# Enabling users to enforce privacy

## Towards a privacy-preserving document life cycle when digitizing and sharing documents

by

**J.L. Ouwehand**

in partial fulfillment of the requirements for the degree of

**Master of Science**

in Software Engineering

at the Open University of the Netherlands, Faculty of Science
Master's Programme in Software Engineering

to be defended publicly on December 18, 2023 at 13:00.

| | | |
|---|---|---|
| Student number: | 851892028 | |
| Course code: | IM9906 | |
| Thesis committee: | dr. Fabian van den Broek (chairman), | Open University |
| | dr. ir. Hugo Jonker (supervisor), | Open University |

Open Universiteit
de beste!
www.ou.nl

# Acknowledgements

A big thank you goes to my family, my wife and my children, for their love and support. I appreciate your patience and understanding, as I spent many, many hours on this work, which meant less time with you. Your love and support have been very important to me. I promise that I will make up for the lost time, as this is only the beginning of a new journey.

Also, I want to thank my supervisor for never stopping believing that I could ever finish this work. Your support and encouragement meant a lot, especially when things were tough.

# Contents

# Summary

Individuals are sometimes asked to share copies of personal documents with an institution, such as a bank, an insurance company or a general practitioner's office. A convenient way to do this is to scan the document and send the resulting digital copy to the institution. However, this practice is not without risk, as the end-to-end confidentiality of the document is not always guaranteed. Despite that more and more institutions may offer secure methods for uploading documents, such as web applications and smartphone apps, email continues to be a prevalent choice [MSR+17] [BJ20]. This is concerning, as email typically lacks end-to-end encryption and, in some cases, may not even be encrypted at all, during transit and at rest on email servers.

Popular methods for digitizing documents are to use a smartphone app or Multi-Function Printer (MFP). Designated smartphone apps often fall short with regard to scan quality compared to MFPs, as they are more prone to issues like tilt, skew, and lighting distortions, which makes MFPs the preferred choice for some users. However, MFPs are typically shared devices and may be located in semi-public places, such as at a library, university campus, copy shop, etc. Individuals that use these devices to digitize documents should be concerned about the confidentiality of their documents, as they have no control over the device and the network it is connected to. For example, unprotected documents may be left on the device after use or could be intercepted during transit, when the device sends the document to the user's email address.

The individual concerns associated with using email and MFPs for document handling are significant in their own right, however, their convergence exacerbates them, as the confidentiality of the document depends on many more factors outside the user's control.

This thesis addresses the threats associated with using email and MFPs for document handling by proposing a design methodology that enables users to enforce the confidentiality of their documents for three phases of the document life cycle: digitizing, reading and sharing. The design methodology navigates the design space based on, both, the established threat model and the core requirements derived from the core interests of core stakeholders – the user, the institution and the MFP manufacturer. This aims to make the design methodology both privacy-preserving and viable.

The proposed method centers on the use of a smartphone for the storage of key material and certificates, and performing cryptographic operations within a Trusted Execution Environment (TEE). Note that smartphones are ubiquitous and inherently personal devices, that people tend to carry with them at all times, which makes them a fitting choice for storing key material.

For digitization, we propose enhancing the MFP with stream-based encryption to ensure no unencrypted data is ever stored on the device. Additionally, we suggest adding a seamless OpenPGP capability, allowing the MFP to encrypt documents using the user's public key before sending them to the user's email address. The user's OpenPGP certificate is stored and managed on the smartphone through an app, which communicates with the MFP via NFC. A key feature of this system is that the smartphone authenticates the MFP on first use by having the user verify the MFP's certificate, which is later used to verify the signature of the received email, sent by the MFP for integrity and authenticity purposes.

For sharing the document, we propose enhancing the user's Mail User Agent (MUA) with a seamless

Secure/Multipurpose Internet Mail Extensions (S/MIME) capability, which automatically discovers an S/MIME certificate associated with the entered email address, belonging to the institution, by means of a novel Well-Known Uniform Resource Identifier. The public key of this certificate is used to encrypt the document before sending it to the institution's email address, ensuring end-to-end confidentiality.

Both, reading and sharing the document, require a key operation to be performed within the smartphone's TEE, before the MUA can decrypt the document. The MUA communicates with the smartphone via a secured channel over Bluetooth Classic (BLC). The MUA and the smartphone mutually authenticate each other on first use, utilizing an out-of-bound channel.

While the proposed design is focused on email and MFPs, it is not limited to these technologies. For example, digitization using a smartphone app could *tune in* on the methodology by encrypting and submitting the document in the same way as the MFP does. Similarly, existing OpenPGP implementations could tune in on using the smartphone as a secure storage for secret key material, rather than storing them on the user's computer.

# 1

# INTRODUCTION

Whether you book a trip, apply for a loan, mortgage, an insurance, or even when you enroll for a course, you may be asked – surprisingly frequently – to share personal documents such as bank statements, diploma's, passports, credit card numbers. A convenient way to achieve this is to scan the requested document and send it to the requester. A typically overlooked downside to this is that you have converted your personal information into digital form and shared it over public channels without any constraints or safeguards. It is possible that, by sheer accident, communication happens over encrypted channels and stringent policies are in place to enforce proper handling of your private data. But you have no way of knowing this in advance. Moreover, due to how computers work, shared data is often copied. Therefore, a copy of the private document may end up unprotected in various unexpected places, such as on a hard disk inside a scanner/printer, in an inbox on a mail server, in a backup of a mail queue, etc.

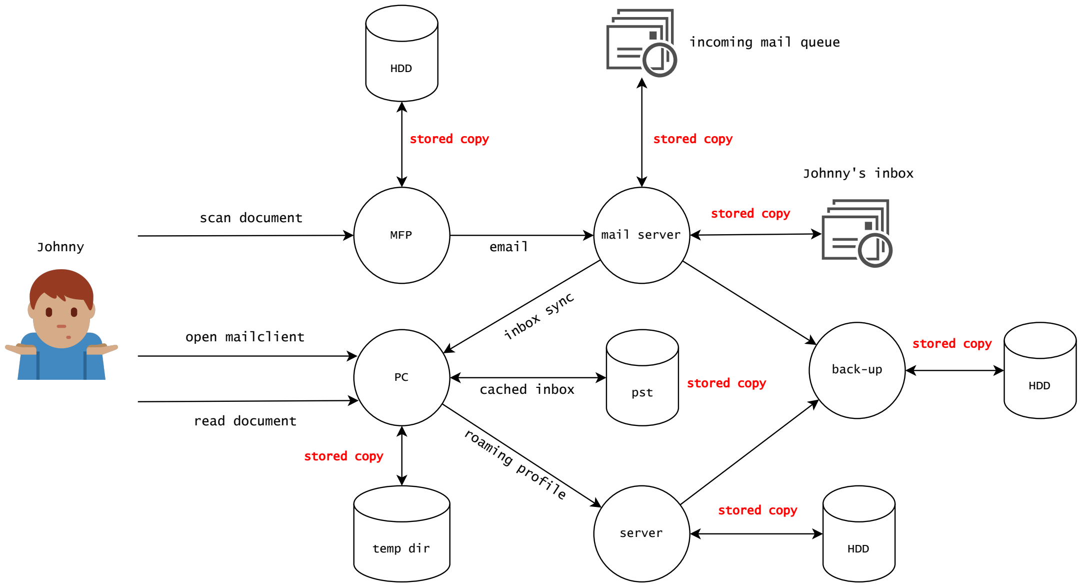


Figure 1.1: An example of how digital copies may end up in a variety of unintended locations within an organization's network.

## 1.1. Example: scanning documents using the MFP at the office

> *Consider Johnny, who is applying for a mortgage. To that end, the bank requested a copy of his passport and pay stubs. Johnny does this by scanning his passport and pay stubs with the Multi-Function Printer (MFP) at the office he works at, a small company with about 50 employees. The MFP stores a digital copy of the scanned documents on its hard disk and forwards the scan to its backend server. The MFP backend server creates an email with the scan as attachment and delivers this to the company's mail server. The mail server then sends these to Johnny's inbox. Johnny checks his email with his mail client, which downloads the mails. Then he forwards the email to the bank.*

Figure 1.1 depicts an example of locations where copies of Johnny's document may end up. This raises some concerns with regard to confidentiality:

- Copies of the document may be stored in various unintended locations, such as the MFP's hard disk, the MFP's backend server, in caches, in backups, etc. This scattering of copies across multiple locations increases the risk of unauthorized access due to the challenge of *consistent* enforcement of access control across all locations.

- Access control is typically managed by administrators within an organization. For example, if Johnny emails documents from his work account to a bank, the company's IT department may access these emails and documents, potentially allowing curious employees easy access.

- Computer hardware, including MFPs and servers, requires protection against both regular use and external maintenance with privileged access. For example, a service engineer servicing the MFP could access Johnny's documents on an MFP's hard disk.

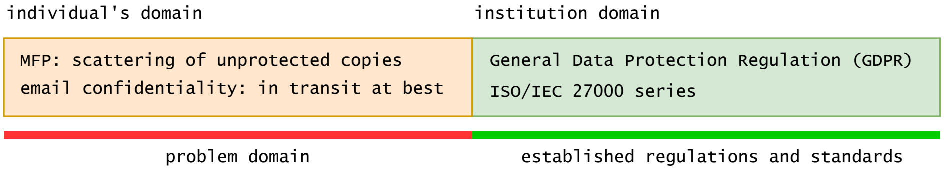
## 1.2. Next: forwarding the email to the bank

Finally, when Johnny forwards the email containing the documents to the bank, the problem is exacerbated, as the protection of the document's confidentiality depends on both, the proper configuration of Mail Transfer Agent (MTA)s used to route the email, and, the extensions installed and configured in Johnny's mail client.

**MTA configuration**
Johnny's email, en route to the bank, may pass through multiple MTAs. Inter MTA communication is typically encrypted. However, this is not a guarantee, as it depends on the configuration of the MTAs. For example, STARTTLS, a widely used extension to the Simple Mail Transfer Protocol (SMTP) protocol, can upgrade an insecure connection to a secure connection using Transport Layer Security (TLS). However, this upgrade fails if either the sending or receiving MTA does not support STARTTLS, or, for example, if a certificate has expired. In such scenarios, the email would be sent in plain text, potentially compromising the document's confidentiality, as it is exposed to anyone who has access to the network traffic. Even when the email is sent using TLS, the email is decrypted at each MTA, which increases the risk of a confidentiality breach.

**Extensions in Johnny's mail client**
End-to-end encryption can be achieved by using extensions in Johnny's mail client. For example, Secure/Multipurpose Internet Mail Extensions (S/MIME) and OpenPGP are two types of extensions that could provide end-to-end encryption, protecting the document's confidentiality and integrity from Johnny's mail client to the bank's mail client. However, these extensions are not commonly used as they are too complex for the average user to configure and use [BLO+15]. It would be safe to assume that Johnny does not use such extensions. In addition, the recipient must have a public key

Figure 1.2: Illustration of the problem domain: a gap in privacy protection between individuals and institutions, when scanning, sharing documents (the individual) and processing them (the institution).


individual's domain
institution domain
MFP: scattering of unprotected copies
email confidentiality: in transit at best
General Data Protection Regulation (GDPR)
ISO/IEC 27000 series
problem domain
established regulations and standards

available that can be used to encrypt the email, which is not always the case. Even if the recipient would provide a public key, then authentication of that key is still required, which is a challenging task in itself, as it requires verifying the key's authenticity through secure channels.

## 1.3. Privacy gap: the institutional framework versus individual practices

Figure 1.2 illustrates the problem domain. Other than individuals, institutions are bound by stringent regulatory requirements and a sense of professional ethics. They are interested in maintaining their good reputation and, therefore, ensuring compliance with laws and standards, such as the General Data Protection Regulation (GDPR) and ISO 27000-series, respectively. This institutional framework, backed by regulations and industry standards, naturally reduces the risk of privacy breaches with regard to shared documents. However, as for the end-to-end process, the individual must first digitize and submit their document to the institution. When an individual does this using a shared MFP and email, the document may be exposed to aforementioned risks. As such, there is a discrepancy between the institutional framework and the individual's modus operandi when scanning and sharing documents with regard to protecting the document's confidentiality. So, while institutions have processes in place to protect the confidentiality of documents, individuals do not, which makes them more susceptible to privacy breaches.

## 1.4. Closing the gap

In this thesis, we aim to close the identified privacy gap by proposing a method that allows individuals to digitize documents using an MFP, while maintaining their confidentiality, and subsequently share them with an institution via email, providing seamless end-to-end encryption. We intend to demonstrate that with fairly simple techniques existing systems can be augmented to protect the confidentiality of personal documents.

We aim to design our approach to align with the typical practices of individuals, which aids in ensuring that it is accessible to average users without a steep learning curve. However, it is important to emphasize that our primary focus is on demonstrating the feasibility of our approach, rather than on usability, which could be addressed in future work.

### 1.4.1. Main research question

The main research question of this thesis is:

**How to safeguard document confidentiality and integrity when individuals digitize documents using Multi-Function Printers (MFPs) and share them with institutions via email?**

**Focus on MFPs**
Our focus on MFPs stems from their status as commonly used, multi-user devices in various public and semi-public settings, including offices, libraries, internet cafés and airport lounges. Many MFPs are equipped with a scan-to-email feature, which allows the individual to easily digitize documents and have the MFP send them directly to their email address. However, the public nature of these MFPs, in combination with using them for digitizing documents, introduces significant concerns with regard to the confidentiality of the documents that are scanned and emailed.

While enterprise-level MFPs are sometimes equipped with security features, such as user authentication, encryption, and secure scanning, these features are typically not available in MFPs that are used in public or semi-public settings. One reason for this is that individuals that use these MFPs are typically anonymous, as opposed to employees in an enterprise setting, who are registered within the organization's network and, as such, can be authenticated when using the MFP.

**Alternative methods**
Note that individuals could also use their smartphone to digitize documents and email them to an institution. However, smartphones are inherently personal devices, which typically contain a vast amount of personal data. This contrasts sharply with the use of MFPs in public or semi-public settings. The distinct contexts of using personal smartphones versus public MFPs present different security and privacy challenges, that each merit a distinct research focus.

**Focus on individuals**
Our focus on individuals stems from real-world scenarios where individuals are sometimes required to digitize and share personal documents for activities like job or mortgage applications, educational enrollments or for opening bank accounts. As they are typically anonymous users of MFPs, they are not able to leverage the security features that are available to employees in an enterprise setting, which makes them more vulnerable to privacy risks. This underscores the need for our focus on individuals, although, the findings of our research are also applicable to other contexts.

**Focus on email as the method of transmission**
Some institutions provide web portals or mobile apps that individuals can use to submit documents. However, in the absence of such institution-provided means, the use of email remains a convenient method. We justify choosing email as the method of transmission for our research due to the inherent privacy risks associated with how the email system works, for example, see [PIBS21], [TFH21] on how email is transmitted in plain text over the internet.

### 1.4.2. Research questions

The following research questions aim to address the challenges of digitizing documents using Multi-Function Printers (MFPs) by anonymous users and sharing them with institutions via email, while maintaining the confidentiality of the document.

**RQ1** How to determine specific threats to the confidentiality of digitized documents when individuals digitize documents and share them with institutions via email?

This research question aims to identify threats to the confidentiality of documents that are digitized using an MFP and shared with an institution via email. The findings from this question will directly inform the design considerations with regard to RQ2 and RQ3.

**RQ2** How to enhance the process of digitizing documents with a Multi-Function Printer (MFP) executed by an anonymous user, in order to ensure the confidentiality and integrity of the document?

This research question aims to develop a design methodology that enhances the process of digitizing documents with an MFP executed by an anonymous user, in order to ensure the confidentiality of the document. Anonymous user in this context refers to a user that is not registered within the network of the organization that owns, manages or operates the MFP.

**RQ3** How to enhance the process of sending an email to an institution containing a document, executed by an anonymous user, in order to ensure the confidentiality and integrity of the document?

This research question aims to develop a design methodology that enhances the process of sending an email to an institution containing a document, executed by an anonymous user, in order to ensure the confidentiality of the document. Anonymous user in this context refers to the notion that the user cannot be authenticated by the institution based on the sender name and email address that is used to send the email.

### 1.4.3. Contributions

Our contributions are:

1. A proposed design that enables individuals to seamlessly digitize a document using an MFP to (their) email, while maintaining the document's confidentiality. The proposed design uses the individual's smartphone to store and protect secret keys.

2. A proposed design that enables individuals to forward an email containing a document to an institution, while maintaining the document's confidentiality by means of seamless end-to-end encryption. The proposed design uses the individual's smartphone to perform cryptographic operations for encrypting and decrypting the email.

3. A proof-of-concept prototype for the secure digitization to email proposed design, which further demonstrates the feasibility of the approach[1].

## 1.5. Thesis structure

The remainder of this thesis is structured as follows. Chapter 2 provides background by focusing on MFP security and Bluetooth security, which is crucial as our proposed design employs Bluetooth for secret exchanges. It also delves into the strength of encryption, a key element in our design to safeguard the confidentiality of documents stored in emails. This chapter further examines the evolving landscape of encryption, considering the impact of quantum computing and cryptanalytic developments, which are vital in understanding potential vulnerabilities over time. Note that the background chapter is not intended to be a comprehensive overview of the topics, but rather to provide context. The background chapter may be skipped by readers who are already familiar with the topics, or, who are not interested in the technical details. Subsequent chapters refer to the background chapter where necessary.

Chapter 3 reviews related work in the field of digitizing documents using MFPs and in the field of protecting the confidentiality of documents when sharing them via email. This offers a context for the research and highlighting gaps this thesis aims to fill.

[1] See `https://github.com/janouwehand/securescanning`

The methodology is detailed in chapter 4, which explains the approach to addressing the aforementioned research questions, and discussing the reliability, generalizability, and limitations of the study.

Chapters 5 analyzes the typical process steps for digitizing and sharing documents, respectively, which establishes a basis for the threat model in chapter 6. These chapters dissect the current practices and identify potential security risks.

Chapter shifts focus to the core requirements for coming up with a design that, both, effectively mitigates the identified threats, and is feasible, as it takes into account the interests of the core stakeholders: the individual, the institution, and the MFP manufacturer.

The design methodology for secure digitization, secure reading, and secure sharing is presented in chapter 8, 9, and 10, respectively. Each chapter discusses various methods that could be used to enhance the document's confidentiality at a specific stage and proposes a design that aligns with the core requirements. At the end of each chapter a process overview and risk mitigation assessment is provided, which details how the proposed design addresses the identified threats of the baseline threat model. Chapter 11 discusses the newly introduced threats of the proposed design.

