over time, the total volume of emails increased, the fraction of encrypted emails decreased. Encryption peaked in 2006 with only 3.72% of all emails being signed, and 0.35% encrypted, and has remained around 0.06% for the last decade. Especially

[1] `https://github.com/developer-observatory/developer-observatory/network/members`, last accessed October 2021

[2] `https://pypi.python.org/pypi/ipdb`

since our university offers free and easy access to S/MIME certificates, we think that our results may even be seen as an upper bound for end-to-end encrypted email in the worldwide ecosystem.

Initially, PGP was the prominent choice for signing emails, which changed when our university started issuing free S/MIME certificates to students and employees. Since then, S/MIME has overtaken PGP as the dominant choice for email signatures; this trend continues as of July 2021. For encrypting emails, however, PGP started out more popular than S/MIME, and has only slightly fallen behind. We observed that the algorithms used align with the current default settings of email clients and tools, which we see as an indication that most users do not deviate from default settings, especially for generated keys and signature algorithms. We found that S/MIME certificates were usually generated with the tools that our university's CA provided, using the same settings for all users. The majority of these use RSA with 2048 bit, while elliptic curve cryptography was almost non existent. For PGP, which is mostly used with open source software, algorithms were constantly changed, with new algorithms introduced, which lead to variations in the observed PGP keys.

After Leibniz University Hannover started issuing S/MIME certificates in 2004, S/MIME became much more relevant in our data set, despite the generation of S/MIME certificates being more complex than PGP keys. Around a third of the S/MIME certificates in our data set were issued by the DFN with additional hurdles like identity checks. Future work might explore why users are willing to fulfill these requirements as opposed to generating a simple secret key.

Generally, users sign more emails than they encrypt. This can partially be explained by the recipient: either they do not use a compatible encryption tool, or their public key is unknown to the sender. This is indicative of a key distribution problem. In addition, our data suggests that even if sender and recipient use encryption, they may not encrypt their email exchange, and even after receiving encrypted email, users do not start to use encryption on a regular basis.

Another reason for not encrypting emails can be usage of multiple devices. We saw that the rate of signed emails drops when users add a second device. Being unable to sign emails on a device might mean that the device does not have their private key, which might mean they are also unable to decrypt incoming messages, as this would usually require the same certificate or key (possibly with different subkeys). For recipients, receiving signed and unsigned emails from the same sender may be confusing, as signing emails can then no longer be taken as a reliable indicator of legit emails. Mail client developers should consider how to sync keys across multiple devices.

The amount of emails in general has increased every year, with a peak in 2020 due to the COVID-19 pandemic and work-at-home orders. We could also observe a large increase of S/MIME signed mails, most likely due to mailinglist emails starting to get signed, which resulted in a large amount of new S/MIME emails. However, at the same time, the percentage of S/MIME-encrypted emails has decreased, which might indicate that people had trouble using their keys from home. Furthermore, the amount of issued S/MIME certificates has decreased by a third (cf. Figure B.3). This might be a result of people delaying the personal identification required to receive a new certificate.

We saw that, for a large German university, email encryption adoption is extremely low. We showed several potential usability problems that may have to do with low adoption rates, such as problems with key management, key rollover and key exchange. While emails are not end-to-end encrypted out of the box, and require manual effort to encrypt, modern messengers use end-to-end encryption by default. If users communicate sensitive information through these encrypted channels, their information is safe in transit, which is currently not true for most email. Therefore, in the final study of this dissertation, we investigated users' understanding (and notice) of end-to-end encryption in messengers.

5.3 On the Limited Impact of Visualizing Encryption: Perceptions of End-to-End Messaging Security

To assess whether making a messaging app's end-to-end encryption more visible can help to improve perceptions of trust, security, and privacy, we conducted a series of online studies, including a summative validation study, with 683 total participants. We found that participants reported the highest security and privacy perceptions when presented with text disclosures; differences in the text disclosures does not matter for usability or app satisfaction. We were surprised to find a negative effect of security icons on how participants rated their perception of trust, security, and privacy. For animations, none of our attempts was statistically significant, but we found a weak positive effect for the progress circle animation on perceived trust, security, and privacy. Animations did not impact usability and app satisfaction. In a summative validation study, we confirmed that visualizing encryption, even with simple text disclosures, improves perceptions, compared to not mentioning security at all. While most participants noticed animations, and correctly assessed that they communicate security, quantitative perceptions of trust, security, and privacy did not increase significantly over a simple text disclosure.

We were also able to replicate the finding that some users think that "older" communication methods, such as SMS and email, are more secure than end-to-end encrypted messengers. We had initially hypothesized that a rich visualization for the process of encryption would emphasize the security properties of encrypted messaging apps, decreasing this misconception. Our results show that a prominent positioning of existing plain text disclosures may suffice.

5.4 Outlook

In conclusion, we think that, in order to improve the adoption of cryptography across developers and users, encrypting by default is promising. However, end-to-end encrypting by default only offers full protection, if all communication and all metadata are encrypted across protocols. For that, we need usable cryptography support for developers, who will need to work together with usability experts in order to make their solutions accessible and comprehensible to a broad audience to drive adoption. We recommend further research to better integrate knowledge from cryptography, software engineering, and user experience. Improving the comprehension and adoption of cryptography by programmers and end users through better usability is a challenging but worthy task, and has the potential to redefine privacy and security for society.

Appendix A

Appendix: Lessons Learned from Using an Online Platform to Conduct Large-Scale, Online Controlled Security Experiments with Software Developers

A.1 Example Notebook

```
{
 "cells": [
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "## Header for the task\n",
    "__Goal__: Some goal."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 0,
   "metadata": {
    "collapsed": false,
    "tasknum": "1"
   },
   "outputs": [],
   "source": [
    "# This is where your code goes"
   ]
  },
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 2",
   "language": "python",
   "name": "python2"
  },
  "language_info": {
    "codemirror_mode": {
     "name": "ipython",
     "version": 2
    },
    "file_extension": ".py",
    "mimetype": "text/x-python",
    "name": "python",
    "nbconvert_exporter": "python",
    "pygments_lexer": "ipython2",
    "version": "2.7.9"
   }
  },
  "nbformat": 4,
  "nbformat_minor": 0
}
```

LISTING A.1: An example ipynb task file. Researchers can provide task descriptions and code stubs.

Appendix B

Appendix: 27 Years and 81 Million Opportunities Later: Investigating the Use of Email Encryption for an Entire University

B.1 S/MIME and PGP Parsing Errors

During the parsing of the emails, we encountered parsing errors for 0.09% of all emails.

While the error rate for S/MIME is consistently low (cf. Figure B.2), the error rate for PGP is higher (cf. Figure B.1). Overall, we encountered parsing errors for up to 1.17% of the S/MIME and 22.09% of the PGP emails per year. While we cannot provide in-depth insights into the parsing errors due to ethics and data protection concerns, Figures B.1 and B.2 show that most of the parsing errors for both S/MIME and PGP were caused by the five most active users of the respective years. For example, we saw most parsing errors for S/MIME in 2005 and PGP in 2004 where our data set included 1,367 S/MIME emails and 47 users and 1,521 PGP emails and 129 users. In this year, the three most active S/MIME users contributed all parsing errors for S/MIME and the five most active PGP users contributed 306 parsing errors (88.26% of all PGP parsing errors). Even more extreme is the peak for PGP email related parsing errors in 2004 which were mostly caused by a single KMail [1] user.

While parsing errors in general are disappointing since they add noise to our results, the fact that the vast majority of errors was caused by a small group of or even single users alleviates their negative effects. Our evaluation focuses on a per-user and not a per-email view. A small number of users whose emails could not be parsed, for example, only has a low impact on the overall data concerning the use of multiple email clients.

[1] cf. `https://apps.kde.org/kmail2/`

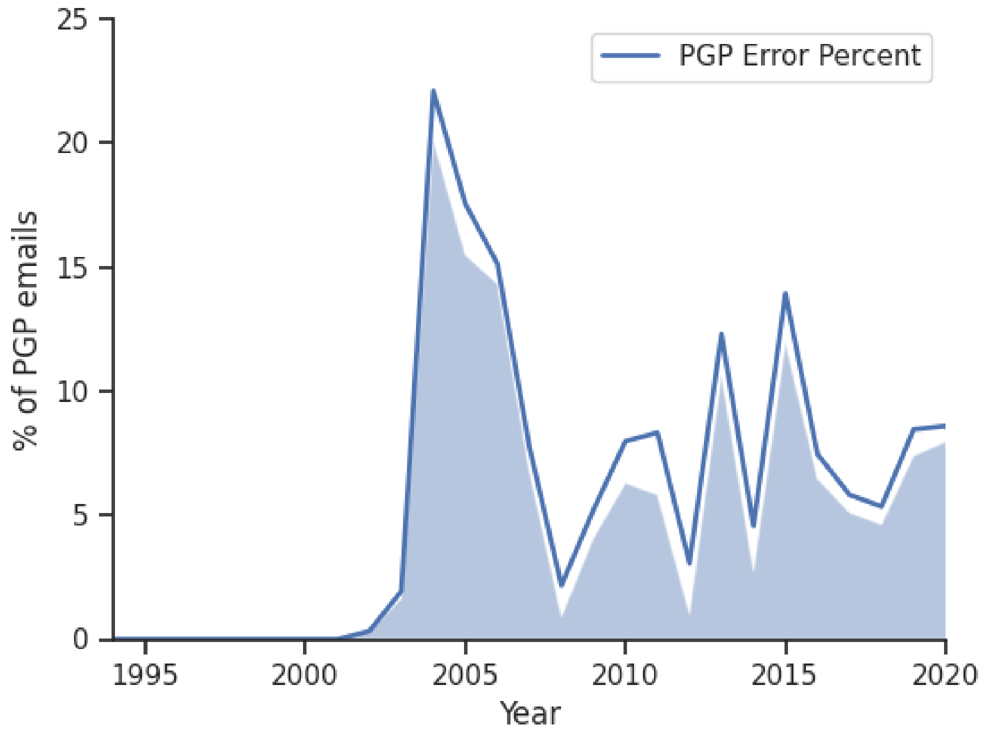


FIGURE B.1: PGP parsing errors over the years. The marked area is the parsing errors of the top 5 users of the year.

B.2 Pseudonymization Table

Table B.1 illustrates the pseudonymization we applied to the data.

B.2.1 Certificates issued for our university

Figure B.3 shows the certificates that were issued by the DFN PKI in each year up until February 19, 2021.

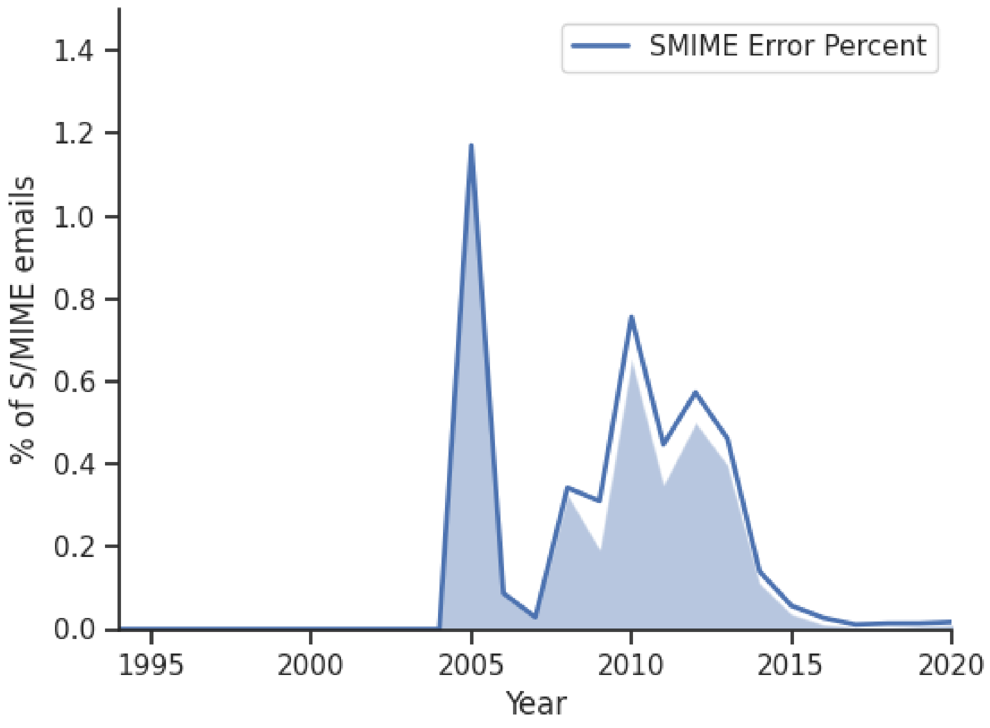


FIGURE B.2: S/MIME parsing errors over the years. The marked area is the parsing errors of the top 5 users of the year.