The thesis concludes with Chapter 12, summarizing the methodology, results, contributions, and generalizability of the study. It also suggests avenues for future research.

# 2

# Background

This chapter elaborates on background topics of the proposed design. First, we start by providing an explanation of frequently used terms throughout this thesis. Next, we give a brief introduction on MFPs and the state of security in MFPs, which provides context for the proposed design. Then, we discuss Bluetooth security, as the proposed design uses Bluetooth to transfer secrets between the user's Mail User Agent (MUA) and the smartphone, it is important to consider the security of the Bluetooth communication channel. Finally, we discuss developments in the field of quantum computing, as the proposed design encrypts the digital document and stores it for a long period of time, which makes the strength of the encryption important to consider.

## 2.1. Frequently used terms

This section provides definitions and explanations of key terms used throughout this thesis. Clarifying these terms and their meaning within the context of this thesis aids in the reader's comprehension of the proposed design.

**Digitization**. Refers to the process of converting a physical document into a digital document.

**DKIM**. DomainKeys Identified Mail (DKIM) is a standard that associates a domain with the email message by affixing a digital signature. The MSA adds a DKIM signature to the email message, which is, ultimately, verified by the MDA. DKIM provides a mechanism for, both, authentication and integrity validation of the email message. For authentication purposes, the recipient can verify that the email message was indeed sent by someone that has demonstrated control over the domain, as the DKIM signature is generated using the private key belonging to the published, public key in the DNS record of the domain for DKIM. For integrity validation purposes, the recipient can verify that the email message has not been tampered with, as the DKIM signature is generated using a hash of the email message.

**MTA**. A Mail Transfer Agent (MTA) is a component that transfers email messages from one MTA to another. It routes the email to its destination, ensuring it gets to the correct server and mailbox. Note that the term MTA generalizes specific roles within the email system. While an MTA may refer to an intermediate mail server, it may also refer to:

a **Mail Submission Agent (MSA)**, which is responsible for email admission, checking the email for spam and viruses, and, policies and rules, adding email headers, such as the DKIM signature, and, subsequently, passing it on.

a **Mail Delivery Agent (MDA)**, which is responsible for the final delivery of the email to the recipient's mailbox, and may also be responsible for the storage of the email. The MDA verifies whether the DKIM signature is valid, indicating that the email has not been tampered with.

**MUA**. A Mail User Agent (MUA) is an application that allows the user to access, manage, and send out their email. Within the context of this thesis, the MUA refers to the desktop application on the user's PC that is used to receive the email with the digital document originating from the MFP, and, subsequently, to forward the email (with the digital document) to the institution. It is important to note that, mostly, the term MUA in context of this thesis actually refers to the MUA add-in.

**MUA add-in**. Refers to a software component that can be installed on within the user's MUA to extend its functionality. Within the context of this thesis, the add-in extends the functionality of the MUA to facilitate the key exchange between the MUA and the smartphone over Bluetooth, automatic discovery of the recipient's Secure/Multipurpose Internet Mail Extensions (S/MIME) certificate based on the entered email address, and, the encryption of the digital document using the recipient's S/MIME certificate.

**Scanning**. Refers to digitization.

**Secure scanning**. Refers to the process of digitization using a Multi-Function Printer (MFP) while protecting the confidentiality and integrity of the digital document.

## 2.2. Multi-Function Printer (MFP) security

MFPs exist in many forms and sizes, ranging from small desktop devices to large industrial machines. They are used to print, copy, scan and fax documents, although the latter is becoming less common. The scanning function is used to convert paper documents into digital documents.

Botha et al. [BS18] provide an overview of security threats and countermeasures for MFPs. They state that MFPs are standalone devices that are connected to a network and run popular services, like Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), Server Message Block (SMB), and SMTP, and, as such, are vulnerable to the same threats as other networked devices. They conclude that MFPs are often overlooked in security audits, that MFPs are often not updated with the latest security patches and that MFPs are often not configured securely. The countermeasures they propose include selecting an MFP that supports security features, such as full disk encryption, support for Advanced Encryption Standard (AES) encryption and functionality to overwrite data on the hard disk. They also propose to configure the MFP securely with regard to authentication and authorization, disabling certain protocols, such as Telnet, updating the firmware, and using secure protocols only.

The countermeasures that Botha et al. propose may be effective, but are complex and not trivial to execute or implement. Note that MFPs are used in various environments, such as offices, schools, hospitals, libraries, airport lounges and government institutions, ranging from small to large organizations. Especially in small and medium-sized organizations, where there is often no dedicated IT staff and formalized processes [HGB19], it is unlikely that these countermeasures are implemented.

The type of documents that individuals scan are often sensitive in nature. Examples include a copy of a passport or a bank statement. Therefore, the lack of security in MFPs is a threat to the privacy of individuals.

A recent survey conducted by Quocirca amongst 507 IT decision-makers in small, medium size and large businesses in the USA and Europe [Quo23] shows that print security is lower on the security agenda than other elements of IT infrastructure, cybersecurity incidents continue to rise, organizations are finding it harder to keep up with print security demands, and, less than one-third are very satisfied with their print supplier's security capabilities. Given these points, it appears that the security of MFPs is often not a high priority for many organizations. This indicates a clear need for the development of more robust security solutions that are straightforward to implement and use.

## 2.3. Bluetooth security

The Bluetooth standard is maintained by the Bluetooth Special Interest Group (SIG) and is publicly available[1]. Bluetooth supports two main protocols: Bluetooth Low Energy (BLE) and Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR), also called Bluetooth Classic (BLC). BLC is the original Bluetooth protocol and is used for high-throughput applications, such as streaming audio, whereas BLE is a newer protocol that is optimized for low-power applications, such as wearable devices and sensors.

As for the transfer of secrets via BLE, it is important to consider the confidentiality of the communication channel. BLE supports encrypted communication out of the box by means of the Simple Secure Pairing (SSP) protocol, which is used for pairing devices and establishing a secure channel between them. However, there are security concerns related to the SSP protocol. Sun et al. describe a threat scenario in which an adversary can eavesdrop on the pairing process and obtain the encryption key [SMS18], which can then be used to decrypt the communication between the two devices. In their survey paper on BLE security and privacy, Cäsar et al. state that BLE security is complex due to the overwhelming number of possible configurations [CPST22]. They conclude that a secure BLE device could be achieved by making use of all the security mechanisms provided by the specification in its most current version. They advise considering complementing BLE security with an application level security layer when BLE devices are used in conjunction with mobile Operating System (OS), such as Android and iOS.

Note that it can be expected that many smartphones currently in use do not support the most recent version of the BLE specification [OAAA23]. Then, in the absence of an application level security layer, the confidentiality of the BLE communication channel can become at risk.

Cäsar et al. mention that BLE provides several security mechanisms that can mitigate Man-In-The-Middle (MITM) attacks, which varies amongst Bluetooth versions and implementations [CPST22]. However, they also state that these mechanisms are not always enabled by default, nor, when one of the devices has insufficient capabilities. Note that confidential communication necessitates preventing MITM attacks, as an adversary can eavesdrop and alter the communication between the two devices. Given the variability of Bluetooth versions and implementations on smartphones 'in use', it appears that the confidentiality of BLE communication cannot be assumed.

Cäsar et al.'s advice to consider complementing BLE security with an application level security layer is in line with the recent findings of Antonioli [Ant23] who demonstrates how two novel vulnerabilities (BLUFFS) in the Bluetooth architecture can be exploited to accomplish device impersonation and MITM, regardless of the Bluetooth hardware and software versions.

Finally, the rule economy of mechanism states that (security) systems should be as simple as possible to minimize potential vulnerabilities. However, in this particular context, we argue that an application

[1] https://www.bluetooth.com/learn-about-bluetooth/tech-overview/

level security layer is justified, as it provides an additional, necessary layer of protection against evolving and sophisticated threats that may bypass existing BLE security measures.

## 2.4. Strength of encryption

Since our proposed design encrypts digital documents that end up in mailboxes for a potentially indefinite period of time, it is important to consider the strength of the encryption, as those emails may become easy targets for cryptanalytic attacks in the future. For example, popular email service providers, like GMail [Goo23a] and Outlook.com, typically store emails indefinitely by default, until they are manually deleted by the user. The strength of the employed encryption must be appropriate given this context. It is important to note that encryption algorithms and parameters (key size, for example) that are considered secure today may not be secure 5, 10 or 20 years from now. With regard to time, the most significant threats include cryptanalytic developments, the advent of quantum computing and increasing computational power of (classical) computers.

### 2.4.1. Cryptanalytic developments

A significant breakthrough in cryptanalysis could potentially render a currently secure encryption algorithm insecure. This risk can be mitigated by using a well-established encryption algorithm that has undergone extensive scrutiny by the cryptographic community. However, even well-established encryption algorithms may become vulnerable to cryptanalytic attacks in the future. Regular updates and adaptations in response to emerging cryptanalytic advancements are necessary to ensure that the encryption algorithm remains secure.

### 2.4.2. Increasing computational power

A familiar phenomenon is that computational power of classical computers increases over time. This poses a significant threat to the security of ciphertext that is stored for a long period of time, such as encrypted emails in mailboxes. When processors become more powerful, they become capable of executing cryptanalytic attacks more efficiently, which reduces the time required to break encryption. This risk could be mitigated by using an encryption algorithm with a sufficiently large key size, which is considered secure for the foreseeable future. For instance, while RSA-2048 may be considered secure today, it is expected that RSA-4096 will remain secure for a longer period of time, with regard to increasing computational power. As such, key size can be considered a measure of future-proofing encryption against increasing computational power of classical computers.

### 2.4.3. The advent of quantum computing

Quantum computing is a rapidly developing field that aims to use quantum mechanical phenomena to perform computations. This field is expected to have a significant impact on the security of encryption algorithms.

**Asymmetric cryptography**
In the (near) future it is expected that quantum computing is fast and stable enough to run Shor's algorithm [Sho97] to break cryptographic algorithms that rely on the discrete logarithmic problem or the integer factorization problem, which include RSA (Rivest-Shamir-Adleman), Diffie-Hellman key exchange (DH), Elliptic Curve Cryptography (ECC) and Digital Signature Algorithm (DSA). In 2021, Gidney et al. estimated that a quantum computer needs around 20 million physical qubits to be able to break RSA-2048 [GE21]. That is far away from the current largest quantum computer, named the Osprey with (just) 433 qubits which was introduced in November 2022 by IBM [IBM22]. However, science into quantum computing seems to be well funded [IBM23] [UoC23] [Kyr23], which

may accelerate advancements in the field.

**Symmetric cryptography**

With regard to symmetric encryption NIST's report NIST.IR.8319 [MD21] states that AES128, 192 and 256 are considered to still be secure in the foreseeable future. However, the report also states that in the advent of quantum computing AES192 and in particular AES256 offer significant better protection with regard to Grover's algorithm [Gro96]. With regard to symmetric cryptanalysis, Grover's algorithm offers quadratic speedup, meaning that N-steps are reduced to sqrt(N) steps, and, therefore, ultimately reducing the effectiveness of the AES key size in half. As such, AES256 offers significantly better protection compared to AES128, as the effectiveness of the key size would be reduced to respectively 128 and 64 bits using Grover's algorithm. This would justify using AES with a 256 bit size for encrypting the digital document, today.

**Post-quantum cryptography**

Post-quantum cryptography is a field that aims to develop cryptographic algorithms that are secure against quantum computers. NIST is currently running a competition-like process that involves the cryptographic community to standardize post-quantum cryptography. The status report on the third round of the NIST post-quantum cryptography standardization process [ACD+22] states that the public-key encryption and key-establishment algorithm that will be standardized is CRYSTALS-KYBER [BDK+18] and that the digital signature algorithm that will be standardized are CRYSTALS-Dilithium [DKL+18], FALCON [SBN+21], and SPHINCS+ [BHK+19].

**Bottom line**

Taking into account the projected future advancements of quantum computing, it is important to consider the following:

- **Symmetric encryption**: Until NIST standardizes a post-quantum symmetric encryption algorithm, NIST recommends employing AES-256 in Galois/Counter Mode (GCM) for current implementations, as current understanding suggests that AES will remain secure against quantum attacks, albeit with increased key size. Moreover, GCM is recommended by NIST[2] for authenticated encryption. Note that authenticated encryption incorporates integrity checks within the encryption process, therefore, obviating the need for a separate Message Authentication Code (MAC), which makes the process more efficient. This added efficiency is relevant, as MFPs typically have limited resources, such as CPU power and volatile memory.

- **Asymmetric encryption**: Special consideration is required for the long-term storage of asymmetric ciphertext, as it is expected that quantum computers eventually will break algorithms that rely on the discrete logarithmic problem or the integer factorization problem, which include RSA, Diffie-Hellman key exchange (DH), Elliptic Curve Cryptography (ECC) and Digital Signature Algorithm (DSA). This is a threat to the long-term confidentiality of the digital document. Note that standards like S/MIME and OpenPGP utilize asymmetric encryption algorithms to encrypt the symmetric encryption key, which is embedded in the email. Therefore, when these standards are used, it is not possible to refrain from using asymmetric encryption for long-term encryption, as emails are typically stored indefinitely. Those emails will inevitably become vulnerable to cryptanalytic attacks when quantum computers become widespread. In this context it could be considered good practice to keep a good hygiene of the email archive, by deleting emails that are no longer relevant.

[2] https://csrc.nist.gov/publications/detail/sp/800-38d/final

# 3

# Related work

This related work chapter is structured in two sections. The first section discusses related work that improves the confidentiality of documents being digitized, while the second section discusses work that improves the confidentiality of documents being shared via email.

## 3.1. Digitization

Our search yielded no work that directly addresses improving the confidentiality of digitizing documents using MFPs. Existing studies primarily focus on the security aspects of multifunction devices in general, for example, [BS18] and [ADFS+23], and address threats and mitigation strategies for MFPs, see also section 2.2.

MFP manufacturers have developed their own proprietary solutions for securing the digitization process, that protects the confidentiality of the digitized documents. These solutions are typically targeted at corporations and integrate with the manufacturer's could environment and the corporation's existing security infrastructure. Examples of MFPs that have such solutions include the Canon imageRUNNER ADVANCE DX series [Can23], HP models that support Access Control (AC) Scan (PRO) [HPA23] and Xerox models supporting ConnectKey [Xer23]. These solutions are vendor-specific, typically not available to (anonymous) individuals and may require the use of the vendor's cloud environment. In contrast, our methodology is available to individuals that utilize a semi-public MFP, does not require the use of a cloud environment and has the potential to become vendor-agnostic.

## 3.2. Sharing: encrypting emails

In their paper 'Why Johnny can't encrypt' [WT99], Whitten and Tygar argue that email encryption is not widely adopted, because it is too difficult to use for the average user, to manage the keys and certificates required for email encryption. They took PGP 5 as an example, which was a popular email encryption program at the time. In 'Why Johnny still can't encrypt' [RAZS15], the study is revisited, this time using PGP 9, which is a more recent version of PGP. As opposed to Whitten and Tygar, the revisited article solely focuses on the usability of PGP 9, and does not discuss causes for their findings. While their findings indicate that users still struggle with encrypted email (using PGP 9), they indirectly attribute this to inadequacies in the user interface of PGP 9, as they propose various improvements to the user interface. Benenson et al. [BLO+15] aim to shed a light on whether Johnny would ever be able to encrypt by measuring human capacity for security tasks. They conclude that some typical tasks necessary for email encryption may be beyond the capacity of the average user, and, as such, Johnny may never be able to encrypt. Although, that is as long as the user is required to manage keys and certificates themselves.

PostGuard (formerly known as IRMAseal) is a novel approach for securing email communication by encrypting emails, based on Identity-Based Encryption (IBE) [BBJ+23] by leveraging the IRMA identity platform [AvdBH+17] as a foundation. This converts the typical public key management problem into an identity management problem. The identity problem is solved using a Trusted Third-Party (TTP), combined with an identity wallet stored on the user's device, for holding the user's private keys. PostGuard's focus is on usability by effectively hiding the complexities of authentication and key management from the user. This is accomplished by means of Yivi (Your Identity Vault Interface) – formerly known as IRMA, a user-friendly app that allows users to manage their identities, rather than their keys, and, as such, abstracts away the complexities of key management to the level that 'Johnny actually can encrypt' [Sta22b].

Relating to our context, note that PostGuard could be a viable solution to secure both the email communication between the MFP and the individual, as well as between the individual and the institution. However, also note that using a TTP could be considered a single point of failure, as the TTP is responsible for managing the identities of all users, and when it is compromised, all users could be affected. Our methodology does not require the use of an additional TTP for key or identity management – apart from the TTPs that are already in place, such as the CA that issues TLS certificates for websites and email servers. Moreover, while PostGuard uses a distinct, custom protocol for encrypted emails, our methodology uses OpenPGP, providing the user with the flexibility to move to (or from) our methodology, while being able to read past (and future) OpenPGP encrypted emails.

ProtonMail [Kob18] is a secure email service that allows users to send encrypted emails to other ProtonMail users, which under the hood uses OpenPGP. ProtonMail's (zero-trust) design prioritizes user-friendliness over flexibility by abstracting away the complexities of key management to the level that 'even Johnny could encrypt'. Although, that is as long as the recipient is also using ProtonMail. Sending encrypted emails to non-ProtonMail users, for example, when sending a personal document to an institution, is only indirectly supported. ProtonMail facilitates this by allowing users to send a password-protected email to non-ProtonMail users, which contains a link to a ProtonMail web server, where the recipient can enter the password to read the email. This is not ideal, as the password has to be communicated out-of-band. Note that this functionality could effectively be achieved by zipping a document with a password, and sending it as an attachment to an email. This has the same disadvantages, as the password has to be communicated out-of-band. As such, ProtonMail's approach does not solve the problem of securely sending personal documents to institutions in a hassle-free, convenient way for both the sender and the recipient.

Much like ProtonMail, our methodology also abstracts away the complexities of key management, although being it without the need to manually exchange a password. Instead, our methodology uses a discovery mechanism to automatically retrieve the institution's public key, which is then used to encrypt the email.

# 4

# METHODOLOGY

In this chapter we outline the approaches and methods employed in our research.

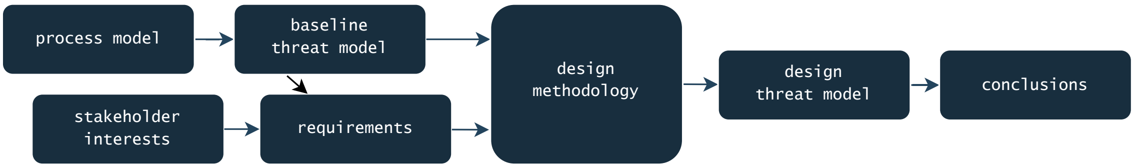


Figure 4.1: Overview of systematic approach.

We adopt a systematic approach, starting with establishing a process model that describes the typical process steps involved in digitizing a document using an MFP and sharing it via email with an institution. Based on this process model we establish a baseline threat model that identifies the threats to the confidentiality of documents, using the STRIDE threat model [KMLS17] for structuring the threats and the DREAD risk assessment model [OWA23] for assessing the risks associated with the threats. Parallel to the threat model, we identify core stakeholders and their typical interests with regard to the context of digitizing and sharing documents. Informed by, both, the baseline threat model and core stakeholder interests, we analyze what core requirements a privacy-enhancing system, that protects the confidentiality and integrity of documents, should meet. Next, we develop a system design methodology, informed by the identified threats and core requirements, for each process: digitizing documents, reading them on a computer and sharing them via email with an institution, with the goal of protecting the confidentiality and integrity of the documents at each stage. Each design methodology evaluates multiple methods and techniques, selecting and justifying the most suitable one in relation to the identified threats and core requirements and ends with a process model and an assessment of how the design methodology mitigates the identified threats. The provided process model combines the selected methods and techniques into a coherent process model, which demonstrates the feasibility of the design methodology. After developing the design methodologies, we conduct a new threat assessment, targeted at identifying new threats that may emerge from the design methodologies. Finally, we assess the efficacy of the proposed design methodology by comparing the baseline threat model to the threat model of the design methodology.

## 4.1. Approach to research questions

The methodology closely aligns with the research questions, which offers a focused approach to each. With regard to **RQ1** it establishes a comprehensive threat model, which lays the groundwork for targeted design strategies in **RQ2** and **RQ3**. In addressing **RQ2** and **RQ3**, the methodology evolves into three design methodologies for each process: digitizing documents, sharing them via email with an institution, and processing them at the institution.

## 4.2. Reliability

The reliability of our research is safeguarded by the systematic approach, which ensures that the research is conducted in a consistent and repeatable manner. This is achieved by taking the typical processes and stakeholders into account, which ensures that the research is grounded in reality. Reliability is further enhanced by the iterative process of threat modeling and re-assessment after each design methodology. This ensures that the design methodologies are informed by the identified threats and core requirements, while also ensuring that they are effective in mitigating the identified threats. In addition, each design methodology is ends with a process model that demonstrates the feasibility of the design methodology, which further enhances the reliability by providing a concrete example of how the results can be applied in practice. However, it should be noted that coming up with a design is a creative process, which is inherently subjective. As such, the reliability of the design methodologies is limited by the subjective nature of the design process, as the selection of methods and techniques is partly based on our preferences and experience, which may differ from those of other researchers.

## 4.3. Limitations

The scope and the time frame of this research necessitate a trade-off between depth and breadth. This trade-off is reflected in the following limitations:

- First, the core stakeholder and core requirement analysis display a limited scope, as it only considers the (obvious) interests of the individual, the institution and the MFP manufacturer. Other stakeholders, such as standardization organizations, regulatory bodies, auditors and software developers, are not considered.

- Second, the design methodologies focus on a specific context, namely the digitization of documents using an MFP and sharing them via email with an institution. Other contexts, such as mobile devices, are not considered.

- Third, the design methodologies exhibit a bias towards well-established standards like OpenPGP and S/MIME. While this preference is partly justified by suitability for email communication, it may limit the exploration of alternative, potentially innovative solutions.

- Fourth, our scope is limited to end-to-end encryption of documents between an individual and an institution. Once the document is received by the institution, protecting the document's confidentiality – and with that, the individual's privacy – is at the discretion of the institution. We justify this limitation by arguing that institutions are bound by laws and regulations, which ensure that the institution handles the document in a privacy-preserving manner.

Aside from these limitations, we argue that the research is sufficiently detailed to serve as a basis for a proof of concept implementation, which we provide in the form of a prototype for secure digitization.

## 4.4. Generalizability

Although the generalizability of our study is somewhat constrained by the specific focus on individuals, MFPs, email and institutions, the notion of incorporating smartphones as holders of OpenPGP private keys, used as a multifactor authentication mechanism, introduces an element of generalizability, as this approach could be applied to other contexts. In addition, the concept of automatic authenticated certificate discovery by means of a Well-Known Uniform Resource Identifier could also be applied to other contexts, for example, in corporate settings where workers could seamlessly exchange end-to-end encrypted emails with other organizations.

# 5

# DIGITIZATION AND SHARING PROCESS ANALYSIS

This chapter gives an overview of the typical steps involved in digitizing and sharing documents with an institution. These steps form the basis for analyzing which threats exist in the following chapter.

## 5.1. DIGITIZATION

With regard to our context a user creates a digital document by digitizing a physical document using an MFP which submits an email containing the digital document to the user's email address. Note that MFPs may be situated in a public space, such as a library, or in a private space, such as a home, and, that as such, there is not always a user directory available. Our model assumes that the user is anonymous and enters their email address manually. Figure 5.1 illustrates this process, while figure 5.2 illustrates the steps involved in digitizing a document. Next follows a description of each step. Each step number is prefixed with 'dig' to distinguish steps amongst stages.

### Digitization process steps

| | |
|---|---|
| **dig-1** | The user positions the physical document on the MFP. |
| **dig-2** | The user instructs the MFP to initiate the scan-to-email function. |
| **dig-3** | The user enters their email address. It is important to note that in the absence of a user directory, the user must |

Figure 5.1: Illustration of the digitization process.

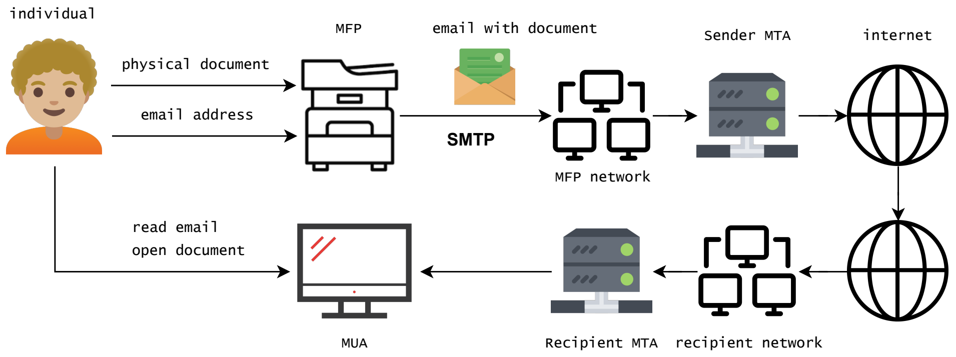
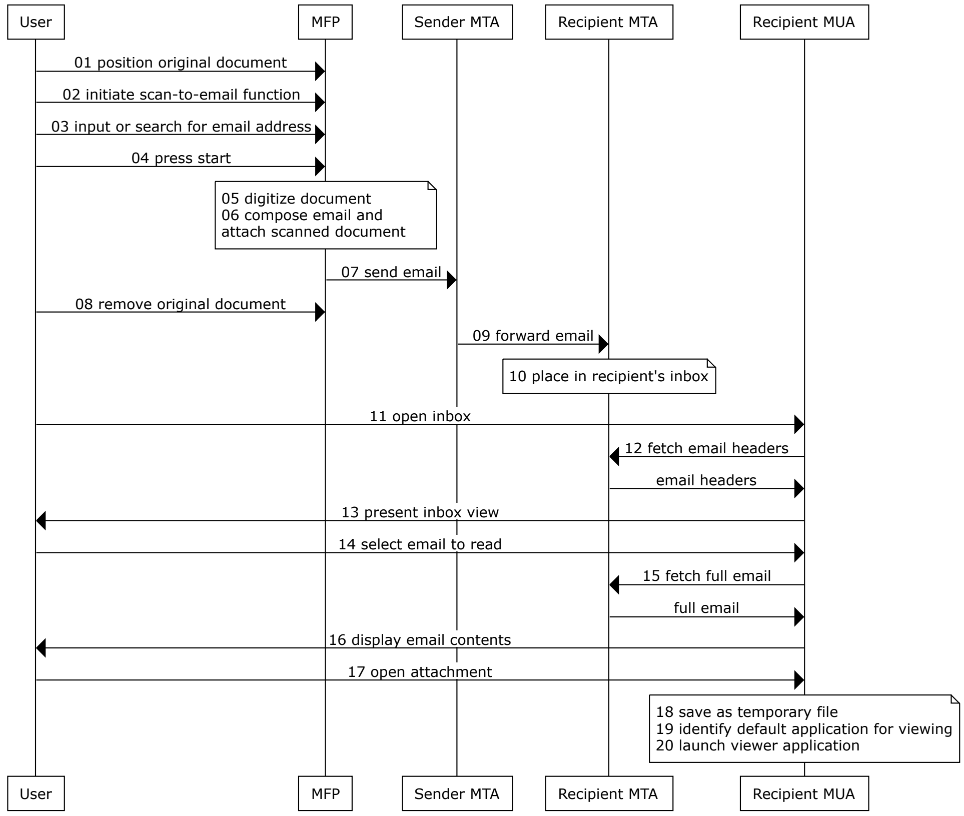


Figure 5.2: Typical process steps of digitization to email using an MFP (dig-*).



enter their email address manually.

**dig-4** The user confirms the settings and presses start to initiate digitization.

**dig-5** The MFP digitizes the document. The MFP may store a copy of the digital document on its hard disk.

**dig-6** The MFP composes an email directed to the user entered email address. The document is attached to the email.

**dig-7** The MFP submits the email to the sender MTA which accepts outgoing email messages for the MFP The MTA ensures that the email complies with the necessary format and protocols and contains all required headers. Note that the MFP acts as the sender MUA in this case.

**dig-8** The user removes the original document from the MFP.

**dig-9** The sender MTA determines the best path for the email to reach its destination, which typically involves a DNS lookup to find the recipient MTA. Then the MSA forwards the email to the discovered recipient MTA.

**dig-10** The recipient MTA stores the email message in the recipient's inbox.

**dig-11** Using the MUA, the user opens their inbox.

**dig-12** The MUA fetches the mail headers from the MDA. This is typically done via the POP or IMAP protocol.

**dig-13** The MUA presents the email messages in the user's inbox to the user.

**dig-14** The user opens the email messages that contains the document as an attachment.

**dig-15** The MUA fetches the full email message from the MDA.

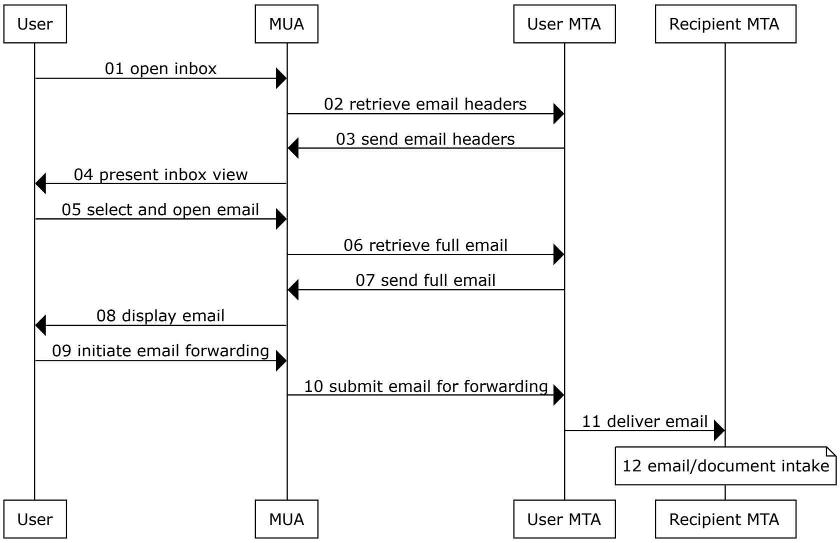
**dig-16** The MUA presents the full email message to the user, including the names of the attachments.

**dig-17** Using the MUA, the user opens the attachment.

**dig-18** The MUA creates a temporary file on the user's hard disk.

**dig-19** Using the operating system, the MUA looks up the viewer application that is associated with the document's MIME type. For example, when the document has the application/pdf MIME type, the MUA may find Acrobat

Figure 5.3: Typical steps involved in forwarding the email to an institution (shr-*).



Reader as the associated viewer application.

**dig-20** The MUA starts the viewer application giving the URI of the temporarily stored file as a parameter.

## 5.2. SHARING

After the user has digitized the document and has received the email containing the digital document, the user forwards it to the institution. Figure 5.3 illustrates the steps involved in forwarding the email to the institution. Next follows a brief description of each step.

### Sharing process steps:

**shr-1** The user opens the inbox using the Mail User Agent (MUA) to manage emails.

**shr-2** The MUA sends a request to the serving MTA to retrieve the list of email headers, which summarize the incoming emails without downloading the full contents of the emails.

**shr-3** The MTA sends the email headers to the MUA, which allows the user to see a snapshot of each email which typically consists out of date and time, sender and subject.

**shr-4** The MUA processes the headers and presents the inbox view to the user, displaying sender information and email subjects.

**shr-5** The user browses the inbox and selects an email, which instructs the MUA to open it to read its contents.

**shr-6** Upon user's request the MUA contacts the serving MTA to retrieve the full content of the selected email message.

**shr-7** The MTA retrieves the full email message from the storage and sends it to the MUA.

**shr-8** The MUA receives the full email and displays the content, including all text and the names of the attachments, to the user.

**shr-9** The user forwards the email message by initiating the forwarding process using the MUA's interface.

**shr-10** The MUA packages the email and submits it to the MTA, to be sent to the institution.

**shr-11** The MUA serving MTA forwards the email to the recipient's MTA.

**shr-12** The recipient's MTA receives the email and initiates the email and document intake process, which could involve virus scanning and spam filtering, and, ultimately storing the document in the institution's Document Management System (DMS) and creating a work item in the institution's workflow system.

# 6

# Baseline threat model

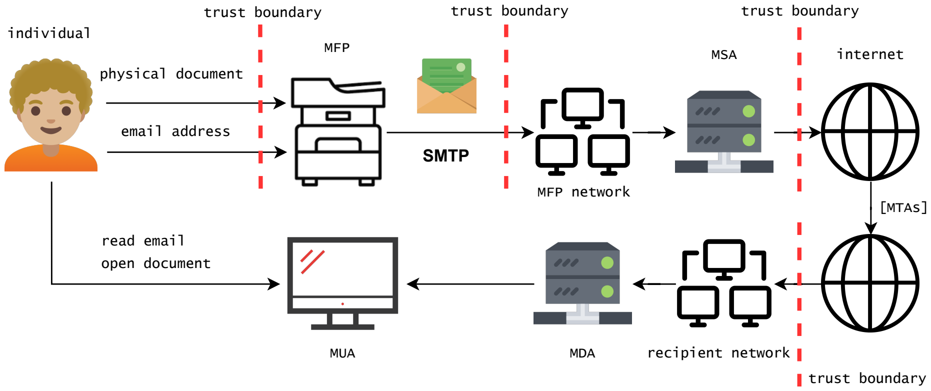
This chapter identifies and analyzes threats within the document life cycle that could compromise the document's confidentiality. The threats and vulnerabilities have been identified through a threat modeling exercise using the STRIDE methodology [**?**] on the process steps described in the preceding chapter.

We make a distinction between threats and vulnerabilities. A threat is a potential danger that could exploit a vulnerability. A vulnerability is a weakness in the system that could be exploited by a threat. For example, a vulnerability could be a weak password, which could be exploited by a threat, such as a brute-force attack.

## 6.1. Digitization

Figure 6.1 shows the trust boundaries for the digitization process, which demarcate the boundaries of where the trust level changes. First there is the user that willfully digitizes the document using the MFP to have the MFP send the document to their own email address. The first trust boundary is between the user and the MFP, because the user has to trust the MFP to not leak the document. The second trust boundary is between the MFP and the Mail Submission Agent (MSA), which is the server that receives the email from the MFP. Here, the MFP places trust on the MSA to securely handle and protect the confidentiality of the document. The third trust boundary is between the MSA and the MTA of the user's email provider which receives the email from the MSA over the internet. Here,

Figure 6.1: Secure document scanning and email via NFC and email with trust boundaries.



the MSA places trust on the MTA to protect the confidentiality of the document. The fourth trust boundary is between the MTA and the Mail Delivery Agent (MDA), which is the server that stores the email for the user in their mailbox.

The significance of these trust boundaries is that the user has to trust the MFP, the MFP has to trust the MSA, the MSA has to trust the MTA and the MTA has to trust the MDA. This chain of trust actually implies that *the user* has to trust the MFP, the MSA, the MTA and the MDA to not leak the document, which underscores the importance of robust security measures at each stage of the document transmission process.

For clarity, MSA, MTA, and MDA, in spite of their individual roles in the email infrastructure, will be collectively referred to as the MTA for the remainder of this chapter.

### 6.1.1. Entering an email address on the MFP

In step **dig-3** the user manually enters their email address on the MFP using the touchscreen or keypad.

#### Spoofing

**risk 1 The threat that a (malicious) user intentionally spoofs another user's email address.**

A user could intentionally impersonate another user by entering the other user's email address. While intentional spoofing is a threat, its impact on confidentiality may be limited in our context. This is because the malicious actor already has access to the physical document being digitized. However, it remains a threat in terms of preventing misuse.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | Low | Intentional spoofing could lead to a malicious user deliberately sending a document of choice to another user, which could lead to apparent non-repudiation. For example, the document could contain incriminating information that the malicious user wants to associate with the other user. |
| *Reproducibility* | High | It is easy to spoof an email address, since the malicious actor would just have to enter the email address of choice. |
| *Exploitability* | High | It is highly exploitable, as it is easy to spoof an email address due to the anonymous nature of the user operating the MFP. |
| *Affected users* | Low | Low, as it is a directed attack on a specific user (email address). |
| *Discoverability* | Medium | It is easy to discover. As the MFP asks the user to enter an email address, it is easy to discover that the MFP would accept any email address. |

**Conclusion risk 1**
While the threat of intentional email address spoofing is present and easy to exploit, its overall risk is moderated due to the low potential for damage and the fact that it targets specific users. This limits its broader impact.

Finally, note that some MFPs let the user first verify their email address before sending the document, which mitigates the risk of intentional spoofing. In that case, the user would receive a verification email from the MFP, which they would have to confirm, for example, by clicking on a link in the email, or, by entering a code that is included in the email. As the user entered an email address not of their own, they would not receive the verification email, which would prevent the document from being sent, effectively mitigating the risk of intentional spoofing.