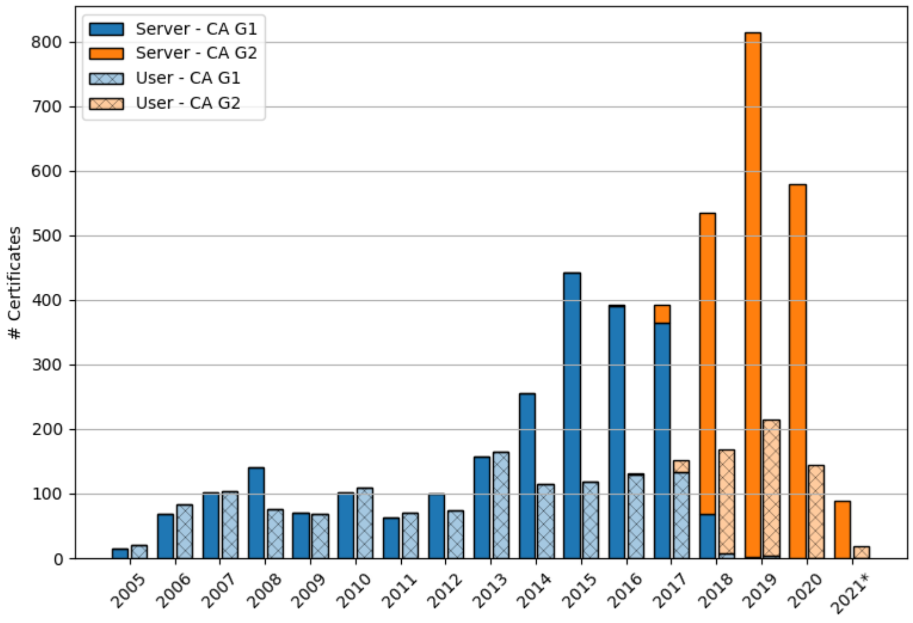


FIGURE B.3: X.509 certificates issued by the DFN PKI for our university from 2005 to 2020 using two different root certificates G1 and G2. G1 expired in 2019. Includes certificates issued until February 19, 2021.

Header-Data	**Format**
Message ID	SHA-256 with salt
User	SHA-256 with salt
User group	Categorized
Sender	SHA-256 with salt
Receivers	SHA-256 with salt
CC list	SHA-256 with salt
BCC list	SHA-256 with salt
Date	Bracketed into week
Client	Raw value
Folder	Categorized
S/MIME-Data	**Format**
Serial Number	SHA-256 with salt[1]
Not Valid Before and After	Bracketed into weeks[1]
Issuer Info	Raw value
Signature Algo	Raw value
Key size	Raw value
Key type	Raw value
PGP-Data[2]	**Format**
KeyID	SHA-256 with salt
Creation date	Bracketed into week
Expiration date	Bracketed into week
Type of key	Raw value
Length	Raw value
Key Algo	Raw value
Digest Algo	Raw value

[1] Applied to the leaf certificate

[2] Applied to both primary and sub key info

TABLE B.1: Collected metadata for all emails including the storage format. For data protection reasons and in consultation with our university's data protection officer and the IT staff, we chose to pseudonymize some values using a hash function or categories.

Appendix C

Appendix: On the Limited Impact of Visualizing Encryption: Perceptions of End-to-End Messaging Security

C.1 Demographics

Table C.1 shows the demographics of participants in all studies.

C.2 Scale of Perceived Trust, Security and Privacy

Ten item scale of perceived trust, security and privacy. Participants choose from a 5-point likert scale on each question.

1. I think the new WhatsApp version is trustworthy.
2. I do not doubt the honesty of the new WhatsApp version.
3. I think the new WhatsApp version is secure.
4. I think only me and the recipient(s) can read our messages.
5. I think other people cannot send a message pretending to be me.
6. I think no one can unnoticeable modify messages sent between me and the recipient(s).
7. I think that if somebody hacks my phone, they will not be able to read my messages.
8. I think only me and the recipient(s) can know the messages were sent.
9. I think the new WhatsApp version does not collect more personal information than strictly needed.
10. I think the new WhatsApp version will not use my personal information for other purposes without my authorization.

	Study: Tool Usage	**Study: Disclosures**	**Study: Icons**	**Study: Animations**	**Study: Validation**
Participants					
Started	173	234	100	253	159
Finished	160	210	90	159	150
Valid ($n =$)	149	196	86	107	145
Gender					
Male	60.7%	54.1%	47.7%	67.3%	40.7%
Female	37.9%	44.9%	51.2%	29.0%	57.9%
Not M/F	1.4%	1.0%	1.2%	2.8%	1.4%
Ethnicity†					
White	78.5%	68.9%	68.6%	71.0%	89.0%
Asian or Pacific Islander	6.0%	14.3%	16.3%	7.5%	6.9%
Black or African American	13.4%	7.7%	11.6%	14.0%	0.0%
Hispanic or Latino	4.7%	12.2%	11.6%	10.3%	2.8%
Native American	0.7%	1.5%	0.0%	0.0%	0.0%
Other & Prefer not to say	0.7%	0.5%	1.2%	0.9%	4.1%
Smartphone OS†					
Android	67.1%	57.7%	41.9%	60.7%	59.3%
iOS	37.6%	50.5%	66.3%	39.3%	40.7%
Other	0.0%	1.0%	3.5%	0.0%	0.0%
No smartphone	1.3%	0.0%	0.0%	0.0%	0.0%
Prefer not to say	0.7%	0.0%	1.2%	0.0%	0.0%
Computer Science					
CS Education	28.9%	24.5%	24.4%	31.8%	26.9%
CS Job	22.1%	26.0%	29.1%	32.7%	19.3%
Age in years					
Mean	37.5	33.9	35.6	32.5	37.7
Std. dev. (σ)	10.7	9.0	10.9	8.4	10.5
Median	35.0	33.0	33.0	31.0	35.0

† Multiple answers allowed, may not sum to 100%

TABLE C.1: Participant demographics.

C.3 Messenger Usage

The following figures show the messenger usage among our participants in the first survey. Figure C.1 shows the preferred messenger for day-to-day conversations, Figure C.2 shows the preferred messenger for sensitive or confidential conversations. Figure C.3 shows all messengers used in the last 6 months.

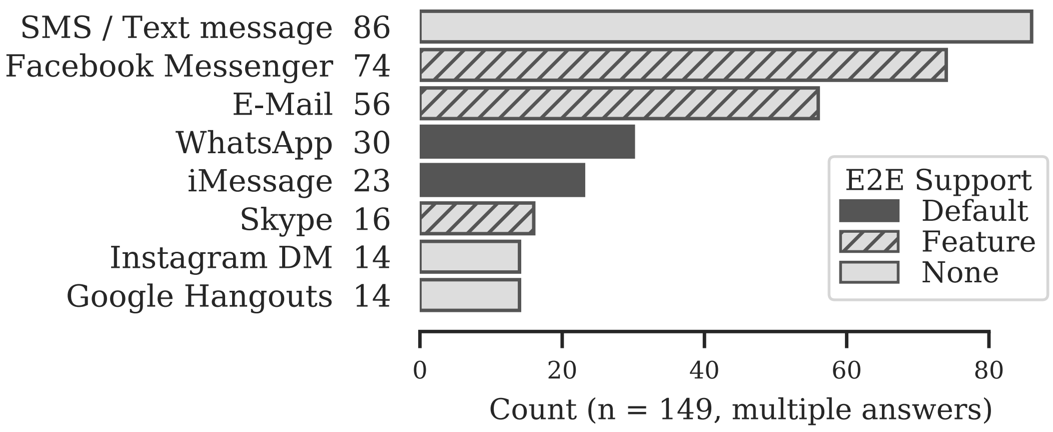


FIGURE C.1: Study: Tool Usage - "Which tools do you prefer for day-to-day conversations?" (Top 8)

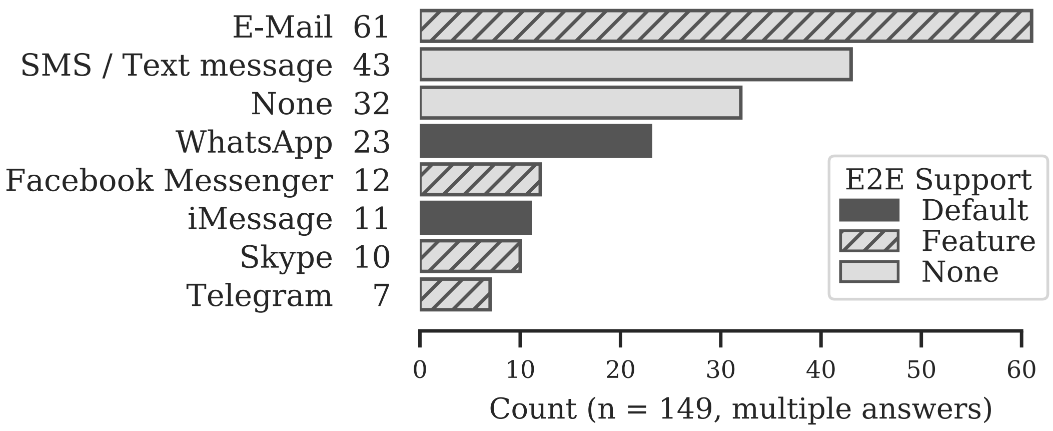


FIGURE C.2: Study: Tool Usage - "Which tools do you prefer for sensitive or confidential conversations?" (Top 8)

C.3.1 Scripted Conversation

The text in Figure C.4 was used in the scripted conversations that were shown to the participants in the videos. The overall screencast took on average 95 seconds.

C.3.2 Data Quality

Participant diversity and data quality on MTurk and Prolific is generally perceived as satisfactory [98, 99, 100]. We followed best practices [75, 100, 155] and required workers to be U.S. residents who have already completed 100+ HITs with a 95% approval rate.

During piloting, we experienced similar data quality issues as reported in recent work [75]. Therefore, we implemented a set of countermeasures, including blocking participants whose IP address came from outside the U.S. or belonged to a VPN or proxy service provider even within the U.S.[1]

[1]We used the `https://iphub.info` service to filter VPNs and proxies.

Name	**End-to-End Encrypted**	**End-to-End Indicator**			**Platforms**		**Downloads**
(Alphabetical order)	(Protocol Name)	Color	Icon	Text	Android	iOS	(On Android)
Email	○	-	-	-	●	●	-
Email with PGP or S/MIME	● (PGP or S/MIME)	d	d	d	●	●	-
Facebook Messenger	◐ (Signal)	●	Lock	●[1]	●	●	5.000M+
FaceTime	● (SRTP)	○	○	○	○	●	-
Google Hangouts	○	-	-	-	●	●	5.000M+
iMessage	● (unknown)	●	○	○	○	●	-
Instagram DM	○	-	-	-	●	●	1.000M+
Kik Messenger	○	-	-	-	●	●	100M+
LinkedIn InMail	○	-	-	-	●	●	500M+
Signal	● (Signal)	○	Lock	○	●	●	50M+
Skype	◐ (Signal)	○	○	●[2]	●	●	1.000M+
SMS	○	-	-	-	●	●	-
Snapchat	○	-	-	-	●	●	1.000M+
Telegram	◐ (MTProto2.0)	●	Lock	●[3]	●	●	500M+
Twitter DM	○	-	-	-	●	●	1.000M+
Viber	● (unknown)	●	Shield[4]	●[5]	●	●	500M+
WhatsApp	● (Signal)	○	Lock	●[6]	●	●	5.000M+

● Yes (for end-to-end encrypted: Yes, by default)
○ No
◐ Has a "secret mode" which uses end-to-end encryption, but is not active by default
d Depends on the client used
1 *Secret conversation*
2 *Private conversation*
3 *Secret chat*
4 *Has an additional Secret Chat, uses a lock icon*
5 *Messages sent in this conversation are encrypted*
6 *Messages to this chat and calls are now secured with end-to-end encryption*

TABLE C.2: List of popular communication tools.

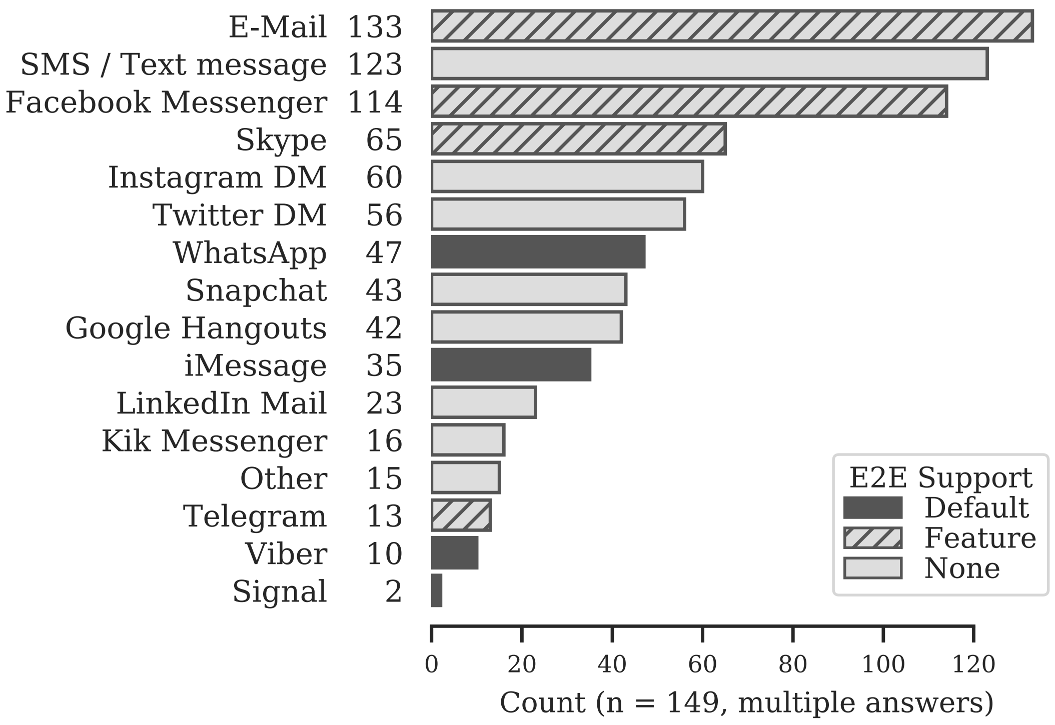


FIGURE C.3: Study: Tool Usage - "Which online communication tools have you used in the last 6 months?"

Me: Hi, darling. I forgot my wallet at home. Could you please look up my credit card number? I need to place an order before I forget.

<Wait 10s>

Remote: Sure, where is it?

Me: It should be on my desk.

<Wait 5s>

Remote: One second.

<Wait 25s>

Remote: It's 1234-5678-9012-3456, valid until 12/21, and the security code is 456.

Me: Thanks

FIGURE C.4: Chat messages displayed in the application.

Following best practices [73, 75], we added three attention checks to all questionnaires.

To remove a potential confound for Studies 2–5, we wanted to include only participants familiar with secure messaging apps. Therefore, we added an MTurk qualification task in which we asked participants which messaging apps they currently used and invited only WhatsApp users to the actual study itself. Workers who had participated in one study were ineligible for all subsequent ones. We paid each participant $0.15 for the short qualification task.

We took advantage of the pre-screening provided by Prolific, where participants had to report the regular use of WhatsApp in a pre-screening questionnaire[2], and required participants to be located in the U.S. or UK, have a 95% or higher approval rate and at least 100 previous submissions.

C.4 Study Video Introduction

We introduced the video to our participants in the following way:

> *"Imagine that WhatsApp will soon release a new version. This new version would have a different user interface in some places but have the same features as the version you are used to. Below we show you a brief video of what this new user interface for WhatsApp might look like."*

C.5 Replication Material

The videos used within this paper are available on our webpage at `https://publications.teamusec.de/2021-soups-e2e/`.

C.6 Survey Questions

C.6.1 Tool Usage Survey (Study 1)

The survey was described to participants in the following way before the consent form:

> *"In the following survey, you will be asked a series of questions about your communication tools (e.g. E-Mail, WhatsApp, SMS) habits. This survey should take approximately 20 minutes to complete. We encourage you to complete it in one sitting. To start the survey, please click on the "next" button below.*
>
> *Thank you for participating in this survey, we appreciate your time and effort!"*

C.6.1.1 General Use of Communication Tools

Q1: Which online communication tools have you used in the last 6 months? *Note: For all answer options with messenger names: All messengers had their icon displayed next to them in order for participants to recognize them easier.*

- Facebook Messenger

[2]The exact question we asked in the pre-screening questionnaire was: Which of the following chat apps do you use regularly? [multiple-choice]

- Snapchat
- WhatsApp
- Google Hangouts
- Twitter DM
- Instagram DM
- LinkedIn Mail
- Kik Messenger
- Skype
- Signal
- Telegram
- iMessage
- Viber
- E-Mail
- SMS / Text message
- Other 1-4
- Prefer not to say

C.6.1.2 Preferences - Day-to-Day

Q2: Which tools do you prefer for day-to-day conversations?

- Facebook Messenger
- Snapchat
- WhatsApp
- Google Hangouts
- Twitter DM
- Instagram DM
- LinkedIn Mail
- Kik Messenger
- Skype
- Signal
- Telegram
- iMessage
- Viber
- E-Mail

- SMS / Text message
- Other 1-4
- Prefer not to say

Q3: Please explain why you prefer <messenger> for day-to-day conversations. *Note: This question was repeated for every messenger that was selected in Q2.*

C.6.1.3 Preferences - Tool Selection - Sensitive

Q4: Which tools do you prefer for sensitive or confidential conversations? *You can assume conversations to be sensitive or confidential whenever you care about protecting them against a person or company that is not involved in the conversation. Example conversations might cover (but are not limited to) your Social Security number, personal health information, bank account details or credit card numbers.*

- Facebook Messenger
- Snapchat
- WhatsApp
- Google Hangouts
- Twitter DM
- Instagram DM
- LinkedIn Mail
- Kik Messenger
- Skype
- Signal
- Telegram
- iMessage
- Viber
- E-Mail
- SMS / Text message
- Other 1-4
- None
- I don't care
- Prefer not to say

Q5a: Please explain why you prefer <messenger> for sensitive or confidential conversations. *Note: This question was repeated for every messenger that was selected in Q4.*

Q5b: Is there a way that you can tell if conversations in <messenger> are confidential? If yes, how? *Note: This question was repeated for every messenger that was selected in Q4.*
Q5c: Please explain why you have chosen None. *Note: This question was only shown if "None" was selected in Q4.*

C.6.1.4 Preferences - Tool Selection - Security

Q6: Regardless of whether you use the tool or not, which tools do you expect provide the best security for your conversations? *Note: This was a Matrix question, with the following options for each element: I think this is secure, I think this is insecure, I don't know this tool*

- Facebook Messenger
- Snapchat
- WhatsApp
- Google Hangouts
- Twitter DM
- Instagram DM
- LinkedIn Mail
- Kik Messenger
- Skype
- Signal
- Telegram
- iMessage
- Viber
- E-Mail
- SMS / Text message
- Other 1-4

For every that was selected secure:
Q7a: Please explain why you expect <messenger> is a secure choice. *Note: This question was repeated for every messenger where the participant selected "I think this is secure" in Q6.*
Q7b: Is there a way that you can tell if conversations in <messenger> are secure? If yes, how? *Note: This question was repeated for every messenger where the participant selected "I think this is secure" in Q6.*

C.6.1.5 Access

Q8a-Q8e: How likely or unlikely is it that the following groups of people can access the messages you send over [your personal E-Mail account (Gmail, Yahoo, ...), Facebook Messenger, WhatsApp, Snapchat, SMS / Text message]? *Note: Each answer had 5-point likert+I don't know as answer options (Extremely unlikely - Extremely likely + I don't know)*

- People who live in my household.
- My employer.
- My internet or mobile phone provider.
- The company that makes the app or the tool's software.
- Cybercriminals (e.g. hackers or organized crime).
- Law enforcement or intelligence agencies (e.g. Police, FBI, NSA or CIA).
- Online advertising companies.
- The manufacturer of your phone.

C.6.1.6 Secret/Private Mode of Messengers

Note: The following questions were only asked, if one of the messengers was selected as used.
Q9: Have you ever used the "Secret conversation" feature in the Facebook Messenger mobile app? Q10: Have you ever used the "private conversation" feature in the Skype app?

C.6.1.7 SEBIS

Note: This is the standardized Security Behaviors Intentions Scale (SEBIS) and was directly taken from their paper [42].
Q11: People take all sorts of actions online every day. Please read the following statements about everyday situations and estimate how often you agree with the following statements.

Each of the statements had the answer options:

- Never
- Rarely
- Sometimes
- Often
- Always

The statements were:

- I set my computer screen to automatically lock if I don't use it for a prolonged period of time.

- I use a password/passcode to unlock my laptop or tablet.
- I manually lock my computer screen when I step away from it.
- I use a PIN or passcode to unlock my mobile phone.
- I do not change my passwords, unless I have to.
- I use different passwords for different accounts that I have.
- When I create a new online account, I try to use a password that goes beyond the site's minimum requirements.
- I do not include special characters in my password if it's not required.
- When someone sends me a link, I open it without first verifying where it goes.
- I know what website I'm visiting based on its look and feel, rather than by looking at the URL bar.
- I submit information to websites without first verifying that it will be sent securely (e.g., SSL, "https://", a lock icon).
- When browsing websites, I mouseover links to see where they go, before clicking them.
- If I discover a security problem, I continue what I was doing because I assume someone else will fix it.
- When I'm prompted about a software update, I install it right away.
- I try to make sure that the programs I use are up-to-date.
- I verify that my anti-virus software has been regularly updating itself.

C.6.1.8 Demographics

Q12: Do you have formal education (Bachelor's degree or higher) in computer science, information technology, or a related field?

- Yes
- No
- Prefer not to say

Q13: Have you held a job in computer science, information technology, or a related field?

- Yes
- No
- Prefer not to say

Q14: What is your gender? ***If you prefer not to say, please fill in a "-".***
<Free Text>

Q15: What is your age in years?. ***If you prefer not to say, please fill in a "-".***
<Free Text>

Q16: What is your highest degree or level of school you have completed? If currently enrolled, select the highest degree received.

- High school education or less
- Some college, no degree
- Trade/technical/vocational training
- Associate's Degree
- Bachelor's Degree
- Graduate degree (master's, doctorate, or professional degree)
- Prefer not to say

Q17: Please specify your ethnicity. (Multiple answers possible)

- White
- Hispanic or Latino
- Black or African American
- Native American or American Indian
- Asian or Pacific Islander
- Prefer not to say
- Other

Q18: Which smartphones do you use?

- Android (e.g. Samsung Galaxy, Google Pixel)
- iOS (e.g. iPhone)
- Windows Phone (e.g. Nokia)
- Blackberry OS (e.g. Blackberry)
- No smartphone
- Other <Free Text>
- Don't know
- Prefer not to say

Q19: What is your current employment status? (Multiple answers possible)

- Employed for wages
- Self-employed

- Military
- Out of work and looking for work
- Out of work but not currently looking for work
- A student
- Retired
- Unable to work
- Prefer not to say
- Other

Q20: Do you have any final thoughts on this survey? Please tell us what you liked about this survey and where we could improve. You'll receive your verification code for MTurk on the next page.

<Free Text>

C.6.1.9 Final Page

Thank you for participating! Please post the following code into your MTurk HIT to verify that you have completed this survey.

<A random ID to verify the results and pay the user on MTurk>

C.6.2 Qualification Survey for Study 2-4

The survey was described to participants in the following way before the consent form:

> *"This is a qualification task with a follow up survey in the next days. In the following qualification task, we will ask you two questions about your online communication behaviour. Completion should take around 30 seconds.*
>
> *Thank you for your participation!"*

C.6.2.1 General Use of Communication Tools

Q1: Which online communication tools have you used in the last 6 months? *Note: All messengers had their icon displayed next to them in order for participants to recognize them easier.*

- Facebook Messenger
- Snapchat
- WhatsApp
- Google Hangouts
- Twitter DM
- Instagram DM

- LinkedIn Mail
- Kik Messenger
- Skype
- Signal
- Telegram
- iMessage
- FaceTime
- Viber
- E-Mail
- SMS / Text message
- Other
- Prefer not to say

Q2: Which operating system are you primarily using on your smartphone?

- iOS (e.g. iPhone)
- Android (e.g. Samsung Galaxy, Google Pixel)
- Windows Phone (e.g. Nokia Lumia)
- Blackberry OS (e.g. Blackberry)
- Don’t know
- Prefer not to say
- No smartphone
- Other

Survey End Thank you for participating! Please post the following code into your MTurk HIT to verify that you have completed this qualification task.

<A random ID to verify the results and pay the user on MTurk>

C.6.3 Disclosures, Icons, and Animation Surveys (Study 2-4)

The survey was described to participants in the following way before the consent form:

> *"In the following survey, you will be asked a series of questions about how you use various communication tools (e.g. E-Mail, WhatsApp, SMS).*
>
> *To participate in this survey you need to be a* ***WhatsApp user on Android or iOS****.*

The survey should take approximately 10 minutes to complete. We encourage you to complete it in one sitting. To start the survey, please click on the "next" button below.

Thank you for participating in this survey, we appreciate your time and effort!"

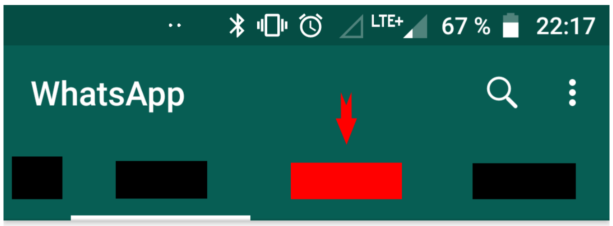
C.6.3.1 Verification

Note: To ensure participants were actual users of WhatsApp on either iOS or WhatsApp we asked them a simple question that could be answered by opening the app once on the phone. If the question was answered wrong, the survey would end and participants asked to return the survey on MTurk, as they were not eligible.