**risk 2 The threat that a user accidentally misspells their email address, which could lead to information disclosure and loss of information.**

A user could unintentionally enter the wrong email address, which could lead to the wrong user receiving the document, comprising the confidentiality of the document. At the same time it could lead to the intended recipient not receiving the document, which leads to a loss of information.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | Unintentional spoofing could lead to the wrong user receiving the document, comprising the confidentiality of the document. |
| *Reproducibility* | n/a | Not applicable, as it is not an attack, but a vulnerability. |
| *Exploitability* | n/a | Not applicable, as it is not an attack, but a vulnerability.supervision. |
| *Affected users* | High | It affects the user that digitizes the document by compromising the confidentiality of the document. |
| *Discoverability* | n/a | Not applicable, as it is not an attack, but a vulnerability. |

### Conclusion risk 2

Misspelling the email address is a vulnerability that could lead to a loss of confidentiality. As such, it is a risk that should be mitigated. Note that entering an email address is a challenging task that may pose difficulties for some users, which exacerbates the risk of unintentional spoofing.

Finally, note that some MFPs let the user first verify their email address before sending the document, which mitigates the risk of unintentional spoofing. However, this extra step may be inconvenient for users seeking a quick way to digitize and send documents. While it enhances security, this step can add time and complexity to what is expected to be a straightforward task.

### Tampering

As tampering refers to the risk of unauthorized modification of data, it is not directly applicable to this scenario of a user entering their email address on the MFP.

### Repudiation

**risk 3 The threat that a (malicious) user denies having typed the email address that was entered on the MFP.**

A user could deny having entered 'that' email address on the MFP, effectively repudiating their action. This could be a deliberate action by a malicious user, or an unintentional action by a user that made a mistake. The fact that anonymous users can operate the MFP makes it difficult to trace the user that entered the email address.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | Low | It could lead to a malicious user deliberately sending a document of choice to another user, which could lead to apparent non-repudiation, refer **risk 1**. |
| *Reproducibility* | High | Without traceability, it is easy to deny having entered the email address. |
| *Exploitability* | Medium | It is easy to exploit, as it is easy to deny having entered the email address when there is no traceability. |
| *Affected users* | Low | Low, as it is a directed attack on a specific user (email address). |
| *Discoverability* | n/a | Not applicable, as it is obvious that a user can easily deny having entered the email address. |

**Conclusion risk 3**
The threat of email repudiation in the context of a user denying having entered a specific email address, is a significant concern as it is easy to exploit. This issue is exacerbated by the anonymity of users, which makes it challenging to trace who entered the email address.

### Information disclosure

Information disclosure is discussed in the context of a user accidentally misspelling their email address, refer **risk 2**.

### Denial of service

**risk 4** **The threat that a malicious user repeatedly enters a non-existent email address on the MFP, which could lead to the associated MSA being flagged for spamming.**

The threat involves a malicious user that repeatedly enters an invalid email addresses on an MFP, which could lead to the MSA being flagged for spamming. Note that this behavior mimics spamming activities that could ultimately cause intermediate email service providers (MTAs) to blacklist the MSA. As such, this behavior could potentially lead to a denial of service for the MSA, which could affect other users that use the MSA to send emails.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | It could lead to the MSA being flagged for spamming, which could lead to a denial of service for the MSA. |
| *Reproducibility* | Low | It is difficult to reproduce, as it requires a malicious user to repeatedly enter an invalid email address on the MFP, which is challenging to do using the MFP's touchscreen or keypad. Note that doing so would likely take a considerable amount of time, which could be noticed by other users. |
| *Exploitability* | Medium | It is fairly easy to exploit as the MFP can only verify the email's address format, but not the validity of the email address. Moreover, the MFP would not receive any feedback from the MSA about the validity of the email address, such as bounce messages, because an MFP does not typically receive messages from an MSA. |

| DREAD element | Risk level | Description |
|---|---|---|
| *Affected users* | High | A blacklist of the MSA could affect other users that use the MSA to send emails. Note that a blacklist is typically done on the basis of the MSA's IP address, which means that all users that use the MSA to send emails could be affected [Lin99]. |
| *Discoverability* | High | It is easy to discover, as the MFP would accept any email address that syntactically conforms to the email address format, regardless of its validity. |

**Conclusion risk 4**
The threat of denial of service is a significant concern when anonymous users can freely enter any email address on the MFP, as it ultimately could lead to a denial of service for the MSA, which would affect other users that use the MSA to send emails.

#### Elevation of privileges

As we are primarily concerned with the confidentiality of the document, elevation of privilege is not directly applicable to this scenario of a user entering their email address on the MFP.

### 6.1.2. Digitizing and sending the document via email

In step **dig-7** the MFP sends the document to the MSA, which is the second step in the digitization process. Next, in step **dig-9** the MSA sends the document to the MTA. Finally, in step **dig-10** the MTA sends the document to the MDA of the user's email provider. Sending the document via email presents a number of threats that are discussed next.

#### Spoofing

The statement is mostly accurate but could benefit from a slight clarification regarding the nature of spoofing and its relevance:

In step **dig-7**, the MFP sends the document to the MSA, marking the second phase in the digitization process. Subsequently, in step **dig-9**, the MSA forwards the document to the MTA. The final stage, step **dig-10**, involves the MTA dispatching the document to the user's email provider's MDA. The process of sending the document via email introduces several potential security threats, which are explored in the subsequent sections.

**risk 5** **The threat that a malicious actor spoofs the MFP to trick a targeted recipient into believing the organization officially endorses or reviews emails, based on the sender's address.**

An MFP may be situated in various locations, such as homes, libraries, airport lounges, copy shops, and supermarkets. The threat may exist due to the MFP's (fixed) sender email address being associated with the organization. This could enable a malicious actor to trick recipients into believing that the email would be officially endorsed or reviewed by the organization with regard to the sender email address. This kind of attack could be used as a stepping stone for more sophisticated attacks, such as phishing.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | The recipient may receive a digitized document from a sender email address that is associated with the organization, for example, mfp@company.com The digitized document could contain a logo and lay-out that is similar to the company's official documents. This could move the recipient to take actions that they would not have taken otherwise, such as going to a spoofed website, controlled by the malicious actor, in order to steal sensitive information. |
| *Reproducibility* | High | This threat can be easily replicated, as it only requires a malicious actor to utilize the MFP to digitize and send a deceptive document, giving the impression of organizational endorsement. |
| *Exploitability* | Low | As crafting a deceptive document requires some effort, combined with the targeted nature of the attack, the exploitability is considered low. Note that for a malicious actor it could be more lucrative to utilize a sender email address from a domain that is similar to the organization's domain under their control, for example, company.com vs. company.co. This approach allows targeting a wider audience, thereby enhancing the potential for exploitation. |
| *Affected users* | Low | The threat only affects the targeted recipient. |
| *Discoverability* | High | It is easy to discover, as the malicious actor utilizes the MFP for normal operation: digitizing and sending a document. |

**Conclusion risk 5**

The threat of email spoofing is a significant concern, as it could be used as a stepping stone for more sophisticated attacks, such as phishing. However, as it is a very targeted attack that requires some effort, its overall risk is moderated.

### Tampering

In our context, where the MFP allows anonymous users to digitize and email a document, the risk of tampering is partially mitigated by the fact that the document is already in the user's possession. However, the risk of tampering also depends on other factors, such as the security measures in place on the MFP and the email system. If these systems are not securely configured, the risk could be significant.

**risk 6** **The threat that an unauthorized actor could alter or manipulate the contents of the email, as it passes from the MFP through the sender MTA to the recipient's MTA, potentially changing its information or inserting malicious contents.**

Note that many email systems are configured to utilize DomainKeys Identified Mail (DKIM) [KCH11] for ensuring the integrity of the email's contents. The recipient's MTA can verify the integrity of the email's contents by checking the DKIM signature, that was added by the sender's MTA. As such, the threat of tampering is mitigated by DKIM, as it is difficult for an unauthorized actor to alter the email's contents without invalidating the DKIM signature. However, DKIM is not widely adopted, and when it is, it is not always configured correctly [WSG$^+$22], which means that it is not always reliable.

| **DREAD element** | **Risk level** | **Description** |
|---|---|---|
| *Damage potential* | High | The recipient may receive a digitized document that has been altered by a malicious actor, which could lead to a loss of information or the introduction of malicious content. |
| *Reproducibility* | Medium | It is difficult to reproduce, as it requires a unauthorized actor to have access to one of the intermediate systems that the email passes through. Moreover, the malicious actor would have to be able to alter the email's contents without invalidating the DKIM signature. As the DKIM signature is not based on the full email, there is a possibility that the unauthorized actor could alter the email's headers without invalidating the DKIM signature.<br>Another important factor to consider is that a DKIM signature can be invalidated by the MTA that receives the email, for example, when an intermediary MTA is configured to add a disclaimer to the email, or, by having an MTA configuration that is not compliant with DKIM or relays emails while altering the email's headers or contents. Although Authenticated Received Chain (ARC) [ALBK19] is designed to somewhat mitigate this issue, it does not cover integrity validation, as it focuses on authentication validation. Moreover, ARC may not be widely adopted as it is a relatively new standard. Bottom line is that integrity enhancing technologies may not always be available or reliable in some configurations, which would make it significantly more easy for a unauthorized actor to tamper with the email's contents. As such, the reproducibility is considered medium. |
| *Exploitability* | Low | As tampering requires access to one of the intermediate mail servers that the email passes through, in combination with the fact that a malicious actor would have to be able to alter the email's contents without invalidating the DKIM signature, the exploitability is considered low. |
| *Affected users* | High | Once an unauthorized actor has set up such a tampering system, it could affect many users. |
| *Discoverability* | Low | It is difficult to discover, as it requires an unauthorized actor to gain access to one of the intermediate email servers. |

Note that even though traffic between an intermediate email servers may not be encrypted, it would still be challenging for a malicious actor to tamper with the email's contents without invalidating the DKIM signature.

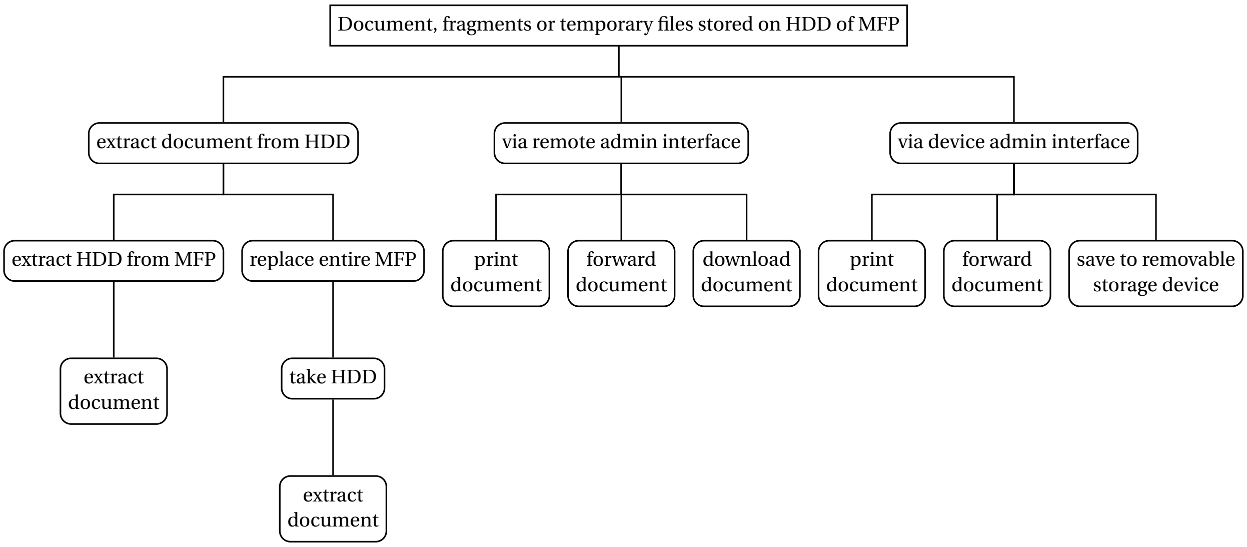
**Conclusion risk 6**

The threat of email tampering is a significant concern, as it could lead to a loss of information or the introduction of malicious content. Since it is mostly difficult to exploit, its overall risk is moderated. However, in the absence of integrity enhancing technologies, such as DKIM, the risk could be significant as it would be significantly easier to tamper with the email's contents.

### Repudiation

Since the MFP sends the email on the basis of the entered email address, the threat of repudiation is already analyzed in **risk 3**.

Figure 6.2: Attack model for **risk 7**, retention of documents, fragments or temporary files on the MFP's hard disk (HDD).


Document, fragments or temporary files stored on HDD of MFP
extract document from HDD
via remote admin interface
via device admin interface
extract HDD from MFP
replace entire MFP
print document
forward document
download document
print document
forward document
save to removable storage device
extract document
take HDD
extract document

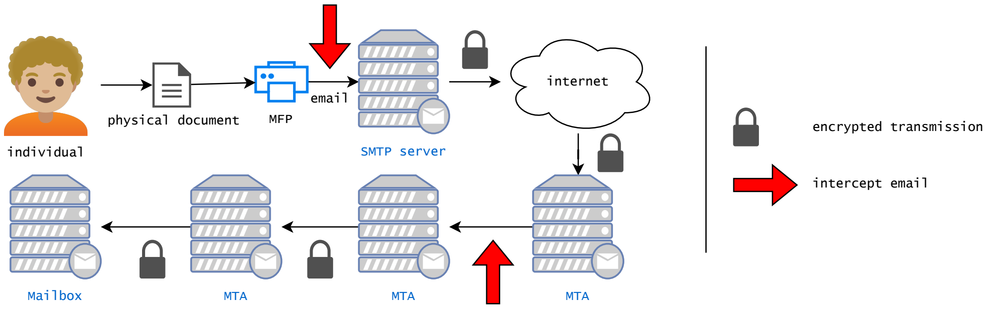
## Information disclosure

**risk 7 The threat that an unauthorized actor could access a digitized document that is stored on the MFP, thereby compromising the confidentiality of the document.**

This threat exploits the potential vulnerability of the MFP storing (unencrypted) copies or fragments of the digitized document.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | An unauthorized actor that accesses a digitized document that is stored on the MFP compromises the confidentiality of the document. |
| *Reproducibility* | High | Once there are no safety measures in place to prevent unauthorized access to the digitized document, it is easy to reproduce. Safety measures could include encryption of the digitized document, full disk encryption, not storing copies or fragments of the digitized document, or, securely deleting the digitized document after it has been sent. However, note that while encryption may be utilized to mitigate this threat, there could be other vulnerabilities that could be exploited to gain access to the digitized document (see exploitability). |
| *Exploitability* | High | It could be easy to exploit, as there could be many ways to access the digitized document. See figure 6.2 for an attack model. |
| *Affected users* | High | It could affect any user that digitizes the document, potentially impacting a large user base. |
| *Discoverability* | High | It is easy to discover, as it only requires an unauthorized actor to access the MFP either physically or remotely. |

Figure 6.3: Risk document compromise by means of network interception (**risk 8**).



**risk 8 The threat that an unauthorized actor obtains the email containing the digitized document due to unencrypted traffic between the MFP and MTA, between MTAs and between MTA and MUA.**

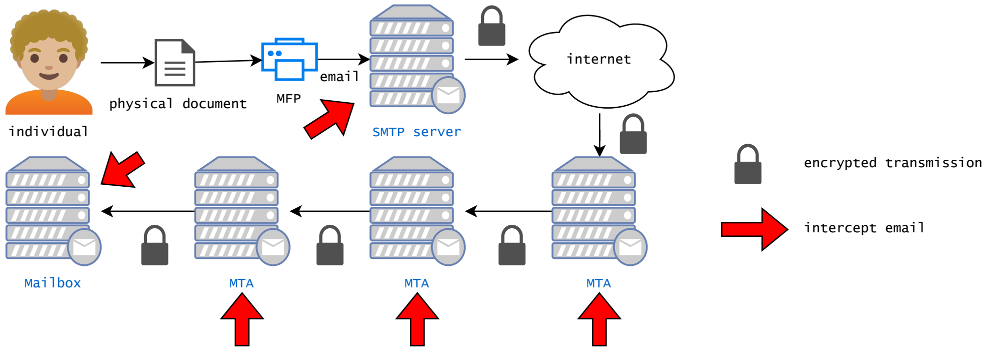
The threat exploits the vulnerability of the email being sent over unencrypted traffic between the MFP and MTA, and between MTAs, see figure 6.3. Note that SMTP [Kle08] does not require encryption, which means that if encryption is not enforced, the email could be sent over unencrypted traffic. Even when employing opportunistic encryption using STARTTLS [Hof02], the email could still be sent over unencrypted traffic if sender or recipient MTA does not support STARTTLS and the MTA does not enforce encryption. Although, Mail Transfer Agent Strict Transport Security (MTA-STS) [MRR+18] could mitigate this issue by requiring the MTA to enforce encryption, it may not be widely adopted as it is a relatively new standard.

| **DREAD element** | **Risk level** | **Description** |
|---|---|---|
| *Damage potential* | High | Unauthorized access to all emails that are sent over unencrypted traffic between the MFP and MTA, or between MTAs, would lead to significant confidentiality breaches. |
| *Reproducibility* | High | Once a malicious actor can intercept unencrypted traffic between the MFP and MTA, or between MTAs, it can consistently replicate the attack. |
| *Exploitability* | Medium | Successful exploitation requires network access, which limits the threat to actors that have access to the network. |
| *Affected users* | High | If the traffic is intercepted, email of all users of the unencrypted MTA are at risk, potentially impacting a large user base. |
| *Discoverability* | Medium | The vulnerability might be discovered by those with knowledge of network security, but it is not immediately obvious to the average user. |

### Conclusion risk 8

The threat that an unauthorized actor obtains the email containing the digitized document due to unencrypted traffic between the MFP and MTA, and between MTAs, is a significant concern, as it could lead to significant confidentiality breaches. Note that the risk is exacerbated by the dependency on correctly implemented security technologies like STARTTLS and MTA-STS, requiring proper configuration across all stages of the email transmission process. This reliance on correct configuration makes the threat more difficult to mitigate.

Figure 6.4: Risk of document compromise as a consequence of unauthorized MTA access, **risk 9**.