Q1: Which operating system are you using on your smartphone?
Note: This question was used to determine if Q2a or Q2b should be displayed. If Other was selected the survey would end.

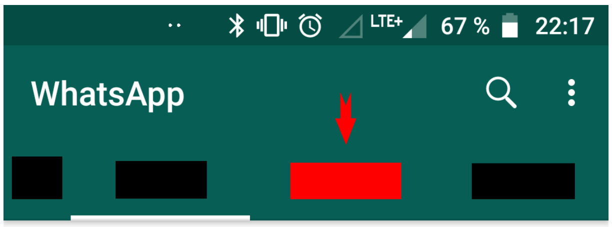
- Android
- iOS
- Other

Q2a: Please open the WhatsApp application on your phone to answer the following question: What is the missing text indicated by the red arrow?
Note: This question was displayed to Android users.



- Camera
- Chat
- Status
- Calls
- Settings
- Favorites

Q2b: Please open the WhatsApp application on your phone to answer the following question: What is the missing text indicated by the red arrow?
Note: This question was displayed to iOS users.



[redacted] [redacted] [redacted] [redacted]

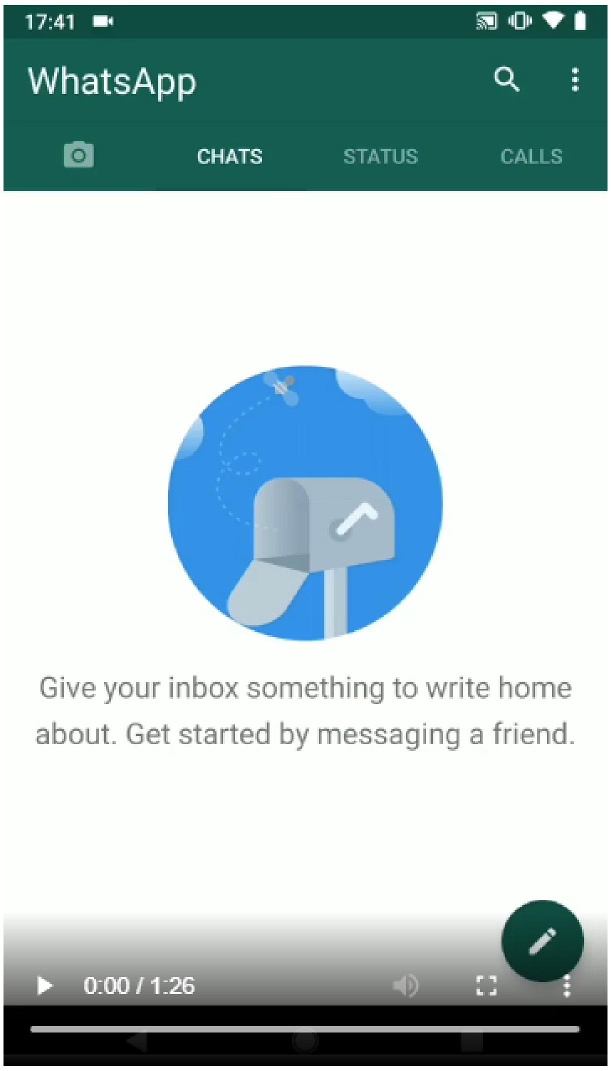
- Status
- Calls
- Camera
- Chats
- Settings
- Favorites

C.6.3.2 Attention Check 1

Q3: Please enter your year of birth.
<Free Text>

C.6.3.3 Conditions

Note: This subsection is where the different conditions were applied in the survey. The only difference is the video that was displayed to the participants. The video element was large enough that participants could see everything, even if they did not use fullscreen mode. The next button to continue with the survey was only enabled when the end of the video was reached.
Imagine that WhatsApp will soon release a new version.
Below we show you a brief video of what a new user interface for WhatsApp might look like.
Please, pay attention to the video and watch it in fullscreen mode.



Give your inbox something to write home about. Get started by messaging a friend.

C.6.3.4 UMUX LITE

Q4: Please read the following statements and estimate how much you agree with them.
Note: Each statement had a 7-point likert scale from Strongly agree to Strongly disagree.

1. This system's capabilities meet my requirements.
2. This system is easy to use.

C.6.3.5 Questions for Screencast

Q5: Please read the following statements and select whether you agree with them or not. *Note: Each statement had a 7-point likert scale from Strongly agree to Strongly disagree.*

1. I think the new WhatsApp version is trustworthy.
2. I do not doubt the honesty of the new WhatsApp version.

3. I think the new WhatsApp version is secure.
4. I think only me and the recipient(s) can read our messages.
5. I think other people cannot send a message pretending to be me.
6. I think no one can unnoticeably modify messages sent between me and the recipient(s).
7. I think that if somebody hacks my phone, they won't be able to read my messages.
8. I think only me and the recipient(s) can know the messages were sent.
9. I think WhatsApp does not collect more personal information than strictly needed.
10. I think WhatsApp will not use my personal information for other purposes without my authorization.

Q6: Please read the following statement and rate how much you agree with it.
Note: Each statement had a 7-point likert scale from Strongly agree to Strongly disagree.

1. Using the new version of WhatsApp would make me feel secure.

C.6.3.6 Net Promoter Score

Q7: On a scale from 0-10, how likely are you to recommend the new WhatsApp version to a friend, family member or colleague?

C.6.3.7 Free Text Questions

Note: Each of these questions had a free text answer field and was displayed on a separate page. It was random if Q8 or Q9 was displayed first.
Q8a: In the video you watched, please describe what you observed happening (if anything) when the messages are sent.
Q8b: What did this indicate to you?
Q9a: In the video you watched, please describe what you observed happening (if anything) when the messages are received.
Q9b: What did this indicate to you?

C.6.3.8 General Use of Communication Tools

Q10: Which online communication tools have you used in the last 6 months?
Note: All messengers had their icon displayed next to them in order for participants to recognize them easier.

- Facebook Messenger

- Snapchat
- WhatsApp
- Google Hangouts
- Twitter DM
- Instagram DM
- LinkedIn Mail
- Kik Messenger
- Skype
- Signal
- Telegram
- iMessage
- FaceTime
- Viber
- E-Mail
- SMS / Text message
- Prefer not to say

Q11: Which smartphones do you use?

- Android (e.g. Samsung Galaxy, Google Pixel)
- iOS (e.g. iPhone)
- Windows Phone (e.g. Nokia)
- Blackberry OS (e.g. Blackberry)
- No smartphone
- Other
- Don't know
- Prefer not to say

C.6.3.9 Messenger Usage

Note: Order of Q12 and Q13 randomized.

Q12: Imagine all your contacts are using all the tools beneath: Which tools do you prefer for sensitive or confidential conversations?

You can assume conversations to be sensitive or confidential whenever you care about protecting them against a person or company that is not involved in the conversation. Example conversations might include (but are not limited to) your Social Security number, personal health information, bank account details or credit card numbers.

- Facebook Messenger
- WhatsApp (new version from the video)
- Twitter DM
- Instagram DM
- Skype
- E-Mail
- SMS / Text message
- iMessage
- None
- I don't care
- Prefer not to say

Q13: Imagine all your contacts are using all the tools beneath: Which tools do you prefer for everyday conversations? You can assume that everyday conversations do not contain confidential information.

- Facebook Messenger
- WhatsApp (new version from the video)
- Twitter DM
- Instagram DM
- Skype
- E-Mail
- SMS / Text message
- iMessage
- None
- I don't care
- Prefer not to say

C.6.3.10 Secret/Private Mode of Messengers

Note: Each of these questions was only displayed if the participants selected the appropriate messenger in Q10.

Q14: Have you ever used the "secret conversation" feature in the Facebook Messenger mobile app?

- Yes
- No
- I don't know

Q15: Have you ever used the "private conversation" feature in the Skype app?

- Yes
- No
- I don't know

Q16: Have you ever used the "secret chat" feature in the Telegram app?

- Yes
- No
- I don't know

C.6.3.11 Free Text Questions

Q17: Please describe in 2-3 sentences what could prevent you from using an updated WhatsApp version as we just showed you.
<Free Text>

C.6.3.12 Attention Check 2

Q18: Which messenger was displayed in the video?
<Free Text>

C.6.3.13 SEBIS Score

Note: This is the standardized Security Behaviors Intentions Scale (SEBIS) and was directly taken from their paper [42].
Q19: SEBIS *Please check C.6.1.7 for the full question+answer options.*

C.6.3.14 Extra Questions for Animations (Survey 4 only)

These questions were added to the Survey with Animations to get a better idea if animations would be noticed.
Q20: Please indicate your agreement to the following general statements: *The answer options were a 5-point likert from "Strongly agree" to "Strongly disagree" with a "I don't know" option.*

1. A ***trustworthy company*** having made a particular service, app, or device suggests to me that messages are being sent securely.
2. The ***length of time*** it takes to process a message when it is being sent or received suggests to me that messages are being sent securely.
3. A service, app, or device being ***advertised as "secure" by its creator*** suggests to me that messages are being sent securely.

4. Seeing messages being ***turned into secret code*** suggests to me that messages are being sent securely.
5. Reading ***news reports*** that indicate a service, app, or device is "secure" suggests to me that messages are being sent securely.
6. Knowing that ***journalists, activists, or dissidents use*** a particular service, app, or device to communicate suggests to me that messages are being sent securely.
7. A service, app, or device being ***expensive to purchase*** suggests to me that messages are being sent securely.
8. A service, app, or device being ***new (e.g., introduced in 2019)*** suggests to me that messages are being sent securely.
9. A service, app, or device being ***endorsed by my government*** suggests to me that messages are being sent securely.

Q21: In general, when using a service, app, or device to communicate with others, how do you determine whether or not messages are being sent *securely*?
<Free Text>
Q22: In general, when using a service, app, or device to communicate with others, how do you determine whether or not messages are being *encrypted* (turned into secret code such that only the authorized parties can view it) as they are sent?
<Free Text>

C.6.3.15 Demographics

Q23: Do you have formal education (Bachelor's degree or higher) in computer science, information technology, or a related field?

- Yes
- No
- Prefer not to say

Q24: Have you held a job in computer science, information technology, or a related field?

- Yes
- No
- Prefer not to say

Q25: What is your gender? *If you prefer not to say, please fill in a "-".*
<Free Text>
Q26: What is your age in years?.

<Free Text>

Q27: What is your highest degree or level of school you have completed? If currently enrolled, select the highest degree received.

- High school education or less
- Some college, no degree
- Trade/technical/vocational training
- Associate's Degree
- Bachelor's Degree
- Graduate degree (master's, doctorate, or professional degree)
- Prefer not to say

Q28: Please specify your ethnicity. (Multiple answers possible)

- White
- Hispanic or Latino
- Black or African American
- Native American or American Indian
- Asian or Pacific Islander
- Prefer not to say
- Other

Q29: What is your current employment status? (Multiple answers possible)

- Employed for wages
- Self-employed
- Military
- Out of work and looking for work
- Out of work but not currently looking for work
- A student
- Retired
- Unable to work
- Prefer not to say
- Other

C.6.3.16 Final Thoughts

Q29: Did you answer all the questions attentive and honest? We're dependent on you being truthful with this question. You'll get paid and won't suffer any consequences at all, and would help us a lot to improve the accuracy of our research.

- No, I wasn't entirely attentive and honest.
- Yes, I filled the questions attentive and honest.

Q30: Do you have any final thoughts on this survey? Please tell us what you liked about this survey and where we could improve. You'll receive your verification code for MTurk on the next page.
<Free Text>

C.6.3.17 Survey End

Thank you for participating! Please post the following code into your MTurk HIT to verify that you have completed this qualification task.
<A random ID to verify the results and pay the user on MTurk>

C.6.4 Verification Survey (Study 5)

Ran on prolific: Some changes required to match the guidelines of Prolific

C.6.4.1 Verification

Unchanged, but participants were allowed to continue, if they failed the verification and were removed afterwards. See Questions Q1 and Q2 from Study 2-4 (Section C.6.3.1). This change was required due to the different terms on Prolific compared to MTurk.

C.6.4.2 Info Text 1

Thanks!
This survey has three parts. In the first part, you will watch a short video depicting a new communication app. In the second part, you will answer questions about the app depicted in the video. In the third part, you will answer more general questions about communication apps.

C.6.4.3 Condition

Note: This subsection is where the different conditions were applied in the survey. The only difference is the video that was displayed to the participants. The video element was large enough that participants could see everything, even if they did not use fullscreen mode. The next button to continue with the survey was only enabled when the end of the video was reached.

Imagine that a new messaging app, Erebus, was recently released to compete with WhatsApp and similar apps. It has become very popular and all of your contacts now use it. Below we show you a brief video of what this app's user interface looks like. The next part of the study will be about this app and its interface. Please, pay close attention to the video and watch it in fullscreen mode.

C.6.4.4 Info Text 2

This next section of the survey (Part 2 of 3) will ask a series of questions about the app depicted in the video you just watched.
Recall that we've asked you to imagine that this app has become very popular and all of your friends and contacts use it.
Assume that only you have physical access to your phone.