> **risk 9** **The threat that an unauthorized actor obtains the email containing the digitized document due to the email passing through multiple MTAs.**

The threat exploits the vulnerability of the email passing through multiple MTAs. Even when encryption between MTAs is enforced, the email could still be accessible to unauthorized actors that have access to the intermediate MTAs, see figure 6.4.

| **DREAD element** | **Risk level** | **Description** |
|---|---|---|
| *Damage potential* | High | Unauthorized access to all emails that are sent through the MTA would lead to significant confidentiality breaches. |
| *Reproducibility* | Medium | This scenario requires the actor to have access to the intermediate MTA, which makes it reproducible for actors that have this kind of access to the MTA. |
| *Exploitability* | Medium | The attack could be feasible for certain attackers, especially insiders or those with advanced capabilities. |
| *Affected users* | High | The emails and documents of all users that use the MTA are at risk, potentially impacting a large user base. |
| *Discoverability* | Medium | Gaining access to the MTA requires some effort, which makes it more difficult to discover. However, it is more discoverable for insiders as they already have access to the MTA. |

### Conclusion risk 9

The threat that an unauthorized actor obtains the email containing the digitized document due to the email passing through multiple MTAs is a significant concern, as it could lead to significant confidentiality breaches.

### Denial of service

As we are primarily concerned with the confidentiality of the document, denial of service is not directly applicable to this scenario of sending the document via email. Note that denial of service could be a consequence of employing MTA-STS, as a sender MTA could refuse to send the email if the recipient MTA does not support encryption. However, this would be a consequence of policy enforcement which actually enhances security, rather than a vulnerability. Although, note that it is possible that a malicious actor with access to an MTA in the email transmission process could configure the MTA to refuse to send emails to a specific recipient MTA, which would be a denial of service attack.

#### Elevation of privileges

An elevation of privilege attack could be possible if the MFP is not properly secured, which could allow an unauthorized actor to gain access to a document that is stored on the MFP, see **risk 7**. The same applies to the MTAs, which could allow an unauthorized actor to gain access to the email containing the digitized document, see **risk 9**.

## 6.2. Reading and sharing

When the user digitizes a document to email, the user typically reads the document on the screen before sharing (forwarding) it. Note that users may also choose to digitize documents only for reading purposes, without sharing them.

### 6.2.1. Reading the document from the email

#### Spoofing

Somebody could send a spoofed email that appears to originate from the MFP, but in fact is sent by a malicious actor. The spoofed email could contain a malicious attachment or link to a malicious website. When the user opens the attachment or clicks on the link, the malicious actor could gain unauthorized access to the user's PC or network. However, we find the risk of this threat in this context to be low, because the user knows when and when not to expect an email from the MFP, which reduces the likelihood of deception. Moreover, the threat of email spoofing is not specific to our context. As such, we do not include this threat in our analysis.

#### Tampering

**risk 10** **The threat that a malicious actor tampers with the email containing the document.**

A malicious actor could tamper with the email containing the document. The likelihood of this threat increases when STARTTLS or TLS is not used to encrypt the connection between the MFP and the MTA, or, when the integrity of the email is not protected, for example, due to the absence or misconfiguration of DKIM.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | The damage potential of this threat is high, because the malicious actor could tamper with the email to contain malicious content, such as a malicious attachment or link to a malicious website. Since the user trusts the email to originate from the MFP, the user is more likely to open the attachment or click on the link. This could result in unauthorized access to the user's PC or network. |
| *Reproducibility* | High | When an email infrastructure lacks the proper configuration of necessary security controls, such as Sender Policy Framework (SPF), DKIM, ARC, and STARTTLS (or just TLS), then the reproducibility of this threat is high, as it could be relatively easy to tamper with the email and email infrastructure may be (mis)configured in a similar way across multiple organizations. |

| DREAD element | Risk level | Description |
|---|---|---|
| *Exploitability* | Low | First the malicious actor needs to gain access to the email infrastructure, which is a non-trivial task. Then the malicious actor needs to tamper with the email, which could be a labor-extensive. As such, the exploitability of this threat is medium. |
| *Affected users* | High | Once the malicious actor has put his setup in place to tamper with passing-by emails, many users could be affected. |
| *Discoverability* | Medium | Improper configuration of the email infrastructure could be discovered by simply analyzing the email headers of a received email, sent by the MFP. Once discovered, the next step for a malicious actor is to gain access to the email infrastructure, which is a non-trivial task. As such, we deem the discoverability of this threat medium. |

**Conclusion risk 10**

The combination of improper configured email controls and a vulnerable MTA could allow a malicious actor to consistently tamper with emails containing documents. This threatens the integrity of the email and the document. Moreover, when malicious content is added to the email, the damage potential increases.

### Repudiation

The user might claim they never received the email with the document. However, this risk is not unique to our scenario and is excluded from our analysis, as it does not relate to the document's confidentiality. Note that an invalid DKIM configuration could increase the likelihood of this phenomenon, as MUAs may not deliver emails to a user's inbox that fail DKIM validation.

### Information disclosure

**risk 11** **The threat of unauthorized access to the document due to temporary unencrypted files stored on the hard disk as a consequence of viewing the document.**

While some MUAs have built-in viewers, others rely on external viewer applications to render the document on the screen. When the used viewer application necessitates the creation of temporary unencrypted files, it makes the document vulnerable to unauthorized access. Factors that exacerbate this risk include: inadequate access control to the temporary folder, use of a shared desktop environment or PC among multiple users, remote file sharing, and unsupervised physical access to the PC or components.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | The damage potential of this threat is high, because it compromises the confidentiality of the document. |
| *Reproducibility* | High | The reproducibility of this threat is high, because the creation of temporary files is a common practice among viewer applications. |
| *Exploitability* | Low | The exploitability of this threat is low, because the malicious actor needs to gain access to the PC or components, which is a non-trivial task. |

| DREAD element | Risk level | Description |
|---|---|---|
| *Affected users* | Low | A malicious actor could only access documents from a PC or components that they have access to. |
| *Discoverability* | Medium | The discoverability of this threat is medium, because the malicious actor needs to gain access to the PC or components, which is a non-trivial task. However, once a malicious actor has access to the PC or components, they may find the temporary files in the temporary folder, along with a broad range of other sensitive data. |

**Conclusion risk 11**
The combination of a viewer application that creates temporary unencrypted files and a vulnerable PC or components could allow a malicious actor to consistently access documents, thereby threatening the confidentiality of the document. However, in the context of our scenario, we find the likelihood of this threat to be low, because the malicious actor needs to have access to the PC or components, which is a non-trivial task.

Finally, it is important to note that PCs often hold a large amount personal data, attractive to malicious actors. Installed malware could automate the search and extraction of documents from temporary folders, increasing the risk.

#### Denial of service

A malicious actor that has access to the MTA that the user's MUA uses to retrieve emails, could delete the email containing the document. This would prevent the user from accessing the document. However, this risk is not unique to our scenario and is excluded from our analysis, as it does not relate to the document's confidentiality.

#### Elevation of privilege

Elevation of privilege in the context of a user reading a document from an email on their PC is not relevant, because the user is a legitimate user of the PC.

### 6.2.2. Sharing the document from the email

This process step is about the user forwarding the email containing the document to the institution. Most of the risks that we identified in the previous section also apply to this process step. These risks include: **risk 2**, **risk 6**, **risk 8** and **risk 9**.

# 7

# Core requirements

In previous chapters we have analyzed the document life cycle, from digitization to disposal, and identified several threats that can compromise the privacy of a document. In this chapter, we define the core requirements for a system that enhances the privacy of a document throughout its life cycle.

Before establishing the core requirements, we first analyze who the stakeholders are and what their interests are with regard to secure digitization, sharing and processing of a document. Next, we identify the core requirements, taking into account the interests of the stakeholders, together with the analysis of the document life cycle process and threats identified in the previous chapters.

## 7.1. Core stakeholders

With regard to secure digitization, sharing, and processing of documents, several stakeholders come into play, each with their unique interests and concerns. Identifying and aligning their interests aids in identifying core requirements for a feasible approach that improves individual's privacy when digitizing and sharing documents with institutions.

### 7.1.1. Individuals

Individuals are users that digitize their physical documents using an MFP in order to share them with institutions. Individuals have different backgrounds and exhibit varied levels of technical expertise. Their interests include:

**individual 1** Keeping documents confidential when digitizing and sharing them with institutions.

**individual 2** Keeping documents accessible for themselves over time, from their mailbox.

**individual 3** Using an MFP that is accessible to individuals, without requiring specialized hardware or incurring additional costs.

**individual 4** Receiving automatic protection of documents, without having to remember to take alternative steps.

### 7.1.2. Institutions

Institutions include organizations, such as government agencies, financial organizations, and other bodies that receive digitized documents from individuals through email. Generally, institutions are interested in compliance with regulations and maintaining a good reputation (avoiding data breaches). Relevant interests include:

**institution 1** Reducing legal liabilities associated with mishandling sensitive information.

**institution 2** Keeping up-to-date with technological advancements, because of their potential to further reduce legal liabilities.

### 7.1.3. MFP manufacturers

Companies that design, manufacture and sell MFPs. In general companies are looking for increasing revenue and market share. With regard to our context, their interests include:

**manufacterer 1** Pioneering in promising state-of-art techniques that can boost market competitiveness.

**manufacterer 2** Investments in innovative technologies should align with projected gains in revenue and market share.

## 7.2. Functional requirements

**requirement 1** **The Multi-Function Printer (MFP) shall employ confidentiality and integrity protection as the default mode of operation for digitizing documents.**

*Rationale.* A user 'just scanning' a document to email should not have to worry about the security of the document. Therefore, the default mode of operation for digitizing documents should be secure, which aligns with the security-by-default[1] principle. This serves the purpose of minimizing the risk of accidental unsecured scans that could lead to a breach of confidentiality. The requirement aligns with stakeholder interests **individual 1**, **individual 4** and **manufacterer 1**.

**requirement 2** **When sending emails, the Mail User Agent (MUA) shall automatically employ confidentiality and integrity protection without exception, when the institution supports this.**

*Rationale.* A user 'just sending' a confidential email to an institution should not have to worry about the security of the email. Therefore, the MUA should employ confidentiality and integrity protection of emails, without exception when the institution supports it, aligning with the security-by-design[2] principle. This requirement aligns with stakeholder interests **individual 1** and **individual 4**.

**requirement 3** **The Mail User Agent (MUA) shall inform the user whether the recipient's domain, based on the entered email address, supports seamless end-to-end encryption.**

*Rationale.* Since emails are typically sent in plain text, users need to know whether the recipient's domain supports seamless end-to-end encryption, as this is only possible when both the sender and recipient's domain support it. This knowledge enables users to make an informed decision about whether to send the email or not. In the case that the recipient's domain *does* support this encryption, informing the user provides assurance, which could enhance trust in the process. This requirement aligns with stakeholder interest **individual 1**.

[1] Note that we purposely employ the term 'security-by-default' rather than 'security-by-design', as we assume that the function of normal, insecure digitization could still be performed on the MFP, albeit by explicitly selecting this option.

[2] Since the security is in the design, as opposed to a setting, we purposely employ the term 'security-by-design'

**requirement 4** **The institution shall support seamless end-to-end encryption of incoming emails by providing a mechanism that allows a Mail User Agent (MUA) to retrieve the Secure/Multipurpose Internet Mail Extensions (S/MIME) certificate for the email address entered by the user, which is, then, used to encrypt the email.**

*Rationale.* This requirement ensures that the institution supports seamless end-to-end encryption of incoming emails, which is necessary for the MUA to encrypt the email. This requirement aligns with stakeholder interests **individual 1**, **individual 4** **institution 1** and **institution 2**.

**requirement 5** **The system will enable users to decrypt previously encrypted emails using a private key associated to the public key of an expired certificate.**

*Rationale.* While it is good practice to renew certificates, this has the potential to cause issues with decrypting older encrypted emails that were encrypted using a certificate that has since expired. This phenomenon conflict with the user's interest in keeping documents accessible over time, refer **individual 2**. As such, this requirement ensures that users can still decrypt previously encrypted emails over time, stored in their mailbox.

## 7.3. Non-functional requirements

### 7.3.1. Accessibility

**requirement 6** **Secure digitization of a document using an Multi-Function Printer (MFP) must be accessible to individuals by not requiring specialized hardware or incurring additional costs.**

*Rationale.* Our ambition is to propose secure scanning as the default mode of operation for digitizing documents using an MFP. As such, it is important that the process stays accessible to individuals, just as the default scanning process is. By not requiring specialized hardware or incurring additional costs, the secure scanning process is about to remain accessible to individuals in the same way as the default scanning process. This requirement corresponds to stakeholder interest **individual 3**.

### 7.3.2. Security

**requirement 7** **Throughout the whole process, starting with digitization, up to the moment that the document eventually arrives at the institution, the document's confidentiality and integrity shall be protected end-to-end, when the institution supports this.**

*Rationale.* This end-to-end security approach ensures that the document is protected against unauthorized access and tampering throughout the process of digitization and transmission, which is at the user's discretion. Once the document arrives at the institution, the protection of the document becomes the responsibility of the institution, and, as such, relies on the institutional framework that consists of policies, procedures, and security measures, that ensure compliance with regulations and standards. This requirement aligns with stakeholder interests **individual 1** and **individual 4**.

**requirement 8** **The Multi-Function Printer (MFP) should ascertain that unprotected documents, document fragments and key information can never be extracted from its hard disk in plaintext.**

*Rationale.* Stored documents or fragments on the MFP's hard disk should never be accessible in plaintext. This requirement aligns with stakeholder interests **individual 1**, **individual 4** and **manufacterer 1**.

### 7.3.3. Constraints

In this subsection we outline specific limitations and assumptions that shape the design of the proposed system. These constraints are critical in guiding the development process and ensuring that the system is both practical and feasible, within the given parameters.

**constraint 1** **It is assumed that users possess basic skills, such as being able to digitize a document using a Multi-Function Printer (MFP), sending or forwarding an email, installing smartphone apps, scanning a QR code, interact with smartphone apps, together with other typical user capabilities.**

*Rationale.* Given our focus on enhancing the document digitization process on MFPs, it is a constraint that users must already be capable of performing scans. Moreover, since we utilize email as the primary communication channel, users must be able to send and forward emails. In general, we assume that users possess basic skills in performing other common tasks, such as interact with smartphone apps, installing smartphone apps, scanning QR codes, etc. The rationale for this constraint is that our proposed system requires users to perform these tasks. Note that tasks like configuring complex settings on an MFP, obtaining S/MIME certificates, configuring email clients to work with encryption, etc. are not considered basic skills. While this constraint does not align with any of the provided stakeholder interests, it is a necessary constraint to ensure that our proposed system is feasible.

**constraint 2** **The Multi-Function Printer (MFP) is presumed to lack a Trusted Platform Module (TPM), but is assumed to have a Cryptographically Secure Random Number Generator (CS-RNG).**

*Rationale.* It is a reasonable assumption that not every MFP is equipped with a TPM, as secure scanning is not a standard feature in all models and incorporating a TPM can be cost-prohibitive for manufacturers. However, most modern MFPs capable of cryptographic operations are presumed to include a CS-RNG, which is essential for generating cryptographic keys, thus balancing the security capabilities within the constraints of typical MFP designs. This constraint aligns with stakeholder interest **manufacterer 2**.

**constraint 3** **It is assumed that Multi-Function Printer (MFP) manufacturers are willing to invest in the development of secure digitization capabilities for their MFPs.**

*Rationale.* MFP manufacturers are primarily interested in increasing revenue and market share. As such, they are willing to invest in the development of secure scanning capabilities for their MFPs, especially when the necessary investment is balanced with projected gains in revenue or market share. Note that from the MFP manufacture's perspective, the research, design and development of accessible secure scanning capabilities for their MFPs, directed at anonymous use rather than enterprise use, may be a promising area, as it is a relatively new concept that has not yet been widely adopted. This constraint aligns with stakeholder interests **manufacterer 1** and **manufacterer 2**.

# 8

# Design methodology: digitization

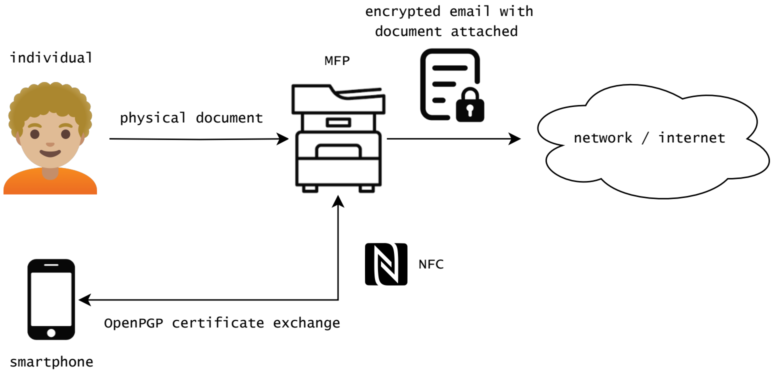
This chapter presents a design methodology for secure digitization: a structured approach aimed at addressing the identified threats and fulfilling the core requirements that have been previously outlined. This methodology serves as the architectural blueprint for developing a system capable of mitigating security risks while balancing stakeholder interests.

Document scanning or digitization is the process of converting a physical document into a digital document. Secure digitization is the process of converting a physical document into a digital document in a secure manner by protecting the document's *confidentiality* and *integrity*. Figure 8.1 provides an overview of the secure digitization process. The following sections provide an in-depth methodology on how this could be achieved.

## 8.1. Secure digitization

The secure digitization process intends to convert a physical document into a digital document while protecting the document's confidentiality and integrity, which aligns with **requirement 7**. It uses a smartphone to manage the user's OpenPGP certificate, for performing key operations, and the MFP's OpenPGP certificate, for authentication of the MFP. The public keys contained in the certificates are used for encrypting the document attachment and signing the email, respectively for protecting confidentiality and integrity of the document. The authenticated MFP's certificate is, later, used to verify the authenticity and integrity of the email when the user reads the email in the MUA, refer

Figure 8.1: Illustration of document digitization using an MFP.



chapter 9. The user's certificate contains the user's email address, which the user can select from a list of registered email addresses on the smartphone, when initializing the secure scanning smartphone app. Subsequent sections elaborate on the secure digitization process.

### 8.1.1. Establishing the encryption key

For secure scanning using an MFP there is more than one way to establish a secret key. For example, a symmetric encryption key could be generated using a key derivation function based on a user-provided password. However, the password-based approach is not necessarily secure as passwords are often re-used [DBC+14] and of poor quality [FH07] [DMR10], which may lead to vulnerable encryption keys. Finally, requiring the user to enter a password on the MFP's touch display or physical keyboard is inconvenient, as it demands certain motor skill and patience from the user.

Another approach would be the use of a symmetric encryption key that is generated by a Cryptographically Secure Random Number Generator (CS-RNG), as they are capable of generating strongly unique keys that are statistically indistinguishable from random [L'E12]. Note that strongly random keys are a prerequisite for robust encryption. Since the MFP is equipped with a CS-RNG, refer **constraint 2**, it is possible to generate a strong encryption key on the MFP itself.

### 8.1.2. Managing the encryption key

The encryption key must be stored securely in a designated location. An approach is to require the user to obtain a physical security key, such as a Yubikey[1] or a Feitian MultiPass FIDO Security Key[2] for key storage. These keys are specifically designed for the purpose of managing and exchanging encryption keys and are also equipped with a CS-RNG which allows strong encryption. Moreover, they contain a Trusted Platform Module (TPM), which can be used for securely storing private keys, and support wireless protocols, such as Bluetooth and NFC, which can typically be found on MFPs also. However, although this would be a viable approach, it would require the user to acquire such a specialized security key, which conflicts with accessibility, refer **requirement 6**.

Alternatively, the MFP could require the user to insert a USB stick or memory card that the MFP uses to exchange keys. Note that USB-sticks are more ubiquitous than physical security keys. However, USB ports are an attack vector for spreading malware [NYE17a], [MS19]. As a consequence, administrators of MFPs, especially in the more public settings, may choose to disable the USB ports or make them physically inaccessible. In addition, a user that digitizes a document to a removable storage device not only has to remove the original after scanning, but also the removable storage device attached to the MFP. This dual responsibility exacerbates the risk of oversight, leaving the removable storage device behind at the MFP.

**Using a smartphone for managing the encryption key**

While especially the security key approach is viable for its security characteristics, we argue that the use of a smartphone, too, is a viable option for managing the encryption key. In addition, smartphones are more ubiquitous than physical security keys [sta22a], [ban23], [bri23]. We contend that using a smartphone to manage encryption keys is, both, a viable and accessible approach for our context, considering the widespread use of smartphones and the tendency of people to bring their smartphone everywhere they go, which aligns with our requirement of accessibility. Moreover, since smartphones are inherently personal devices [MP20], which are typically equipped with secure hardware for storing encryption keys, and, various communication channels, including Wi-Fi, NFC, and Bluetooth, they make a fitting option for managing encryption keys.

[1] https://www.yubico.com/

[2] https://www.ftsafe.com/products/fido/multi

### Selecting the email address

In the model default scanning process the has to enter their email address on the MFP's touch display or physical keyboard, see **dig-3**. Having the individual type their email address on the MFP's touch display or physical keyboard is not convenient, as it requires the user to have a certain level of motor skill and patience. Moreover, the risk exists that the user makes a typo, which could result in the document being sent to the wrong email address. Since we have established to use a smartphone for managing the encryption key, we argue that it is feasible to have the user select their email address from a list of registered email addresses on the smartphone. This is a viable approach, as the smartphone's operating system typically provides an API for accessing the user's email addresses. Finally, note that this implies that the proposed design necessitates the existence of a smartphone app, as such functionality is not available in the smartphone's operating system itself. This app needs to be developed and maintained. Given our assumption that MFP manufacturers are willing to invest in research and design, refer **constraint 3**, the MFP manufacturer could develop and maintain the app. However, we aim to propose a design that has the potential of becoming a standard rather than a proprietary solution. As such, while MFP manufacturers develop and maintain the app, we argue that the app should be open source, which allows other parties to contribute to the app's development and maintenance. Moreover, this could make apps from different manufacturers interoperable, akin to Time-based One-Time Password (TOTP) apps used for Two-Factor Authentication (2FA), which would be beneficial for the user.

### Leveraging the smartphone's secure hardware

The smartphone's secure hardware can be used to, both, *store* and *use* secret keys, in such a way that the key is never exposed to the smartphone's operating system. Note that the secure hardware refers to the Trusted Execution Environment (TEE) of the smartphone, which is an isolated operating system capable of performing cryptographic operations. In addition, access to the smartphone can be protected, for example, by a PIN or biometric authentication, which adds an extra layer of security in case the smartphone is lost or stolen. We argue that using a smartphone for backing the document's encryption key is a viable option, as this method provides adequate protection by means of the smartphone's secure hardware and (protected) access to the smartphone itself. Moreover, it does not conflict with the requirement of accessibility, refer **requirement 6**, given the ubiquity of smartphones.

### Where to store the encryption key?

An important aspect of transmitting the document is the format to be used for the encrypted document in an email. The format should be platform-independent, as the email could be read on various platforms, such as Windows, macOS, Linux, Android and iOS. Common industry standards are Secure/Multipurpose Internet Mail Extensions (S/MIME) and OpenPGP, which are both based on public key cryptography. These are established standards that either could be viable option for our context.

While both standards encompass a trust model, S/MIME relies on a centralized trust model based on Certificate Authority (CA). A CA is responsible for issuing certificates to users. The process of obtaining a certificate from a CA is cumbersome, mostly not free of charge, and requires the user to provide personal information. For example, when ordering a one-year DigiCert personal S/MIME certificate, the user is expected to provide a Certificate Signing Request (CSR) which contains the user's name and email address. [Dig23]. However, creating a CSR could be considered a technical challenge for the average individual, which makes method less accessible. Even when this process would be simplified, for example by means of a wizard that utilizes a CA's Application Programming Interface (API), then another hurdle would be the incurred cost of obtaining the certificate, which conflicts with **requirement 6** (accessibility).

OpenPGP, on the other hand, relies on the Web of Trust, a decentralized trust model, which just means that users can vouch for each other by signing each other's public key. As such, reputation wise, when many users have signed a public key, it is more likely that the public key is trustworthy. As OpenPGP does not rely on a CA, a user can create their own public-private key pair, thereby omitting the complexities involved in obtaining a certificate from a CA.

On the question of which standard best fits our context, we argue that OpenPGP is more suitable than S/MIME, as creating a public-private key pair is less complex than obtaining an S/MIME certificate, as it can be done independently, without the need for a CA. This independence does not only simplify the process, but also eliminates potential delays and complexities associated with CA-based certificate issuance in S/MIME. Section 8.2 discusses sending the email and the use of OpenPGP in more detail.

### 8.1.3. Secure communication between MFP and smartphone

As we have established to use OpenPGP for encrypting the email, the smartphone needs to share the OpenPGP certificate with the MFP, as it contains the public key that the MFP uses to encrypt the document's encryption key. Note that the user's email address can also be included in the certificate, which solves the issue of communicating the user's email address to the MFP, refer 8.1.2. This aligns with how OpenPGP is typically used [FDC+07].

Both MFPs and smartphones typically contain various communication channels, like Near Field Communication (NFC), Bluetooth and Wi-Fi. Any communication channel would be a viable alternative for the exchange of the certificate, however NFC seems to be most fitting:

- NFC is a (very) short-range communication channel which corresponds well with the user's proximity to the MFP. The short-range characteristic of NFC naturally mitigates the risk of eavesdropping and man-in-the-middle attacks. Although, it must be noted that it has been demonstrated that NFC communication can be eavesdropped over longer distances [EPFB13] [JvdBdR16].

- The data size for transfer is small, a scenario in which NFC fits. Note that the size of the public key depends on the algorithm and chosen key size. For example, a 2048-bit RSA (Rivest-Shamir-Adleman) public key is 256 bytes, an Elliptic Curve Cryptography (ECC) public key is typically 32 bytes. However, public key sizes of post-quantum algorithms can be substantially larger. For example, the CRYSTALS-KYBER / ML-KEM algorithm has a public key size ranging from 736 bytes for 'light' mode up to 1440 bytes for 'paranoid' mode [BDK+18]. Nonetheless, even these larger key sizes of a public key are not an issue for NFC, when working with Host Card Emulation (HCE) [Sma14] on smartphones, which is the technique we utilize for exchanging the encryption key in our proof of concept.

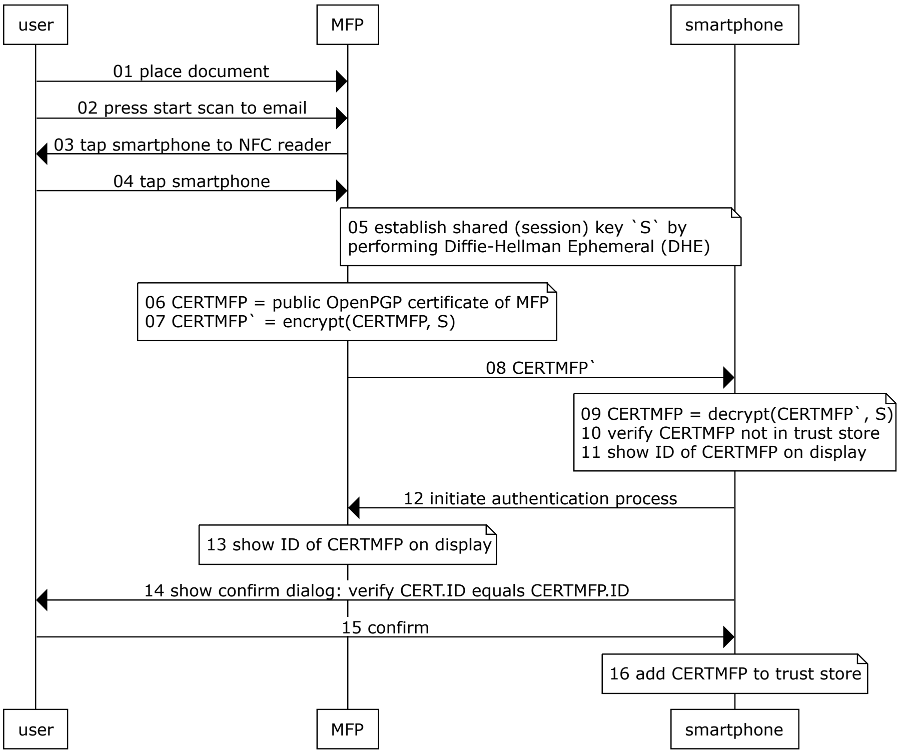
- NFC offers a user-friendly 'tap and go' experience which aligns with the requirement of accessibility. This is in contrast to Bluetooth and Wi-Fi, which usually necessitate initial setup, such as pairing or connecting to a network.

While we acknowledge that not all MFPs feature NFC capabilities, we contend that manufacturers, already investing in research and design, would be willing to allocate a modest budget needed for additional NFC hardware, refer **constraint 3**.

**Secure transmission over NFC**

Although the smartphone and MFP exchange their public OpenPGP certificates with each other, which are not secret, we argue that the NFC communication between smartphone and MFP necessitates a secure channel for two reasons. First, the OpenPGP certificate contains the user's email

Figure 8.2: Establishing trust in the MFP on first use


user
MFP
smartphone
01 place document
02 press start scan to email
03 tap smartphone to NFC reader
04 tap smartphone
05 establish shared (session) key `S` by performing Diffie-Hellman Ephemeral (DHE)
06 CERTMFP = public OpenPGP certificate of MFP
07 CERTMFP` = encrypt(CERTMFP, S)
08 CERTMFP`
09 CERTMFP = decrypt(CERTMFP`, S)
10 verify CERTMFP not in trust store
11 show ID of CERTMFP on display
12 initiate authentication process
13 show ID of CERTMFP on display
14 show confirm dialog: verify CERT.ID equals CERTMFP.ID
15 confirm
16 add CERTMFP to trust store
user
MFP
smartphone

address, which could be considered sensitive information. Second, employing encryption for all communication, including non-sensitive information, establishes strong security posture, which could complicate efforts for attackers to identify and exploit sensitive information.

Leveraging NFC for data exchange naturally minimizes security risks due to the proximity characteristic of NFC communication. However, MFPs may be situated in more public places, like libraries, with less supervision but being used by many anonymous individuals. In that scenario the likelihood of tampering with the MFP's NFC reader increases. We are particularly interested in the risks of data interception and MITM attacks as they target the *confidentiality* and *integrity* of the data transmitted. Note that MITM attacks over NFC are possible [ACC+21]. However, the method used in the article is not directly applicable to our context as it is highly complex, and, also, the additional hardware is easily detectable. Although the probability of MITM attacks over NFC seems low, we argue that protecting against MITM attacks is a reasonable precaution, as developments in technology could make MITM attacks more feasible in the future, and, the consequences of a successful MITM attacks could be severe. For example, an MFP may be set up to capture encryption keys. When an attacker is also able to intercept encrypted emails, the attacker could easily decrypt the emails and obtain the documents.

**Establishing trust of the MFP on first use**

A method to establish an (unauthenticated) secure channel is by employing Diffie-Hellman Ephemeral (DHE) to establish a session key. The session key is, then, used to encrypt all NFC communication. Next, the MFP can communicate its certificate to the smartphone, for which the smartphone verifies whether the certificate is trusted. As OpenPGP's trust model relies on the web of trust, the smartphone could verify whether the MFP's certificate is trusted by checking whether the certificate has been signed by a trusted user. The trusted user could be the MFP's manufacturer or the organization that hosts the MFP. However, this implies OpenPGP key management overhead, as OpenPGP certificates for this purpose need to be created, signed and managed. We argue that although this is a viable approach, authentication could be established using a more direct approach without the overhead for key management, which is to have the user verify and authenticate the MFP's certificate on the smartphone using a Trust On First Use (TOFU) approach. As such, an MFP could manage the OpenPGP certificate by itself, by creating it and rolling it over when it expires. Note that TOFU can be established using the smartphone's and MFP's display, which effectively is an out-of-bound channel, to display a readable fingerprint of the MFP's certificate. The user can then verify whether the fingerprint on the smartphone's display matches with the fingerprint on the MFP's display, and, if so, confirm that the MFP's certificate is trusted, after which the smartphone adds the MFP's certificate to its list of trusted certificates. Figure 8.2 illustrates this approach.

Another Out-of-Band authentication (OOB authentication) method is to have the MFP display a QR code that the user scans with the smartphone's camera. The QR code could contain a secret key, generated using a CS-RNG, that the smartphone uses to encrypt subsequent communication over NFC. Although we did not model this approach in this context of secure digitization, we argue that this method is a viable alternative that may have the added benefit of reducing human error when comparing fingerprints. We did, however, model this approach in the context of secure viewing (refer section 9.2).

Finally, note that the Web of Trust approach and the TOFU approach could be employed in tandem, for example, by showing the user the trust level of the MFP's certificate on the confirmation dialog for verification.

### 8.1.4. Rigid clean-up policy

The MFP must ascertain that unprotected documents and fragments should never be retrievable from its hard disk, refer **requirement 8**. To adhere to this requirement it is important to consider the internal working of an MFP when digitizing a document as the MFP could store the document on the MFP's hard disk (refer **dig-19**). For larger documents consisting out of multiple pages, MFPs typically cache digitized pages to disk during the digitization process. As a consequence, a sudden power outage could mean that unprotected parts of the document could be retrieved from the disk. Moreover, even when the MFP manages to delete the temporary files generated during the digitization process, they could possibly be recovered due to how file systems typically perform deletions [TK18]. As such, this requirement contains two aspects:

1. Unprotected documents and fragments must be deleted *securely*.

2. Unprotected documents and fragments must *always* be deleted.

Note that suitable approaches largely depend on the characteristics of the MFP itself. For example, some MFPs have a lot of Random Access Memory (RAM) which makes a 'no disk caching' strategy possible, performing digitization and encryption entirely in RAM. Other MFPs may utilize full disk encryption, which makes it less feasible – however not impossible – to recover fragments from the disk. More affordable MFPs may not have a large amount of RAM and may not have disk encryption options. Because of this variety, we assume that the MFP may cache unprotected pages to disk during the digitization process, which needs to be addressed, refer **requirement 8**.

**Secure deletion of temporary files**
Simply deleting a file from disk could mean that the file could be easily recoverable. There are various methods that provide reasonable assurance that a deleted file cannot be recovered anymore. The CLEAR method as described in the NIST 800-88 [RFW15] protects against casual recovery attempts, which could be sufficient for our context. In addition, the CLEAR method can be implemented efficiently, catering to MFPs with lower specifications.

**Ensuring deletion of temporary files**
Temporary files generated during the digitization process, stored on the MFP's hard disk, may contain unprotected fragments of digitized documents. This is a threat to confidentiality because the files could be obtained. As such, it is imperative that the MFP deletes the temporary files as soon as possible. For example, the MFP could clean up all temporary files after the scanning process has completed. However, there are a few issues with that approach. For example, temporary files could be obtained through the network that the MFP is connected to, for instance via the remote admin interface or a file share during the scanning process. Moreover, in the case of a sudden power failure the temporary files may not be deleted.

A method to overcome the aforementioned consequences is to have the MFP write temporary files in encrypted form to disk. However, this comes at a performance cost. For example, when digitizing a large multipage document, the MFP could create a temporary file for each digitized page. After the last page is digitized the MFP, then, must perform a decryption action on each of temporary file in order to ultimately construct the final document by combining all pages and re-applying encryption. Note that applying encryption on the final document could also require caching data to disk.

**Employing stream-based encryption**
Another method is to utilize stream-based encryption. As the final document needs to be encrypted, utilizing stream-based encryption enables the creation of an encrypted document page-by-page as each is being digitized[3]. This obviates the need to cache pages temporarily to disk, as the MFP can directly write the encrypted page to disk. For example, in the case of a sudden power failure the MFP contains a partly digitized document which happens to be fully encrypted. We argue that this method is more suitable, as it is both effective and efficient, making it fit for MFPs with limited hardware capabilities. As for encryption, it is important to note that the MFP should keep the symmetric encryption key solely in memory, as storing the key on disk would defeat the purpose of stream-based encryption. Figure 8.3 illustrates this method. Finally, note that the CLEAR method can be applied to the encrypted document, which provides reasonable assurance that even the encrypted document cannot be recovered anymore.

### 8.1.5. Secure scanning process alignment

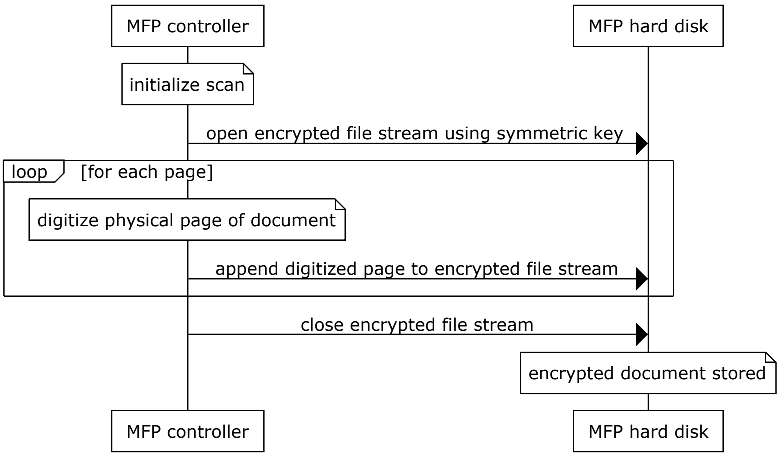
Our proposed design for secure scanning introduces new elements as compared to the modelled unsecure scanning process. These include the use of a smartphone and the secure scanning smartphone app. As we assume that users are familiar with the unsecure scanning process and are capable of performs certain additional tasks, such as holding a smartphone to an NFC reader, installing an app and performing initial setup, refer **constraint 1**, we argue that the secure scanning process, as proposed, is accessible to the user.