C.6.4.5 Questions

Q3a-l: Please rate your agreement with the following statements: *Note: Each question had a 5-point likert (Strongly Agree-Strongly Disagree) as answer option together with an "I don't know option."*

- Erebus prevents my phone company or internet provider from being able to read the messages I send using the app.
- Erebus prevents the makers of the app from being able to read the messages I send using the app.
- Erebus prevents government intelligence agencies from being able to read the messages I send using the app.
- Erebus prevents computer hackers from being able to read the messages I send using the app.
- Erebus prevents other people from sending messages on the app pretending to be me.
- Erebus is unsafe to use.
- Erebus is untrustworthy.
- Erebus is secure.
- Sending a message on Erebus is more secure than sending the message as a standard email.
- Sending a message on Erebus is more secure than sending the message as a text message / SMS.
- Sending a message on Erebus is more secure than conveying the message in person.
- Sending a message on Erebus is more secure than sending the message as a direct message (DM) on Twitter.

Q4a-l: Why? *The why was displayed after each of the questions on Q3a-l.*

C.6.4.6 Questions for Screencast

Q5: In the video you watched, please describe what you observed happening (if anything) when messages were sent or received.
<Free Text>
Q6: What did this indicate to you?
<Free Text>

C.6.4.7 Info Text 3

This third and final section of the study will ask a series of more general questions about your use of a range of communication tools.

C.6.4.8 General Use of Communication Tools

Q6: Which of the communication tools listed below have you used in the last 6 months? (Select all that apply)
Same answer options as Q10 in Survey 2-4 (C.6.3.8)

C.6.4.9 Sensitive Conversations

For this question, please assume that you and all of your contacts regularly use all of the tools listed below and have them already installed on your phone.

Imagine that you need to have a **sensitive** conversation that contains confidential information or that you otherwise wish to protect from people or organizations not involved.
Q7: Which tool(s) listed below would you consider using? (Select all that apply)

- Erebus
- Facebook Messenger
- WhatsApp
- Twitter DM
- Instagram DM
- Skype
- Snapchat
- E-Mail
- iMessage
- SMS / Text message
- None of the above
- Prefer not to say

Q8: Briefly, why did you select those tools?
<Free Text>

C.6.4.10 Casual Conversations

For this question, please assume that you and all of your contacts regularly use all of the tools listed below and have them already installed on your phone.
Imagine that you need to have a **casual** conversation that does not contain any confidential or secret information.
Q9: Which tool(s) listed below would you consider using? (Select all that apply)

- Erebus
- Facebook Messenger
- WhatsApp
- Twitter DM
- Instagram DM
- Skype
- Snapchat
- E-Mail
- iMessage
- SMS / Text message
- None of the above
- Prefer not to say

Q10: Briefly, why did you select those tools?
<Free Text>

C.6.4.11 Secret/Private Mode of Messengers

Note: Each of these questions was only displayed if the participants selected the appropriate messenger in Q6.
Q11: Have you ever used the "secret conversation" feature in the Facebook Messenger mobile app?

- Yes
- No
- I don't know

Q12: Have you ever used the "private conversation" feature in the Skype app?

- Yes
- No

- I don't know

Q13: Have you ever used the "secret chat" feature in the Telegram app?

- Yes
- No
- I don't know

C.6.4.12 Additional Questions

Q14: Please indicate your agreement or disagreement that each of the following factors suggests that messages are being sent securely in a messaging app. *Note: Answer options are a 5-point likert scale (Strongly Agree - Strongly Disagree) with an "I don't know" option.*

- A trustworthy company made the app
- I observe a noticeable processing delay when messages are encrypted/decrypted as they are sent or received
- The maker of the app advertises the app as secure
- I can see messages in encrypted form (i.e., after they are turned into secret code)
- News articles report that the app is secure
- The app states that it uses encryption whenever I open the messaging window
- Journalists, activists, or dissidents use the app to communicate
- Members of the military use the app to communicate
- Technologically savvy friends or colleagues recommend the app
- The app is expensive to purchase
- The app was newly released
- The app has been endorsed by my government
- I have seen advertisements for this app
- The app has a black background
- The app is highly rated in the app store
- The app has a professional-looking logo
- Messages disappear after they are sent or viewed
- The app has many different language options other than English

Q15: In general, when using a service, app, or device to communicate with others, how do you determine whether or not messages are being sent *securely*?
Q16: In general, when using a service, app, or device to communicate with others, how do you determine whether or not messages are being *encrypted* (turned into secret code such that only the authorized parties can view it) as they are sent?

<Free Text>

C.6.4.13 Demographics

Unchanged, see Survey 2-4 (Section C.6.3.15).

C.6.4.14 Survey End

Unchanged.

Bibliography

[1] Noura Abdi, Kopo M. Ramokapane, and Jose M. Such. "More than Smart Speakers: Security and Privacy Perceptions of Smart Home Personal Assistants". In: *Proc. 15th Symposium on Usable Privacy and Security (SOUPS'19)*. USENIX Association, 2019.

[2] Ruba Abu-Salma, Kat Krol, Simon Parkin, Victoria Koh, Kevin Kwan, Jazib Mahboob, Zahra Traboulsi, and M Angela Sasse. "The Security Blanket of the Chat World: An Analytic Evaluation and a User Study of Telegram". In: *Proc. 2nd European Workshop on Usable Security (EuroUSEC'17)*. The Internet Society, 2017.

[3] Ruba Abu-Salma, Elissa M. Redmiles, Blase Ur, and Miranda Wei. "Exploring User Mental Models of End-to-End Encrypted Communication Tools". In: *Proc. 8th USENIX Workshop on Free and Open Communications on the Internet (FOCI'18)*. USENIX Association, 2018.

[4] Ruba Abu-Salma, M. Angela Sasse, Joseph Bonneau, Anastasia Danilova, Alena Naiakshina, and Matthew Smith. "Obstacles to the Adoption of Secure Communication Tools". In: *Proc. 38th IEEE Symposium on Security and Privacy (SP'17)*. IEEE, 2017.

[5] Yasemin Acar, Michael Backes, Sven Bugiel, Sascha Fahl, Patrick McDaniel, and Matthew Smith. "Sok: Lessons learned from android security research for appified software platforms". In: *Proc. 37th IEEE Symposium on Security and Privacy (SP'16)*. IEEE, 2016.

[6] Yasemin Acar, Michael Backes, Sascha Fahl, Simson Garfinkel, Doowon Kim, Michelle L. Mazurek, and Christian Stransky. "Comparing the Usability of Cryptographic APIs". In: *Proc. 38th IEEE Symposium on Security and Privacy (SP'17)*. IEEE, 2017.

[7] Yasemin Acar, Michael Backes, Sascha Fahl, Doowon Kim, Michelle L Mazurek, and Christian Stransky. "You Get Where You're Looking For: The Impact of Information Sources on Code Security". In: *Proc. 37th IEEE Symposium on Security and Privacy (SP'16)*. IEEE, 2016.

[8] Yasemin Acar, Sascha Fahl, and Michelle L. Mazurek. "You are Not Your Developer, Either: A Research Agenda for Usable Security and Privacy Research Beyond End Users". In: *Proc. 2016 IEEE Secure Development Conference (SecDev'16)*. IEEE, 2016.

[9] Yasemin Acar, Christian Stransky, Dominik Wermke, Michelle L. Mazurek, and Sascha Fahl. "Security Developer Studies with GitHub Users: Exploring a Convenience Sample". In: *Proc. 13th Symposium on Usable Privacy and Security (SOUPS'17)*. USENIX Association, 2017.

[10] Yasemin Acar, Christian Stransky, Dominik Wermke, Charles Weir, Michelle L Mazurek, and Sascha Fahl. "Developers Need Support Too: A Survey of Security Advice for Software Developers". In: *Proc. 2017 IEEE Secure Development Conference (SecDev'17)*. IEEE, 2017.

[11] Anne Adams and Martina Angela Sasse. "Users are not the enemy: Why users compromise security mechanisms and how to take remedial measures". In: *Communications of the ACM* 42.12 (1999), pp. 40–46.

[12] Omer Akgul, Wei Bai, Shruti Das, and Michelle L. Mazurek. "Evaluating In-Workflow Messages for Improving Mental Models of End-to-End Encryption". In: *Proc. 30th Usenix Security Symposium (SEC'21)*. USENIX Association, 2021.

[13] Abrar Al-Heeti. *WhatsApp: 65B messages sent each day, and more than 2B minutes of calls*. `https://www.cnet.com/news/whatsapp-65-billion-messages-sent-each-day-and-more-than-2-billion-minutes-of-calls/`. May 2018.

[14] Tarfah Alrashed, Ahmed Hassan Awadallah, and Susan Dumais. "The Lifetime of Email Messages: A Large-Scale Analysis of Email Revisitation". In: *Proc. 2018 Conference on Human Information Interaction & Retrieval (CHIIR '18)*. ACM, 2018.

[15] Apple Inc. *Human Interface Guidelines - iOS Design Themes*. `https://developer.apple.com/ios/human-interface-guidelines/overview/themes/`. 2020.

[16] Apple Inc. *iMessage - Learn more about Messages*. `https://support.apple.com/explore/messages`. 2019.

[17] Erinn Atwater, Cecylia Bocovich, Urs Hengartner, Ed Lank, and Ian Goldberg. "Leading Johnny to Water: Designing for Usability and Trust". In: *Proc. 11th Symposium on Usable Privacy and Security (SOUPS'15)*. USENIX Association, 2015.

[18] Autocrypt team. *Autocrypt Level 1: Enabling encryption, avoiding annoyances*. `https://autocrypt.org/level1.html`.

[19] Noa Avigdor-Elgrabli, Roei Gelbhart, Irena Grabovitch-Zuyev, and Ariel Raviv. "More than Threads: Identifying Related Email Messages". In: *Proc. 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. ACM, 2018.

[20] Wei Bai, Doowon Kim, Moses Namara, Yichen Qian, Patrick Gage Kelley, and Michelle L. Mazurek. "Balancing Security and Usability in Encrypted Email". In: *IEEE Internet Computing* 21.3 (2017).

[21] Wei Bai, Moses Namara, Yichen Qian, Patrick Gage Kelley, Michelle L. Mazurek, and Doowon Kim. "An Inconvenient Trust: User Attitudes toward Security and Usability Tradeoffs for Key-Directory Encryption Systems". In: *Proc. 12th Symposium on Usable Privacy and Security (SOUPS'16)*. USENIX Association, 2016.

[22] Wei Bai, Michael Pearson, Patrick Gage Kelley, and Michelle L. Mazurek. "Improving Non-Experts' Understanding of End-to-End Encryption: An Exploratory Study". In: *Proc. 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2020.

[23] Terry Benzel. "The science of cyber security experimentation: the DETER project". In: *Proc. 27th Annual Computer Security Applications Conference (ACSAC'11)*. ACM, 2011.

[24] K. P. Burnham. "Multimodel Inference: Understanding AIC and BIC in Model Selection". In: *Sociological Methods & Research* 33.2 (2004), pp. 261–304. URL: `http://smr.sagepub.com/cgi/doi/10.1177/0049124104268644`.

[25] J Callas, L Donnerhacke, H Finney, D Shaw, and R Thayer. *OpenPGP message format (RFC 4880)*. `https://datatracker.ietf.org/doc/html/rfc4880`. 2007.

[26] Kathy Charmaz. *Constructing Grounded Theory*. SAGE Publications, 2014.

[27] Jeremy Clark, Paul C. van Oorschot, Scott Ruoti, Kent E. Seamons, and Daniel Zappala. "SoK: Securing Email—A Stakeholder-Based Analysis". In: *ArXiv e-prints* (2018). arXiv: `1804.07706 [cs.CR]`. URL: `http://arxiv.org/abs/1804.07706`.

[28] Sandy Clark, Travis Goodspeed, Perry Metzger, Zachary Wasserman, Kevin Xu, and Matt Blaze. "Why (Special Agent) Johnny (Still) Can't Encrypt: A Security Analysis of the APCO Project 25 Two-Way Radio System". In: *Proc. 20th Usenix Security Symposium (SEC'11)*. USENIX Association, 2011.

[29] Juliet Corbin and Anselm Strauss. "Grounded theory research: Procedures, canons and evaluative criteria". In: *Zeitschrift für Soziologie* 19.6 (1990), pp. 418–427.

[30] Don Davis. "Compliance Defects in Public Key Cryptography". In: *Proc. 5th Usenix Security Symposium (SEC'96)*. USENIX Association, 1996.

[31] Alexander De Luca, Sauvik Das, Martin Ortlieb, Iulia Ion, and Ben Laurie. "Expert and Non-Expert Attitudes towards (Secure) Instant Messaging". In: *Proc. 12th Symposium on Usable Privacy and Security (SOUPS'16)*. USENIX Association, 2016.

[32] Sergej Dechand, Alena Naiakshina, Anastasia Danilova, and Matthew Smith. "In Encryption We Don't Trust: The Effect of End-to-End Encryption to the Masses on User Perception". In: *Proc. 2019 IEEE European Symposium on Security and Privacy (EuroS&P'19)*. IEEE, 2019.

[33] Sergej Dechand, Dominik Schürmann, Karoline Busse, Yasemin Acar, Sascha Fahl, and Matthew Smith. “An Empirical Study of Textual Key-Fingerprint Representations”. In: *Proc. 25th Usenix Security Symposium (SEC'16)*. USENIX Association, 2016.

[34] Albesë Demjaha, Jonathan Spring, Ingolf Becker, Simon Parkin, and M. Angela Sasse. “Metaphors considered harmful? An exploratory study of the effectiveness of functional metaphors for end-to-end encryption”. In: *Proc. Workshop on Usable Security (USEC'18)*. The Internet Society, 2018.

[35] Massimiliano Di Penta, R.E.K. Stirewal, and Eileen Kraemer. “Designing your Next Empirical Study on Program Comprehension”. In: *Proc. 15th IEEE International Conference on Program Comprehension (ICPC'07)*. IEEE, 2007.

[36] Whitfield Diffie and Martin Hellman. “New directions in cryptography”. In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.

[37] Verena Distler, Carine Lallemand, and Vincent Koenig. “Making Encryption Feel Secure: Investigating how Descriptions of Encryption Impact Perceived Security”. In: *Proc. 5th European Workshop on Usable Security (EuroUSEC'20)*. IEEE, 2020.

[38] Verena Distler, Marie-Laure Zollinger, Carine Lallemand, Peter B. Rønne, Peter Y. A. Ryan, and Vincent Koenig. “Security - Visible, Yet Unseen?” In: *Proc. CHI Conference on Human Factors in Computing Systems (CHI'19)*. ACM, 2019.

[39] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J. Alex Halderman. “Neither Snow Nor Rain Nor MITM...: An Empirical Analysis of Email Delivery Security”. In: *Proc. 2015 Internet Measurement Conference (IMC'15)*. ACM, 2015.

[40] S Dusse, P Hoffman, B Ramsdell, L Lundblade, and L Repka. *RFC2311: S/MIME Version 2 Message Specification*. `https://datatracker.ietf.org/doc/html/rfc2311`. 1998.

[41] Manuel Egele, David Brumley, Yanick Fratantonio, and Christopher Kruegel. “An empirical study of cryptographic misuse in android applications”. In: *Proc. 20th ACM Conference on Computer and Communication Security (CCS'13)*. ACM, 2013.

[42] Serge Egelman and Eyal Péer. “Scaling the Security Wall: Developing a Security Behavior Intentions Scale (SeBIS)”. In: *Proc. 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI'15)*. ACM, 2015.

[43] M Elkins, D Del Torto, R Levien, and T Roessler. *RFC3156: MIME Security with OpenPGP*. `https://datatracker.ietf.org/doc/html/rfc3156`. 2001.

[44] William Enck, Patrick Traynor, Patrick D. McDaniel, and Thomas F. La Porta. "Exploiting open functionality in SMS-capable cellular networks". In: *Proc. 12th ACM Conference on Computer and Communication Security (CCS'05)*. ACM, 2005.

[45] Ksenia Ermoshina, Francesca Musiani, and Harry Halpin. "End-to-End Encrypted Messaging Protocols: An Overview". In: *Proc. 6th International Conference on Internet Science (INSCI'19)*. Springer, 2016.

[46] Facebook Inc. *Facebook Messenger*. `https://www.messenger.com/features`. 2019.

[47] Sascha Fahl, Yasemin Acar, Henning Perl, and Matthew Smith. "Why Eve and Mallory (Also) Love Webmasters: A Study on the Root Causes of SSL Misconfigurations". In: *Proc. 9th ACM Symposium on Information, Computer and Communication Security (ASIACCS'14)*. ACM, 2014.

[48] Sascha Fahl, Marian Harbach, Thomas Muders, Lars Baumgärtner, Bernd Freisleben, and Matthew Smith. "Why Eve and Mallory love Android: An analysis of Android SSL (in)security". In: *Proc. 19th ACM Conference on Computer and Communication Security (CCS'12)*. ACM, 2012.

[49] Sascha Fahl, Marian Harbach, Thomas Muders, Matthew Smith, and Uwe Sander. "Helping Johnny 2.0 to Encrypt His Facebook Conversations". In: *Proc. 8th Symposium on Usable Privacy and Security (SOUPS'12)*. ACM, 2012.

[50] Sascha Fahl, Marian Harbach, Henning Perl, Markus Koetter, and Matthew Smith. "Rethinking SSL Development in an Appified World". In: *Proc. 20th ACM Conference on Computer and Communication Security (CCS'13)*. ACM, 2013.

[51] Federal Trade Commission. *FTC Alleges Operators of Two Commercial Websites Failed to Protect Consumers' Data*. `https://www.ftc.gov/news-events/press-releases/2019/04/ftc-alleges-operators-two-commercial-websites-failed-protect`. visited 24/08/2021. Apr. 2019.

[52] Federal Trade Commission. *FTC Requires Zoom to Enhance its Security Practices as Part of Settlement*. `https://www.ftc.gov/news-events/press-releases/2020/11/ftc-requires-zoom-enhance-its-security-practices-part-settlement`. visited 24/08/2021. Nov. 2020.

[53] Adrienne Porter Felt, Alex Ainslie, Robert W. Reeder, Sunny Consolvo, Somas Thyagaraja, Alan Bettes, Helen Harris, and Jeff Grimes. "Improving SSL Warnings: Comprehension and Adherence". In: *Proc. 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI'15)*. ACM, 2015.

[54] Adrienne Porter Felt, Erika Chin, Steve Hanna, Dawn Song, and David Wagner. "Android permissions demystified". In: *Proc. 18th ACM Conference on Computer and Communication Security (CCS'11)*. ACM, 2011.

[55] Adrienne Porter Felt, Robert W. Reeder, Alex Ainslie, Helen Harris, Max Walker, Christopher Thompson, Mustafa Embre Acer, Elisabeth Morant, and Sunny Consolvo. “Rethinking Connection Security Indicators”. In: *Proc. 12th Symposium on Usable Privacy and Security (SOUPS’16)*. USENIX Association, 2016.

[56] Felix Fischer, Konstantin Böttinger, Huang Xiao, Christian Stransky, Yasemin Acar, Michael Backes, and Sascha Fahl. “Stack Overflow Considered Harmful? The Impact of Copy&Paste on Android Application Security”. In: *Proc. 38th IEEE Symposium on Security and Privacy (SP’17)*. IEEE, 2017.

[57] Danyel Fisher, A. J. Brush, Eric Gleave, and Marc A. Smith. “Revisiting Whittaker & Sidner’s "Email Overload" Ten Years Later”. In: *Proc. 2006 20th Anniversary Conference on Computer Supported Cooperative Work*. ACM, 2006.

[58] Ronald P Fisher and R Edward Geiselman. *Memory-Enhancing Techniques for Investigative Interviewing: The Cognitive Interview*. Charles C Thomas Publisher, 1992.

[59] Alain Forget, Saranga Komanduri, Alessandro Acquisti, Nicolas Christin, Lorrie Faith Cranor, and Rahul Telang. *Security behavior observatory: Infrastructure for long-term monitoring of client machines*. Tech. rep. Carnegie Mellon University, CyLab, 2014.

[60] Ian D. Foster, Jon Larson, Max Masich, Alex C. Snoeren, Stefan Savage, and Kirill Levchenko. “Security by Any Other Name: On the Effectiveness of Provider Based Email Security”. In: *Proc. 22nd ACM Conference on Computer and Communication Security (CCS’15)*. ACM, 2015.

[61] Simson Garfinkel and Heather Richter Lipford. “Usable Security: History, Themes, and Challenges”. In: *Synthesis Lectures on Information Security, Privacy, and Trust* 5.2 (2014), pp. 1–124.

[62] Simson L. Garfinkel, David Margrave, Jeffrey I. Schiller, Erik Nordlander, and Robert C. Miller. “How to Make Secure Email Easier To Use”. In: *Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI’05)*. ACM, 2005.

[63] Simson L. Garfinkel and Robert C. Miller. “Johnny 2: A User Test of Key Continuity Management with S/MIME and Outlook Express”. In: *Proc. 1st Symposium on Usable Privacy and Security (SOUPS’05)*. ACM, 2005.

[64] Simson L. Garfinkel, Jeffrey I. Schiller, Erik Nordlander, David Margrave, and Robert C. Miller. “Views, Reactions and Impact of Digitally-Signed Mail in e-Commerce”. In: *Proc. 9th International Conference on Financial Cryptography and Data Security (FC’05)*. Springer, 2005.

[65] Shirley Gaw, Edward W. Felten, and Patricia Fernandez-Kelly. “Secrecy, Flagging, and Paranoia: Adoption Criteria in Encrypted Email”. In: *Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI’06)*. ACM, 2006.

[66] Nina Gerber, Verena Zimmermann, Birgit Henhapl, Sinem Emeröz, and Melanie Volkamer. "Finally Johnny Can Encrypt: But Does This Make Him Feel More Secure?" In: *Proc. 13th International Conference on Availability, Reliability and Security (ARES'18)*. ACM, 2018.

[67] Peter Leo Gorski, Luigi Lo Iacono, Dominik Wermke, Christian Stransky, Sebastian Möller, Yasemin Acar, and Sascha Fahl. "Developers Deserve Security Warnings, Too: On the Effect of Integrated Security Advice on Cryptographic API Misuse". In: *Proc. 14th Symposium on Usable Privacy and Security (SOUPS'18)*. USENIX Association, 2018.

[68] Matthew Green and Matthew Smith. "Developers are Not the Enemy!: The Need for Usable Security APIs". In: *IEEE Security & Privacy* 14.5 (2016), pp. 40–46.

[69] Glenn Greenwald and Ewen MacAskill. *NSA Prism program taps into user data of Apple, Google and others*. 2013.

[70] Glenn Greenwald, Ewen MacAskill, and Laura Poitras. *Edward Snowden: the whistleblower behind the NSA surveillance revelations*. 2013.

[71] D. F. Hamilton, J. V. Lane, P. Gaston, J. T. Patton, D. J. MacDonald, A. H. R. W. Simpson, and C. R. Howie. "Assessing treatment outcomes using a single question: the Net Promoter Score". In: *The Bone & Joint Journal* 96.5 (2014), pp. 622–628.

[72] Julia Hanson, Miranda Wei, Sophie Veys, Matthew Kugler, Lior Strahilevitz, and Blase Ur. "Taking Data Out of Context to Hyper-Personalize Ads: Crowdworkers' Privacy Perceptions and Decisions to Disclose Private Information". In: *Proc. CHI Conference on Human Factors in Computing Systems (CHI'20)*. ACM, 2020.

[73] David J. Hauser and Norbert Schwarz. "Attentive Turkers: MTurk participants perform better on online attention checks than do subject pool participants". In: *Behavior Research Methods* 48.1 (2016), pp. 400–407.

[74] Internet Security Research Group (ISRG). *Let's Encrypt Stats*. `https://letsencrypt.org/stats/#percent-pageloads`. visited 24/08/2021. Sept. 2021.

[75] Ryan Kennedy, Scott Clifford, Tyler Burleigh, Ryan Jewell, and Philip Waggoner. *The Shape of and Solutions to the MTurk Quality Crisis*. `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3272468`. 2018.

[76] Klaus Krippendorff. *Content Analysis: An Introduction to Its Methodology (2nd ed.)* SAGE Publications, 2004.

[77] Jon A. Krosnick and Stanley Presser. "Question and Questionnaire Design". In: *Handbook of Survey Research*. Emerald Publishing, 2010, pp. 263–314.

[78] Stefan Krüger, Sarah Nadi, Michael Reif, Karim Ali, Mira Mezini, Eric Bodden, Florian Göpfert, Felix Günther, Christian Weinert, Daniel Demmler, and Ram Kamath. "CogniCrypt: Supporting Developers in Using Cryptography". In: *Proc. 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE'17)*. IEEE Press, 2017.

[79] A. Lerner, E. Zeng, and F. Roesner. "Confidante: Usable Encrypted Email: A Case Study with Lawyers and Journalists". In: *Proc. 2017 IEEE European Symposium on Security and Privacy (EuroS&P'17)*. IEEE, 2017.

[80] James R. Lewis, Brian Utesch, and Deborah E. Maher. "UMUX-LITE: when there's no time for the SUS". In: *Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'13)*. ACM, 2013.

[81] Fanny Lalonde Lévesque and Jóse M. Fernandez. "Computer security clinical trials: Lessons learned from a 4-month pilot study". In: *Proc. 7th USENIX Workshop on Cyber Security Experimentation and Test (CSET'14)*. USENIX Association, 2014.

[82] Max-Emanuel Maurer, Alexander De Luca, and Tobias Stockinger. "Shining Chrome: Using Web Browser Personas to Enhance SSL Certificate Visualization". In: *Proc. 13th IFIP TC 13 International Conference on Human-Computer Interaction (INTERACT '11)*. Springer, 2011.