When a user enters the MFP and intends to scan a document, the MFP does not know yet whether the user is a first-time user or a returning user. First-time users may not yet have the secure scanning app installed on their smartphones, whereas returning users likely prefer not to receive detailed instructions. Without making assumptions about the user, the MFP could facilitate both first-time and returning users simultaneously, for example, by displaying a message like "Welcome, please hold your smartphone to the NFC reader to start the secure scanning process" while also displaying a QR code, accompanied by an instructive text, that a first-time user can scan with their smartphone to download the secure scanning app.

**First-time user**
First-time users need to download the secure scanning app on their smartphone. After that, the user

[3] Using AES, which is a block cipher, each page would be encrypted in multiple blocks of 128 bits directly written to the open file stream.

Figure 8.3: Illustration of page by page encryption by means of streamed encryption.


MFP controller
MFP hard disk
initialize scan
open encrypted file stream using symmetric key
loop
[for each page]
digitize physical page of document
append digitized page to encrypted file stream
close encrypted file stream
encrypted document stored
MFP controller
MFP hard disk

needs to perform initial setup, which includes registering the user's email address and generating the OpenPGP certificate. The user can select their email address from a list of registered email addresses linked to accounts on the user's smartphone. This obviates the need for the user to type their email address and, which mitigates the risk of typos. Note that the smartphone app has no way of verifying whether the email address is under the user's control, because the app does not have a central infrastructure to verify the email address. We argue that since the user selects an email address belonging to a registered account on the smartphone, the inability to verify control over the email address is not a concern, as the user has already demonstrated control over the registered account on the smartphone.

**Returning user**

Returning users have already downloaded and configured the secure scanning app on their smartphone. We envision that those users can initiate the secure scanning process instantly by holding their smartphone to the MFP's NFC reader.

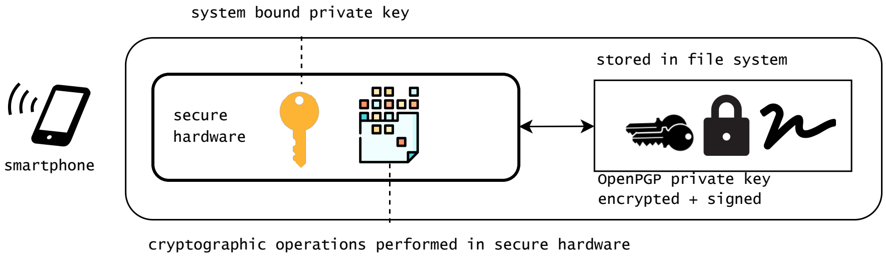
## 8.2. Transmitting the document via email

After digitizing the document, the MFP sends an encrypted email containing the digital copy of the document to the user's email address, in accordance with **requirement 7**. Since email is encrypted, various email related risks to security are mitigated, which include **risk 6**, **risk 8** and **risk 9**.

### 8.2.1. Encrypted email

Since the email that the MFP sends to the user contains the document as an attachment, it implies that the email content type is multipart/mixed, indicating that the email contains multiple parts with each its own content type. OpenPGP allows to selectively encrypt parts of an email, which allows for encrypting solely the document part, while leaving the email body unencrypted. Note that the MFP does not output sensitive information in the email body, which means that the email body does not necessitate encryption. Encrypting solely the document part allows for a more streamlined process when reading or forwarding the email and document, as the user does not need to go through the process of decrypting the email body part using their smartphone just to obtain nugatory text, while having to go through that same process again when opening the document, refer figure 9.5 and 10.5.

Figure 8.4: Leveraging the smartphone's secure hardware for protecting the OpenPGP's private key.



### 8.2.2. Validity period of the OpenPGP key pair

While OpenPGP does not impose a validity period on a certificate, we argue that a validity period is necessary, as it conforms to best practices in key management [Nat20]. We argue that a validity period of 2 years is a reasonable trade-off between usability (convenience) and security as it aligns both with NIST's recommendation for key management [Nat20] of a validity period between 1 and 2 years, and, the industry's standard of 825 days (2 years and 3 months) for new S/MIME certificates [CA/23].

An important consideration with regard to expiring OpenPGP certificates is the impact this has on older emails that contain digitized documents, stored in the user's mailbox. For example, when the user's OpenPGP certificate expires, the user may no longer be able to decrypt documents contained in these older emails, that were encrypted with the private key belonging to the expired certificate. We argue that while the public OpenPGP certificate may expire, and the user's smartphone app rolls over to a new certificate, the private key of the old certificate must be retained, as it is required to decrypt older emails, aligning with **requirement 5**. This is a viable approach, as private keys can be stored securely on the smartphone. Upon reading an older email, when the MUA requests to decrypt a key, the smartphone can determine which private key to use, based on the OpenPGP key-id contained in the email.

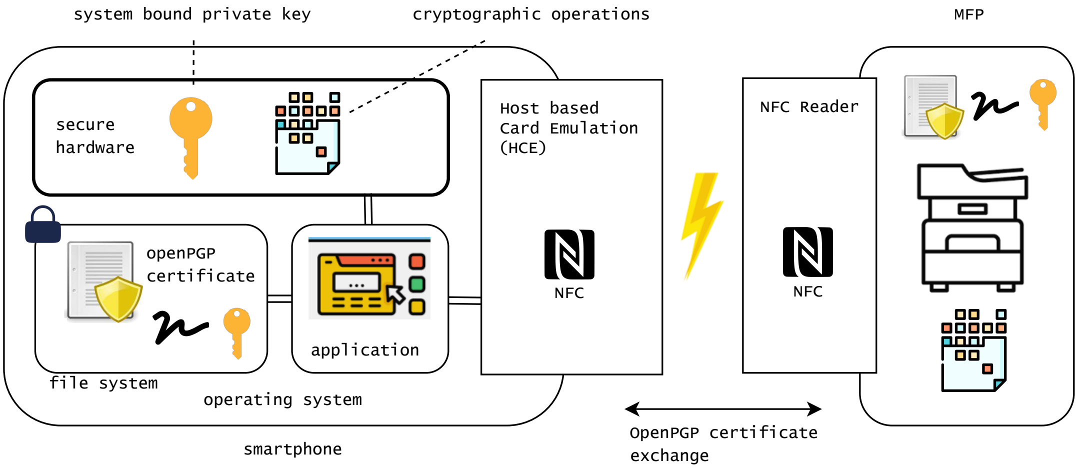
### 8.2.3. Utilizing OpenPGP on an MFP

Although we did not identify *any* MFP models that supports OpenPGP, we do contend that it is feasible to implement OpenPGP on an MFP, when the MFP is capable of performing cryptographic operations. Note that while OpenPGP is a standard that describes the format of encrypted data, it uses similar cryptographic primitives as used by S/MIME, which is supported by various MFPs. Therefore, there seem to be no inherent technological limitations that would prevent MFPs from supporting OpenPGP.

**Authenticity and integrity of emails sent by the MFP**
When the MFP sends an email containing the encrypted document, it is important that the email is authentic and that the integrity of the email is preserved. A straightforward method to achieve this is by having the MFP sign the OpenPGP encrypted email with the private key of it OpenPGP key pair. As we choose to equip the MFP with a OpenPGP certificate, refer 8.1.3, the MFP can sign the email with its private key.

**Verifying the authenticity and integrity of emails using the MFP's OpenPGP certificate**
The MFP's OpenPGP certificate that the MFP sends to the smartphone, can be used by the smartphone to verify the authenticity and integrity of the email sent by the MFP when the user intends to read an email, before the smartphone allows to decrypt the symmetric encryption key of the email. Note that upon reading a document, the MUA requests the smartphone to decrypt the symmetric encryption

Figure 8.5: Illustration of components in the secure scanning process.



key of the email, which the MUA, then, uses to decrypt the email. As the smartphone can retrieve the MFP's OpenPGP certificate from its authenticated certificates store, it can easily verify the authenticity and integrity of the email sent by the MFP (refer 8.1.3). This topic is discussed in more detail in the succeeding chapter on reading the document.

### 8.2.4. Utilizing OpenPGP on a smartphone

As with the MFP, the same applies to the smartphone: while mobile operating systems typically do not support OpenPGP out of the box, it is feasible to implement OpenPGP, as smartphones are capable of performing cryptographic operations.

**Storage location of the smartphone's OpenPGP private key**
By leveraging the smartphone's secure hardware, the private key of the OpenPGP key pair can be stored securely. The secure hardware ensures the private key is protected and not directly accessible by any applications or external libraries, which aligns with best practices in cryptographic key management, for example, see [Nat20] section '5.5.2. Protective measures'. However, a consideration is that private keys stored in a smartphone's secure hardware typically cannot be exported. This has two important implications. First, the private key cannot be backed up, which means that the user loses access to all encrypted emails when the smartphone is lost or stolen. Second, the private key cannot be used on other devices, which contradicts OpenPGP's multi-device usage feature of using the same key pair on multiple devices.

This issue can be addressed by employing a Key Encapsulation Mechanism (KEM) approach that uses a smartphone bound key pair for the encryption of the OpenPGP's private key, which is, then, stored in the (less secure) smartphone's file system. This is security trade-off: while the encryption of the OpenPGP's private key is protected using the smartphone's secure hardware, the encrypted private key is stored in the smartphone's file system. As a consequence, cryptographic operations involving the OpenPGP's private key are performed outside the secure hardware, which increases the risk of compromise by malicious applications that can access the app's address space. We argue that this approach is viable, as the risk of compromise is mitigated by the fact that the OpenPGP's private key is encrypted at rest, and, that a mobile's app address space is typically sandboxed, which limits the attack surface [PR19]. Figure 8.4 gives illustrates this approach.

**Storage location of the smartphone's OpenPGP certificate**
The OpenPGP certificate, which contains the public key does not require the same level of protection

as compared to the private key, as it is intended to be shared with others. Therefore, the OpenPGP certificate can be stored in the file system, which is directly accessible by the mobile app. However, it is important to protect the integrity of the OpenPGP certificate, as a malicious actor could alter or replace the certificate with a malicious one. This could result in the MFP sending the encrypted document to the email address of that malicious actor, who can also decrypt the document using their private key.

A method that could be used to protect the integrity of the OpenPGP certificate is to have the mobile app create a cryptographic signature of the certificate using the smartphone bound private key, which is then stored with the certificate in the file system. This effectively binds the certificate to the smartphone, as the signature can be verified using the smartphone bound public key before each use.

**Backups**
As we intend to keep emails containing encrypted documents accessible to the user for a long time, it is important to consider the risk of losing access to the private keys of the OpenPGP key pair stored on the smartphone, in accordance with **requirement 5**. For example, when the smartphone is lost or stolen, the user loses access to the private keys, which means that the user can no longer decrypt emails containing encrypted documents. For this reason it is important to have a backup of the private keys. Note that since the design stores the OpenPGP private key encrypted on the smartphone's file system, rather than in the smartphone's secure hardware, refer figure 8.4, it is possible to create a backup of the user's private OpenPGP key(s).

One method is to have the user perform a periodical manual backup of the private keys, for example, by exporting the private keys to a file, protected with a password, which the user can store in a secure location. However, this method is prone to human error, as the user may forget to perform the backup, the user may lose the backup file or the user may forget the password.

Another method is to leverage the smartphone's backup functionality, which is typically provided by the smartphone's operating system. On Android, for example, the operating system provides an API that allows apps to store data in the smartphone's backup, which is synchronized with the user's Google account, and, stored in the cloud in encrypted form [And23a] [And23b]. While this method is more convenient and allows for automatic periodical backups, it is important to note that the backup is stored in the Google cloud, and, that Google technically has the capability to access them. As such, this is a notable trade-off between convenience and security, in which the user would have to place their trust in Google's security and privacy practices. We argue that this is a reasonable trade-off, as Google's privacy and security practices, as described in their privacy policy [Goo23b] and security whitepapers on key management practices [Goo23d] [Goo23c], could be considered of a high standard, which outweighs the risk of losing access to the private keys.

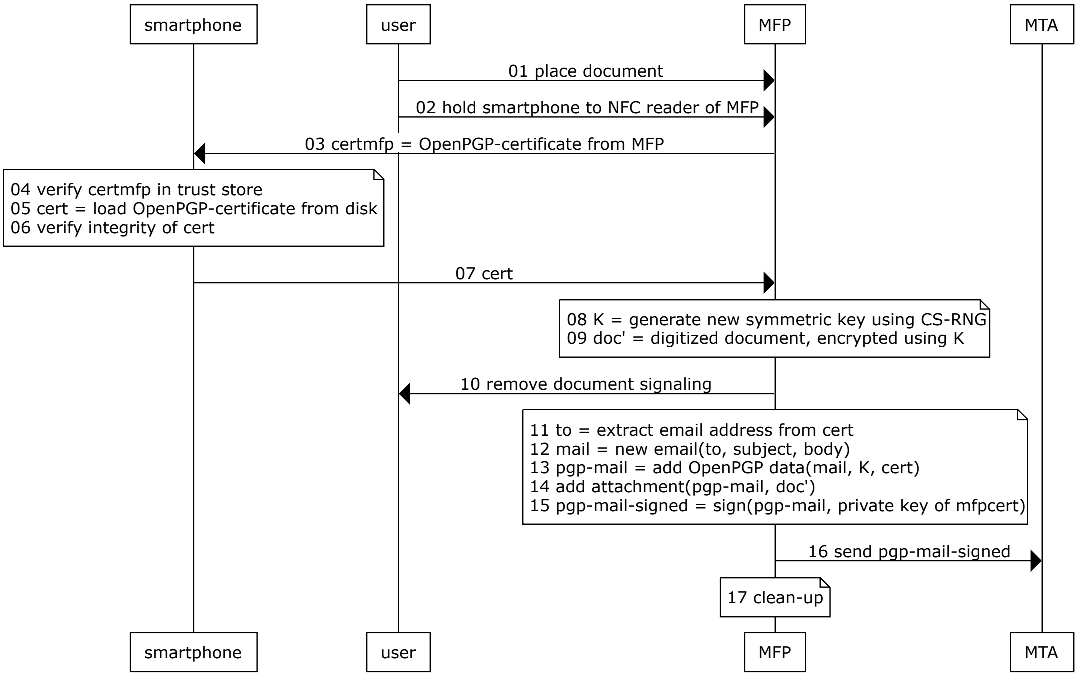
## 8.3. Process overview

This section provides an overview of the secure scanning process. Figure 8.5 shows the components involved in the secure scanning process, while figure 8.6 illustrates the process itself. Next follows a description of the process steps. Note that the selected method for establishing a secure channel using DHE and the process of streamed encryption is not illustrated in the process description, as this has been discussed in previous sections (refer 8.1.3).

**Process steps for secure scanning**

| | |
|---|---|
| **DIG-1** | The user places the physical document on the MFP. |

Figure 8.6: Secure document scanning and email via NFC and email.


smartphone
user
MFP
MTA
01 place document
02 hold smartphone to NFC reader of MFP
03 certmfp = OpenPGP-certificate from MFP
04 verify certmfp in trust store
05 cert = load OpenPGP-certificate from disk
06 verify integrity of cert
07 cert
08 K = generate new symmetric key using CS-RNG
09 doc' = digitized document, encrypted using K
10 remove document signaling
11 to = extract email address from cert
12 mail = new email(to, subject, body)
13 pgp-mail = add OpenPGP data(mail, K, cert)
14 add attachment(pgp-mail, doc')
15 pgp-mail-signed = sign(pgp-mail, private key of mfpcert)
16 send pgp-mail-signed
17 clean-up
smartphone
user
MFP
MTA

**DIG-2** The user positions their smartphone against the MFP's NFC reader, which initiates the secure scanning process. Note that Host Card Emulation (HCE) is used for communication between the smartphone and the MFP and that HCE does not require the user to open the Secure Scan app, as the app's HCE service handler is automatically invoked when the smartphone is held against the MFP's NFC reader.

**DIG-3** The MFP sends its public OpenPGP certificate to the smartphone over NFC.

**DIG-4** The smartphone verifies whether the MFP's public OpenPGP certificate is already registered in the smartphone's trust store of public OpenPGP certificates of trusted MFPs. If not, the user is prompted to confirm the registration of the MFP's public OpenPGP certificate, aligning with the principle of TOFU, see figure 8.2. The following steps assume that the MFP's public OpenPGP certificate is registered.

**DIG-5** The smartphone loads the public OpenPGP certificate from disk.

**DIG-6** The smartphone verifies the certificate's integrity by verifying the cryptographic signature of the certificate using the smartphone bound public key, which belongs to the smartphone bound private key, stored in the smartphone's secure hardware. Note that the signature was created using the smartphone bound private key, and, as such, that a valid signature indicates that the certificate has not been tampered with.

**DIG-7** The smartphone sends the OpenPGP certificate to the MFP over NFC. Note that the NFC channel is encrypted using a session key established by means of DHE, see figure 8.2. Note that the DHE process is omitted from the process description for brevity.

**DIG-8** The MFP generates a symmetric encryption key using its Cryptographically Secure Random Number Generator (CS-RNG). This key is used for encrypting the document and the other email parts, conform the OpenPGP standard. Note that the encrypted document using the symmetric encryption key is already created before the MFP crafts the OpenPGP email message. This is because the MFP uses stream-based encryption, which enables the MFP to encrypt the document page-by-page, as each page is being digitized. As a consequence, when applying encryption to the email parts, the MFP can already attach the encrypted document to the OpenPGP email message.

**DIG-9** The MFP digitizes the document page-by-page and encrypts each page using the encryption key by means of stream-based encryption, which prevents the MFP from caching unprotected fragments of the document to disk.