[83] Juan Ramón Ponce Mauriés, Kat Krol, Simon Parkin, Ruba Abu-Salma, and M. Angela Sasse. "Dead on Arrival: Recovering from Fatal Flaws in Email Encryption Tools". In: *The LASER Workshop: Learning from Authoritative Security Experiment Results (LASER 2017*. USENIX Association, 2017.

[84] Susan E. McGregor, Elizabeth Anne Watkins, Mahdi Nasrullah Al-Ameen, Kelly Caine, and Franziska Roesner. "When the Weakest Link is Strong: Secure Collaboration in the Case of the Panama Papers". In: *Proc. 26th Usenix Security Symposium (SEC'17)*. USENIX Association, 2017.

[85] Sarah Meyer. *A Tale of Two Data Breaches – Under Armour and Panera Bread*. `https://www.cpomagazine.com/cyber-security/a-tale-of-two-data-breaches-under-armour-and-panera-bread/`. visited 24/08/2021. Apr. 2018.

[86] Micah Lee. *Edward Snowden explains how to reclaim your privacy*. `https://theintercept.com/2015/11/12/edward-snowden-explains-how-to-reclaim-your-privacy/`. 2015.

[87] Microsoft Corporation. *Skype main page*. `https://www.skype.com/`. 2019.

[88] Kai Mindermann, Philipp Keck, and Stefan Wagner. "How Usable are Rust Cryptography APIs?" In: *Proc. 2018 IEEE International Conference on Software Quality, Reliability and Security (QRS'18)*. IEEE, 2018.

[89] Mainack Mondal, Günce Su Yilmaz, Noah Hirsch, Mohammad Taha Khan, Michael Tang, Christopher Tran, Chris Kanich, Blase Ur, and Elena Zheleva. "Moving Beyond Set-It-And-Forget-It Privacy Settings on Social Media". In: *Proc. 26th ACM Conference on Computer and Communication Security (CCS'19)*. ACM, 2019.

[90] Sarah Nadi, Stefan Krüger, Mira Mezini, and Eric Bodden. ""Jumping Through Hoops": Why do Java Developers Struggle With Cryptography APIs?" In: *Proc. 38th IEEE/ACM International Conference on Software Engineering (ICSE'16)*. ACM, 2016.

[91] Alena Naiakshina, Anastasia Danilova, Eva Gerlitz, Emanuel von Zezschwitz, and Matthew Smith. ""If You Want, I Can Store the Encrypted Password": A Password-Storage Field Study with Freelance Developers". In: *Proc. CHI Conference on Human Factors in Computing Systems (CHI'19)*. ACM, 2019.

[92] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, Marco Herzog, Sergej Dechand, and Matthew Smith. "Why Do Developers Get Password Storage Wrong?: A Qualitative Usability Study". In: *Proc. 24th ACM Conference on Computer and Communication Security (CCS'17)*. ACM, 2017.

[93] Anandatirtha Nandugudi, Anudipa Maiti, Taeyeon Ki, Fatih Bulut, Murat Demirbas, Tevfik Kosar, Chunming Qiao, Steven Y. Ko, and Geoffrey Challen. "PhoneLab: A Large Programmable Smartphone Testbed". In: *Proc. 1st International Workshop on Sensing and Big Data Mining (SENSEMINE'13)*. ACM, 2013.

[94] Jakob Nielsen. "Enhancing the Explanatory Power of Usability Heuristics". In: *Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'94)*. ACM, 1994.

[95] Marten Oltrogge, Yasemin Acar, Sergej Dechand, Matthew Smith, and Sascha Fahl. "To Pin or Not to Pin—Helping App Developers Bullet Proof Their TLS Connections". In: *Proc. 24th Usenix Security Symposium (SEC'15)*. USENIX Association, 2015.

[96] Marten Oltrogge, Erik Derr, Christian Stransky, Yasemin Acar, Sascha Fahl, Christian Rossow, Giancarlo Pellegrino, Sven Bugiel, and Michael Backes. "The Rise of the Citizen Developer: Assessing the Security Impact of Online App Generators". In: *Proc. 39th IEEE Symposium on Security and Privacy (SP'18)*. IEEE, 2018.

[97] Marten Oltrogge, Nicolas Huaman, Sabrina Amft, Yasemin Acar, Michael Backes, and Sascha Fahl. "Why Eve and Mallory Still Love Android: Revisiting TLS (In)Security in Android Applications". In: *Proc. 30th Usenix Security Symposium (SEC'21)*. USENIX Association, 2021.

[98] Stefan Palan and Christian Schitter. “Prolific.ac — A subject pool for online experiments”. In: *Journal of Behavioral and Experimental Finance* 17 (2018), pp. 22–27.

[99] Eyal Peer, Laura Brandimarte, Sonam Samat, and Alessandro Acquisti. “Beyond the Turk: Alternative platforms for crowdsourcing behavioral research”. In: *Journal of Experimental Social Psychology* 70 (2017), pp. 153 –163.

[100] Eyal Peer, Joachim Vosgerau, and Alessandro Acquisti. “Reputation as a sufficient condition for data quality on Amazon Mechanical Turk”. In: *Behavior Research Methods* 46.4 (2014), pp. 1023–1031.

[101] Andrea Peterson. *Edward Snowden sent Glenn Greenwald this video guide about encryption for journalists. Greenwald ignored it.* `https://www.washingtonpost.com/news/the-switch/wp/2014/05/14/edward-snowden-sent-glenn-greenwald-this-video-guide-about-encryption-for-journalists-greenwald-ignored-it/`. 2014.

[102] Prolific. *Prolific | Online participant recruitment for surveys and market research.* `https://prolific.co/`. 2020.

[103] Elissa M. Redmiles, Sean Kross, and Michelle L. Mazurek. “How I Learned to Be Secure: A Census-Representative Survey of Security Advice Sources and Behavior”. In: *Proc. 23nd ACM Conference on Computer and Communication Security (CCS’16).* ACM, 2016.

[104] Elissa M Redmiles, Sean Kross, and Michelle L Mazurek. “How Well Do My Results Generalize? Comparing Security and Privacy Survey Results from MTurk, Web, and Telephone Samples”. In: *Proc. 40th IEEE Symposium on Security and Privacy (SP’19).* IEEE, 2019.

[105] Karen Renaud, Melanie Volkamer, and Arne Renkema-Padmos. “Why Doesn’t Jane Protect Her Privacy?” In: *Proc. on Privacy Enhancing Technologies 2014 (PETS’14).* Springer, 2014.

[106] Reporters Committee for Freedom of the Press. *Whistleblowers to journalists: protect your data.* `https://www.rcfp.org/journals/whistleblowers-journalists/`. visited 24/08/2021.

[107] Eric Rescorla. “The transport layer security (tls) protocol version 1.3”. In: *Transport* draft-ietf-tls-tls13-18.1 (2016), pp. 1–118.

[108] Ronald L Rivest, Adi Shamir, and Leonard Adleman. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. In: *Communications of the ACM* 21.2 (1978), 120—126.

[109] Juan Carlos Roca, Juan José García, and Juan José de la Vega. “The importance of perceived trust, security and privacy in online trading systems”. In: *Information Management & Computer Security* 17.2 (2009), pp. 96–113.

[110] Franziska Roesner, Brian T Gill, and Tadayoshi Kohno. “Sex, Lies, or Kittens? Investigating the Use of Snapchat’s Self-Destructing Messages”. In: *Proc. 18th International Conference on Financial Cryptography and Data Security (FC’14).* Springer, 2014.

[111] Volker Roth, Tobias Straub, and Kai Richter. “Security and usability engineering with particular attention to electronic mail”. In: *International Journal of Human-Computer Studies* 63.1 (2005).

[112] Christoph Rottermanner, Peter Kieseberg, Markus Huber, Martin Schmiedecker, and Sebastian Schrittwieser. “Privacy and Data Protection in Smartphone Messengers”. In: *Proc. 17th International Conference on Information Integration and Web-based Applications & Services (iiWAS’15).* ACM, 2015.

[113] Mekhala Roy. *Marriott data breach exposed 5 million unencrypted passport numbers.* `https://searchsecurity.techtarget.com/news/252455488/Marriott-data-breach-exposed-5-million-unencrypted-passport-numbers/`. visited 24/08/2021. Jan. 2019.

[114] Andrew Ruef, Michael Hicks, James Parker, Dave Levin, Michelle L. Mazurek, and Piotr Mardziel. “Build It, Break It, Fix It: Contesting Secure Development”. In: *Proc. 23nd ACM Conference on Computer and Communication Security (CCS’16).* ACM, 2016.

[115] Scott Ruoti, Jeff Andersen, Scott Heidbrink, Mark O’Neill, Elham Vaziripour, Justin Wu, Daniel Zappala, and Kent Seamons. “"We’re on the Same Page": A Usability Study of Secure Email Using Pairs of Novice Users”. In: *Proc. CHI Conference on Human Factors in Computing Systems (CHI’16).* ACM, 2016.

[116] Scott Ruoti, Jeff Andersen, Travis Hendershot, Daniel Zappala, and Kent E. Seamons. “Private Webmail 2.0: Simple and Easy-to-Use Secure Email”. In: *Proc. 29th Annual Symposium on User Interface Software and Technology (UIST’16).* ACM, 2016.

[117] Scott Ruoti, Jeff Andersen, Tyler Monson, Daniel Zappala, and Kent Seamons. “A Comparative Usability Study of Key Management in Secure Email”. In: *Proc. 14th Symposium on Usable Privacy and Security (SOUPS’18).* USENIX Association, 2018.

[118] Scott Ruoti, Jeff Andersen, Daniel Zappala, and Kent E. Seamons. “Why Johnny Still, Still Can’t Encrypt: Evaluating the Usability of a Modern PGP Client”. In: *ArXiv e-prints* (2016). arXiv: `1510.08555 [cs.CR]`.

[119] Scott Ruoti, Nathan Kim, Ben Burgon, Timothy Van Der Horst, and Kent Seamons. “Confused Johnny: When Automatic Encryption Leads to Confusion and Mistakes”. In: *Proc. 9th Symposium on Usable Privacy and Security (SOUPS’13).* ACM, 2013.

[120] Scott Ruoti and Kent E. Seamons. "Johnny's Journey Toward Usable Secure Email". In: *IEEE Security & Privacy* 17.6 (2019).

[121] M. A. Sasse, S. Brostoff, and D. Weirich. "Transforming the 'Weakest Link' — a Human/Computer Interaction Approach to Usable and Effective Security". In: *BT Technology Journal* 19.3 (2001), pp. 122–131.

[122] J. Schaad, B. Ramsdell, and S. Turner. *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification*. `https://tools.ietf.org/html/rfc8551`. Apr. 2019.

[123] Nora Cate Schaeffer and Stanley Presser. "The Science of Asking Questions". In: *Annual Review of Sociology* 29.1 (2003), pp. 65–88.

[124] Emily Schechter. *A secure web is here to stay*. `https://security.googleblog.com/2018/02/a-secure-web-is-here-to-stay.html?m=1`. visited 24/08/2021. Feb. 2018.

[125] Stuart E Schechter, Rachna Dhamija, Andy Ozment, and Ian Fischer. "The Emperor's New Security Indicators". In: *Proc. 28th IEEE Symposium on Security and Privacy (SP'07)*. IEEE, 2007.

[126] Svenja Schröder, Markus Huber, David Wind, and Christoph Rottermanner. "When SIGNAL hits the Fan: On the Usability and Security of State-of-the-Art Secure Mobile Messaging". In: *Proc. 1st European Workshop on Usable Security (EuroUSEC'16)*. The Internet Society, 2016.

[127] Steve Sheng, Levi Broderick, Colleen Alison Koranda, and Jeremy J Hyland. "Why Johnny Still Can't Encrypt: Evaluating the Usability of Email Encryption Software". In: *Proc. 2nd Symposium on Usable Privacy and Security (SOUPS'06)*. ACM, 2006.

[128] Christos Siaterlis and Marcelo Masera. "A Review of Available Software for the Creation of Testbeds for Internet Security Research". In: *Proc. 1st International Conference on Advances in System Simulation (SIMUL'09)*. IEEE, 2009.

[129] Janet Siegmund, Norbert Siegmund, and Sven Apel. "Views on Internal and External Validity in Empirical Software Engineering". In: *Proc. 37th IEEE/ACM International Conference on Software Engineering (ICSE'15)*. IEEE, 2015.

[130] Signal. *Signal Support - Is it private? Can I trust it?* `https://support.signal.org/hc/en-us/articles/360007320391-Is-it-private-Can-I-trust-it-`. visited 02/01/2021.

[131] Signal, a 501c3 nonprofit. *Signal Messenger*. `https://www.signal.org`. 2019.

[132] Edward Snowden. *Without encryption, we will lose all privacy. This is our new battleground*. `https://www.theguardian.com/commentisfree/2019/oct/15/encryption-lose-privacy-us-uk-australia-facebook`. visited 24/08/2021. Oct. 2019.

[133] Jennifer Sobey, Robert Biddle, Paul C. van Oorschot, and Andrew S. Patrick. "Exploring User Reactions to New Browser Cues for Extended Validation Certificates". In: *Proc. 13th European Symposium on Research in Computer Security (ESORICS'08)*. Springer, 2008.

[134] Andreas Sotirakopoulos, Kirstie Hawkey, and Konstantin Beznosov. "On the Challenges in Usable Security Lab Studies: Lessons Learned from Replicating a Study on SSL Warnings". In: *Proc. 7th Symposium on Usable Privacy and Security (SOUPS'11)*. ACM, 2011.

[135] Statista Inc. *Most popular mobile messaging apps in the United States as of September 2019, by monthly active users*. `https://www.statista.com/statistics/350461/mobile-messenger-app-usage-usa`. 2019.

[136] Christian Stransky, Yasemin Acar, Duc Cuong Nguyen, Dominik Wermke, Elissa M. Redmiles, Doowon Kim, Michael Backes, Simson Garfinkel, Michelle L. Mazurek, and Sascha Fahl. "Lessons Learned from Using an Online Platform to Conduct Large-Scale, Online Controlled Security Experiments with Software Developers". In: *Proc. 10th USENIX Workshop on Cyber Security Experimentation and Test (CSET'17)*. USENIX Association, 2017.

[137] Christian Stransky, Dominik Wermke, Johanna Schrader, Nicolas Huaman, Yasemin Acar, Anna Lena Fehlhaber, Miranda Wei, Blase Ur, and Sascha Fahl. "On the Limited Impact of Visualizing Encryption: Perceptions of E2E Messaging Security". In: *Proc. 17th Symposium on Usable Privacy and Security (SOUPS'21)*. USENIX Association, 2021.

[138] Christian Stransky, Oliver Wiese, Volker Roth, Yasemin Acar, and Sascha Fahl. *Companion Website - EMail Paper*. `https://publications.teamusec.de/2022-oakland-email/`. 2021.

[139] Anselm Strauss and Juliet M Corbin. *Grounded theory in practice*. SAGE Publications, 1997.

[140] Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and Lorrie Faith Cranor. "Crying Wolf: An Empirical Study of SSL Warning Effectiveness." In: *Proc. 18th Usenix Security Symposium (SEC'09)*. USENIX Association, 2009.

[141] Joshua Tan, Lujo Bauer, Joseph Bonneau, Lorrie Faith Cranor, Jeremy Thomas, and Blase Ur. "Can Unicorns Help Users Compare Crypto Key Fingerprints?" In: *Proc. CHI Conference on Human Factors in Computing Systems (CHI'17)*. ACM, 2017.

[142] O. Tange. "GNU Parallel - The Command-Line Power Tool". In: *;login: The USENIX Magazine* 36.1 (2011).

[143] H. Tankovska. *Statista - Number of e-mail users worldwide from 2017 to 2024*. `https://www.statista.com/statistics/255080/number-of-e-mail-users-worldwide/`. 2021.

[144] Telegram FZ LLC. *Telegram FAQ*. `https://telegram.org/faq#q-so-how-do-you-encrypt-data`. 2019.

[145] ThinkAutomation. *The 50-year history of email*. `https://www.thinkautomation.com/histories/the-50-year-history-of-email/`. visited 24/08/2021.

[146] Trend Micro Incorporated. *TalkTalk Reports Breach, up to 4 Million Unencrypted Records Stolen*. `https://www.trendmicro.com/vinfo/de/security/news/cyber-attacks/talktalk-breach-up-to-4-million-unencrypted-records-stolen`. visited 24/08/2021. Oct. 2015.

[147] Alexander Ulrich, Ralph Holz, Peter Hauck, and Georg Carle. "Investigating the OpenPGP Web of Trust". In: *Proc. 16th European Symposium on Research in Computer Security (ESORICS'11)*. Springer, 2011.

[148] Nik Unger, Sergej Dechand, Joseph Bonneau, Sascha Fahl, Henning Perl, Ian Goldberg, and Matthew Smith. "SoK: Secure Messaging". In: *Proc. 36th IEEE Symposium on Security and Privacy (SP'15)*. IEEE, 2015.

[149] Anthony Vance, David Eargle, Jeffrey L. Jenkins, C. Brock Kirwan, and Bonnie Brinton Anderson. "The Fog of Warnings: How Non-essential Notifications Blur with Security Warnings". In: *Proc. 15th Symposium on Usable Privacy and Security (SOUPS'19)*. USENIX Association, 2019.

[150] Elham Vaziripour, Devon Howard, Jake Tyler, Mark O'Neill, Justin Wu, Kent Seamons, and Daniel Zappala. "I Don't Even Have to Bother Them! Using Social Media to Automate the Authentication Ceremony in Secure Messaging". In: *Proc. CHI Conference on Human Factors in Computing Systems (CHI'19)*. ACM, 2019.

[151] Elham Vaziripour, Justin Wu, Mark O'Neill, Daniel Metro, Josh Cockrell, Timothy Moffett, Jordan Whitehead, Nick Bonner, Kent Seamons, and Daniel Zappala. "Action Needed! Helping Users Find and Complete the Authentication Ceremony in Signal". In: *Proc. 14th Symposium on Usable Privacy and Security (SOUPS'18)*. USENIX Association, 2018.

[152] Elham Vaziripour, Justin Wu, Mark O'Neill, Jordan Whitehead, Scott Heidbrink, Kent Seamons, and Daniel Zappala. "Is that you, Alice? A Usability Study of the Authentication Ceremony of Secure Messaging Applications". In: *Proc. 13th Symposium on Usable Privacy and Security (SOUPS'17)*. USENIX Association, 2017.

[153] Viber Media S.à r.l. *Features | Viber*. `https://www.viber.com/features/`. 2019.

[154] Rick Wash, Emilee Rader, and Chris Fennell. "Can People Self-Report Security Accurately?: Agreement Between Self-Report and Behavioral Measures". In: *Proc. CHI Conference on Human Factors in Computing Systems (CHI'17)*. ACM, 2017.

[155] WeAreDynamo.org. *Guidelines for Academic Requesters*. `http://wiki.wearedynamo.org/index.php/Guidelines_for_Academic_Requesters`. 2017.

[156] Tara Whalen and Kori M. Inkpen. "Gathering Evidence: Use of Visual Security Cues in Web Browsers". In: *Proc. Graphics Interface 2005 Conference (GI'05)*. Canadian Human-Computer Communications Society, 2005.

[157] WhatsApp LLC. *end-to-end encryption - WhatsApp*. `https://blog.whatsapp.com/end-to-end-encryption/?lang=en`. visited 02/01/2021.

[158] WhatsApp LLC. *WhatsApp - Features*. `https://www.whatsapp.com/features/`. 2019.

[159] Steve Whittaker and Candace Sidner. "Email Overload: Exploring Personal Information Management of Email." In: *Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'96)*. ACM, 1996.

[160] Alma Whitten and J. D. Tygar. "Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0". In: *Proc. 8th Usenix Security Symposium (SEC'99)*. USENIX Association, 1999.

[161] Justin Wu and Daniel Zappala. "When is a Tree Really a Truck? Exploring Mental Models of Encryption". In: *Proc. 14th Symposium on Usable Privacy and Security (SOUPS'18)*. USENIX Association, 2018.

[162] Yuxi Wu, Panya Gupta, Miranda Wei, Yasemin Acar, Sascha Fahl, and Blase Ur. "Your Secrets Are Safe: How Browsers' Explanations Impact Misconceptions About Private Browsing Mode". In: *Proc. 27th International Conference on World Wide Web (WWW'18)*. International World Wide Web Conferences Steering Committee, 2018.

[163] Alexander Zeier, Alexander Wiesmaier, and Andreas Heinemann. "27 Years and 81 Million Opportunities Later: Investigating the Use of Email Encryption for an Entire University". In: *Proc. 14th International Workshop on Security (IWSEC'19)*. Springer, 2019.

[164] Philip Zimmermann. *PGP Version 2.6.2 User's Guide*. `ftp://ftp.pgpi.org/pub/pgp/2.x/doc/pgpdoc1.txt`. Oct. 1994.

[165] Mary Ellen Zurko and Richard T. Simon. "User-Centered Security". In: *Proc. 1996 New Security Paradigms Workshop (NSPW'96)*. ACM, 1996.

CURRICULUM VITAE: Christian Stransky

Personal Information

Name	Christian Stransky

Education

08/2015	**M.Sc. Computer Science** Leibniz University Hannover, Germany.
09/2013	**B.Sc. Computer Science** Leibniz University Hannover, Germany.

Experience

Academic service: Web Chair Workshop on Security Information Workers (WSIW) 2021, 2020, 2019, 2018; Web Chair EuroUSEC Workshop 2017 (Co-Located with *IEEE* EuroS&P)

PC Member at: The 2021 European Symposium on Usable Security (EuroUSEC 2021); 5th European Workshop on Usable Security (EuroUSEC 2020); 7th Workshop on Context Systems Design, Evaluation and Optimization (CoSDEO 2020) (Co-Located with *IEEE* PerCom); 6th Workshop on Context Systems Design, Evaluation and Optimization (CoSDEO 2018) (Co-Located with *IEEE* PerCom)

CISPA Helmholtz Center for Information Security, Germany (2021 - 2022): Full time researcher.

Information Security Group, Leibniz University Hannover, Germany (2017 - 2021): Full time researcher.

Information Security and Cryptography Group, CISPA, Saarland University, Germany (2015 - 2017): Full time researcher.

Peer-reviewed Papers (selected)

C. Stransky, O. Wiese, V. Roth, Y. Acar and S. Fahl, 27 Years and 81 Million Opportunities Later: Investigating the Use of Email Encryption for an Entire University, To appear in 2022 IEEE Symposium on Security and Privacy, SP 2022

C. Stransky, D. Wermke, J. Schrader, N. Huaman, Y. Acar, A. L. Fehlhaber, M. Wei, B. Ur and S. Fahl, On the Limited Impact of Visualizing Encryption: Perceptions of E2E Messaging Security, Seventeenth Symposium on Usable Privacy and Security, SOUPS 2021

N. Huaman, B. von Skarczinski, C. Stransky, D. Wermke, Y. Acar, A. Dreissigacker, S. Fahl, A Large-Scale Interview Study on Information Security in and Attacks against Small and Medium-sized Enterprises, 30th USENIX Security Symposium, USENIX Security 2021

D. Wermke, C. Stransky, N. Huaman, N. Busch, Y. Acar and S. Fahl, Cloudy with a Chance of Misconceptions: Exploring Users' Perceptions and Expectations of Security and Privacy in Cloud Office Suites, Sixteenth Symposium on Usable Privacy and Security, SOUPS 2020

P. L. Gorski, L. L. Iacono, D. Wermke, C. Stransky, S. Möller, Y. Acar and S. Fahl, Developers Deserve Security Warnings, Too: On the Effect of Integrated Security Advice on Cryptographic API Misuse, Fourteenth Symposium on Usable Privacy and Security, SOUPS 2018

M. Oltrogge, E. Derr, C. Stransky, Y. Acar, S. Fahl, C. Rossow, G. Pellegrino, S. Bugiel, M. Backes, The Rise of the Citizen Developer: Assessing the security impact of online app generators, 2018 IEEE Symposium on Security and Privacy, SP 2018

Y. Acar, C. Stransky, D. Wermke, C. Weir, M. Mazurek, S. Fahl; Developers Need Support, Too: A Survey of Security Advice for Software Developers; IEEE Secure Development Conference 2017

C. Stransky, Y. Acar, D. C. Nguyen, D. Wermke, D. Kim, E. M. Redmiles, M. Backes, S. L. Garfinkel, M. L. Mazurek and S. Fahl; Lessons Learned from Using an Online Platform to Conduct Large-Scale, Online Controlled Security Experiments with Software Developers; 10th USENIX Workshop on Cyber Security Experimentation and Test, CSET 2017

Y. Acar, C. Stransky, D. Wermke, M. Mazurek, S. Fahl; Security Developer Studies with GitHub Users: Exploring a Convenience Sample; Thirteenth Symposium on Usable Privacy and Security, SOUPS 2017

F. Fischer, K. Böttinger, H. Xiao, C. Stransky, Y. Acar, M. Backes, S. Fahl; StackOverflow Considered Harmful? - The Impact of Copy & Paste on Android Application Security; 2017 IEEE Symposium on Security and Privacy, SP 2017

Y. Acar, M. Backes, S. Fahl, S. Garfinkel, D. Kim, M. Mazurek, C. Stransky; Comparing the Usability of Cryptographic APIs; 2017 IEEE Symposium on Security and Privacy, SP 2017

Y. Acar, M. Backes, S. Fahl, D. Kim, M. Mazurek, C. Stransky: How Internet Resources Might Be Helping You Develop Faster but Less Securely, IEEE Security & Privacy (Volume: 15, Issue: 2), 2017

Y. Acar, M. Backes, S. Fahl, D. Kim, M. Mazurek, C. Stransky: You Get Where You're Looking For - The Impact of Information Sources on Code Security, 2016 IEEE Symposium on Security and Privacy, SP 2016