**DIG-10** Once the MFP is done digitizing the document, it starts it visible and audible warning system to remind the user to remove the original document from the MFP. The signaling continues until the user removes the original document from the MFP.

|  |  |
|---|---|
| **DIG-11** | The MFP extracts the user's email address from the OpenPGP certificate. |
| **DIG-12** | The MFP crafts a new email message, attaches the encrypted document to the email for sending it to the user's email address. |
| **DIG-13** | The MFP uses the public key of the user's OpenPGP certificate to encrypt symmetric encryption key, received from the smartphone, which is then attached to the email message, conforming to the OpenPGP standard. Note that the attachment is not included yet. |
| **DIG-14** | The MFP adds the encrypted document as an attachment to the email message. Note that the attachment has already been encrypted by the MFP using the symmetric encryption key. |
| **DIG-15** | The MFP adds its own OpenPGP certificate to the email message and signs the email message with its private key. |
| **DIG-16** | The MFP submits the signed email message to the MTA for delivery. |
| **DIG-17** | The MFP deletes the temporary files from disk using NIST's CLEAR method [RFW15]. |

## 8.4. Baseline risk mitigation overview

This section provides an overview of the risk mitigations that the secure scanning process provides with regard to the identified threats for the default digitization process. However, note that the design methodology also introduces new risks, as it introduces new components and processes. Those risks will be identified and discussed in the threat mitigation assessment chapter.

**risk 1**: *The threat that a (malicious) user intentionally spoofs another user's email address.*

Our design methodology mitigates this threat by having the user select an existing email address from the smartphone's linked accounts via the smartphone's API, which is then used for the OpenPGP certificate. As such, the user cannot easily spoof an email address that they do not own. Although, it is not impossible, as the user could alter the source code of the mobile app to use a different email address. However, this would require non-trivial technical knowledge. As such, we argue that this threat is mitigated to a reasonable extent.

**risk 2**: *The threat that a user accidentally misspells their email address, which could lead to information disclosure and loss of information.*

The design methodology prevents the user to accidentally misspell their email address, as the user selects an existing email address from the smartphone's linked accounts via the smartphone's API. As such, the user does not have to spell out their email address on the phone or MFP, which mitigates the risk of misspelling.

**risk 3**: *The threat that a (malicious) user denies having typed the email address that was entered on the MFP.*

This is a hard to tackle phenomenon since users of the MFP are anonymous. However, the design methodology mitigates this risk to a reasonable extent, as a consequence of having the user select an existing email address from the smartphone's linked accounts via the smartphone's API. A caveat is, however, when the MFP, aside from secure scanning, also supports regular scanning to email, which allows the user to enter the email address on the MFP. We argue that secure scanning should be the only option for scanning to email, as this mitigates various risks, including this one.

**risk 4**: *The threat that a malicious user repeatedly enters a non-existent email address on the MFP, which could lead to the associated MSA being flagged for spamming.*

While email rate limiting could prevent an MTA from flagging the MFP's email as spam, it would not prevent that a user the manually enters non-existing email addresses repeatedly, potentially leading to the MTA flagging the MFP's email as spam. Although this threat is more hypothetical, as it requires a user to intentionally enter non-existing email addresses repeatedly, using the MFP's UI, we argue that also this threat is mitigated to a reasonable extent by having the user select an existing email address from the smartphone's linked accounts via the smartphone's API. This is because the user cannot enter a non-existing email address, as the user selects an existing email address from the smartphone's linked accounts via the smartphone's API. A caveat is that a crafty user could alter the source code of the mobile app to have the MFP subsequently send email to non-existing email addresses. However, this would require non-trivial technical knowledge.

**risk 5**: *The threat that a malicious actor spoofs the MFP to trick a targeted recipient into believing the organization officially endorses or reviews emails, based on the sender's address.*

This risk is partly mitigated by having the user select an existing email address from the smartphone's linked accounts via the smartphone's API. Note that this is a very targeted attack, and that an attacker probably has easier ways to launch a targeted spoofing attack.

**risk 6**: *The threat that an unauthorized actor could alter or manipulate the contents of the email, as it passes from the MFP through the sender MTA to the recipient's MTA, potentially changing its information or inserting malicious contents.*

This risk is mitigated by having the MFP sign the email with its private key, which is verified by the smartphone using the MFP's public key. Note that the user authenticates the MFP's OpenPGP certificate on first use, by means of an out-of-band channel, refer 8.1.3. As such, the smartphone can verify the authenticity and integrity of the email sent by the MFP, once they have received the email. A valid signature indicates that the email has not been tampered with. It should be noted however that this mitigation relies on the proper protection of the MFP's private key. Once the MFP's private key is compromised, an attacker could sign emails with the MFP's private key, which would not be detected by the smartphone.

**risk 7**: *The threat that an unauthorized actor could access a digitized document that is stored on the MFP, thereby compromising the confidentiality of the document.*

The design methodology employs stream-based encryption, which prevents the MFP from caching unprotected fragments of the document to disk. Even when the MFP scans a document consisting out of multiple pages, the MFP adds each page to the encrypted document stream, which is stored on disk. As such, the MFP does not store unprotected fragments of the document to disk. In addition, after the scanning process, the MFP deletes the (encrypted) temporary file from disk using NIST's CLEAR method [RFW15], which provides reasonable assurance that it cannot be recovered anymore. We argue that this mitigates the risk effectively.

**risk 8**: *The threat that an unauthorized actor obtains the email containing the digitized document due to unencrypted traffic between the MFP and MTA, between MTAs and between MTA and MUA.*

Since the design methodology employs an additional encryption layer, using OpenPGP, the protection of the email's confidentiality does not rely on inter-server encryption when the email passes from one MTA to another.

**risk 9**: *The threat that an unauthorized actor obtains the email containing the digitized document due to the email passing through multiple MTAs.*

The OpenPGP encryption layer protects the email's confidentiality when it is stored or processed by the MTAs that the email passes through.

# 9

# Design methodology: reading

In classical OpenPGP email implementations, the OpenPGP's private key is usually stored on the user's computer where the MUA can access the private key directly. However, storing the OpenPGP's private key on a smartphone, as proposed, introduces several benefits. First, it significantly enhances security by establishing a three-factor security model, which necessitates physical access to the smartphone to be able to decrypt emails (something you have), unlocking the private key using biometric authentication (something you are), and access credentials to the MUA to open the mailbox (something you know). Second, it allows for greater flexibility and convenience, as the key can be easily used across multiple PCs, without the need for repeated exporting and importing of the private key. This could be a significant advantage for users who frequently switch between different PCs or use shared PCs.

Once the user receives the email from the MFP, the user needs to be able to read the OpenPGP-encrypted (and signed) e-mail that contains the document. Email is managed using a MUA, such as Outlook or Thunderbird. This presents three challenges:

1. Although various MUAs can support OpenPGP, our implementation is not a typical OpenPGP implementation, as the private key of the user's OpenPGP key pair resides on the user's smartphone, backed by the smartphone's secure hardware. As a consequence, there is the need to deviate from the typical OpenPGP implementation by the fact that the private key is not directly available to the MUA.

Figure 9.1: Illustration of reading a protected document.

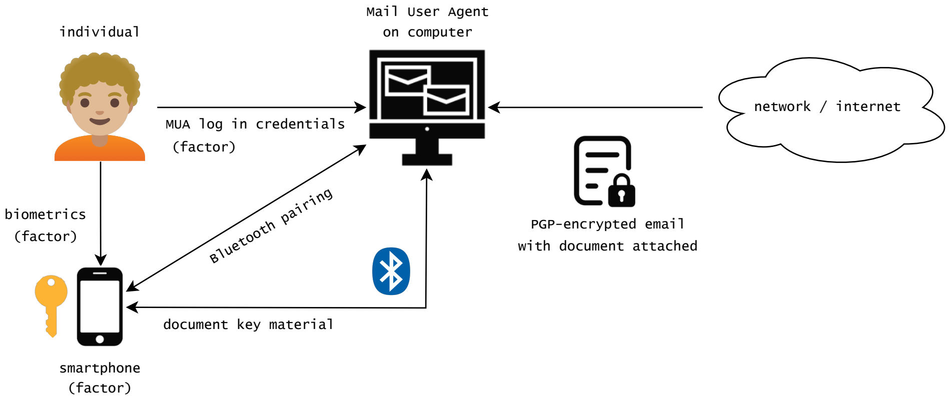
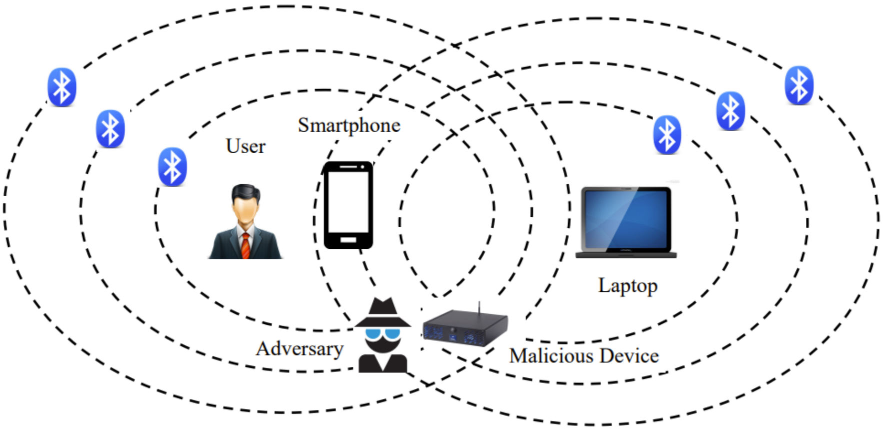


Figure 9.2: Privacy threat scenario in a typical Bluetooth application. Source: [SMS18]



2. Since the private key of the user's OpenPGP key pair is stored on the user's smartphone, the MUA needs to communicate with the smartphone in order to somehow be able to decrypt the email. This requires a confidential communication channel between the MUA and the smartphone.

3. Once the email is decrypted, the MUA needs to handle the secure viewing of the document, which means that the document must not leave unprotected copies or fragments of the document on the PC's hard disk.

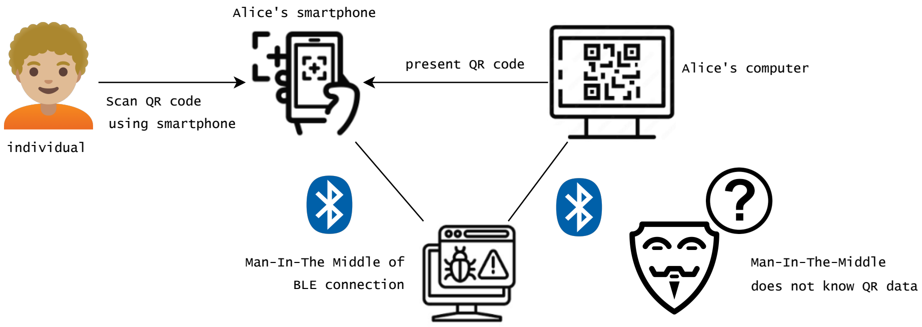
We define secure reading as the process of decrypting the email and viewing the document while ensuring that the process does not expose the document to the operating system or other applications. An illustration of the reading process is shown in figure 9.1.

Note that a standard MUA is not capable of handling the aforementioned challenges, as it is typically not designed to communicate with a smartphone, or, to work with OpenPGP in this way. The functionality of a MUA can typically be extended by means of add-ins, which is a software component that is loaded into the MUA's process memory space and that can extend the functionality of the MUA by utilizing the MUA's Application Programming Interface (API). In the following sections, when we refer to the MUA, we refer to the MUA add-in, unless stated otherwise.

## 9.1. Enhanced OpenPGP implementation

As the private key of the user's OpenPGP certificate's key pair is stored on the user's smartphone, the MUA needs to communicate with the smartphone in order to somehow be able to decrypt the email. This means that a custom OpenPGP implementation is required capable of overriding standard OpenPGP methods. OpenPGP utilizes both symmetric and asymmetric cryptography, where the private key is used for decrypting the symmetric key that was used for encrypting the message. As such, (only) the method of decrypting the symmetric key deviates from a typical OpenPGP implementation. Instead of using the private key directly, the MUA needs to implement this method in such a way that it can communicate with the smartphone in order to send the encrypted symmetric key and receive the decrypted symmetric key, that the MUA can, then, use to decrypt the message. Figure 9.5 illustrates how we envision the OpenPGP implementation to work.

Figure 9.3: Out-of-Band authentication through QR code.



## 9.2. Secure channel between MUA and smartphone

The MUA needs to communicate with the smartphone in order to exchange the email's encryption key. While the MFP utilizes NFC as a communication channel, NFC is not typically available on a PC. Other communication channels, such as Bluetooth and Wi-Fi Direct, are more ubiquitous for PCs. However, a disadvantage of using Wi-Fi Direct is that any established Wi-Fi connection must disconnect first before the computer can connect to the smartphone and establish a peer-to-peer Wi-Fi Direct connection. This is not the case for Bluetooth, because the computer could maintain already established Bluetooth connections while establishing a new connection with the smartphone.

Alternatively, the smartphone and MUA could connect to the same Wi-Fi or wired network in order to exchange the encryption key. However, this typically requires the user to configure the network settings of the MUA and smartphone, which could be a complex task for an average user. Moreover, it also implies that a server must be available on the network to facilitate the communication between the smartphone and the MUA. While the MUA could start a server, firewalls and other network security measures may prohibit or interfere with the communication, making this method also less reliable.

Bluetooth does not require the user to configure complex network settings, apart from enabling Bluetooth on the smartphone and computer, and pairing the devices. This is a more typical task for an average user. Moreover, Bluetooth is more ubiquitous than NFC on computers, which aligns with **requirement 6** (accessibility). Note that PCs equipped with a Wi-Fi adapter are typically also equipped with Bluetooth. As such, we argue that Bluetooth is a more suitable option for communication between the smartphone and the MUA than Wi-Fi Direct or Wi-Fi (or NFC).

Finally, note that it could be possible too, to use a USB cable to connect the smartphone to the computer. Although this could be a viable method for establishing a secure communication channel, USB connections can pose security risks, as they can potentially expose both the smartphone and PC to direct data breaches or malware transfer, for example, see [NYE17b] and [NS23]. BLE, on the other hand, being a wireless technology, naturally has a strong emphasis on security, as it incorporates various features like encryption and secure pairing methods out of the box. As such, we argue that Bluetooth is a more suitable option than USB.

**Bluetooth communication**
Bluetooth supports two main protocols: Bluetooth Low Energy (BLE) and Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR), also called Bluetooth Classic (BLC). Although both protocols could be used in our context, BLE and BLC cater to different use cases. While BLC is typically used for streaming data, BLE is a more efficient protocol that is typically used for (continuously) exchanging small amounts of data, such as between a smartphone and a wearable device. Moreover, BLE sup-

ports various features that are interesting for our use case which are not supported by BLC. These include Received Signal Strength Indicator (RSSI) for proximity detection, BLE scanning and BLE advertising for devices for presence detection, and the notion of publish and subscribe, which allows the smartphone to receive events from the MUA in an efficient manner. As such, we argue that BLE better suits our use case than BLC. Finally, note that the Low Energy part of BLE refers to the fact that BLE is designed to be energy efficient, which aligns with the use of smartphones, since they are typically battery powered.

**Bluetooth security**
The use of Bluetooth is not without security risks. Figure 9.2 illustrates that a malicious device could eavesdrop on the communication between the MUA and the smartphone. In background, see section 2.3, we have established that the security of BLE is complex and that adding an application-level layer of protection should be considered to mitigate various attacks. Relating this to our context, individuals may use older smartphones (and computers) that do not support the most recent version of the BLE specification. Since we intend to use BLE for exchanging the document's encryption key, we argue that employing this extra layer of protection is a prudent measure.

A method to add a layer of protection to the BLE communication channel is to use an Out-of-Band authentication (OOB authentication) step after the BLE pairing process when the MUA detects first use of the paired BLE connection. A feasible OOB method is to have the MUA display a QR code that the user scans with the smartphone's camera. Figure 9.4 illustrates this method. Using this additional channel for authentication provides a reasonable assurance of authenticity, because the user must be physically present at the computer in order to scan the QR code using the smartphone's camera. Note that in our context the user already uses a smartphone and a computer, making this method accessible, which aligns with **requirement 6**. In addition, it is reasonable to assume that many individuals are accustomed to scanning QR codes due to their widespread use.

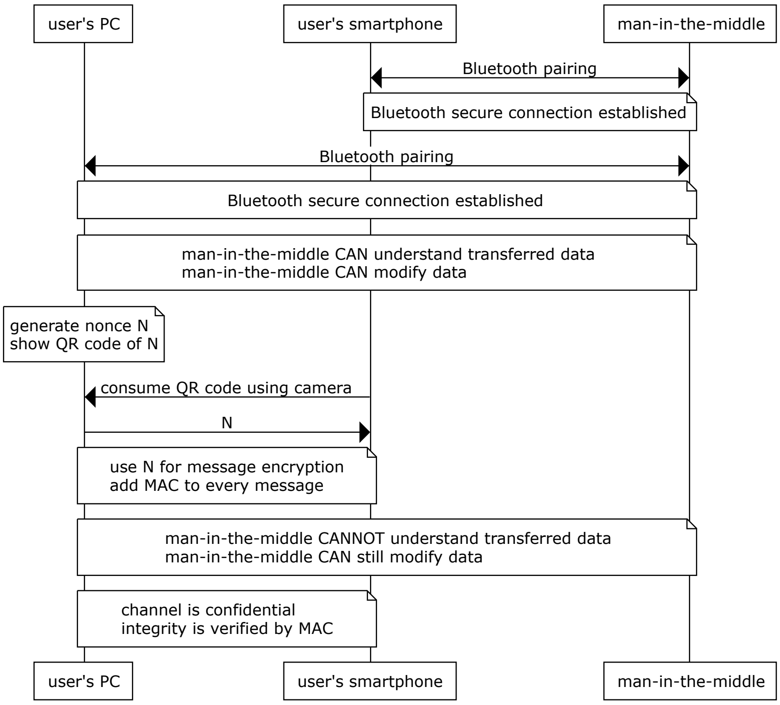
As for the QR's content, it could contain a randomly generated key used for encrypting successive communication between the MUA and smartphone. The combination of using a dual channel for key exchange provides a reasonable assurance of authenticity. As for integrity, the MITM can still alter the transferred data, however, this can be detected by means of a Message Authentication Code (MAC) added to each message, as illustrated in figure 9.4. Instead of utilizing a MAC for integrity, both the MUA and smartphone could also utilize AES in GCM mode, which provides both confidentiality and integrity. We argue that this is a more straightforward approach, as it obviates the need for an extra MAC step.

## 9.3. Document viewing

After the MUA has obtained the encryption key from the smartphone, the MUA can decrypt the document and display it to the user. It is imperative that the document is not copied or stored in an unprotected manner during the viewing process in order to prevent copies from scattering (refer **??**). As such, the MUA cannot rely on the operating system's associated application for viewing the document, as this could potentially expose the document to the operating system or other applications.

To comply with the aforementioned requirement, the MUA could use a designated *external* document viewer that is capable of decrypting the document in-memory and displaying it to the user. However, initiating an external document viewer implies that the document must be copied to the viewer's process memory, which is typically done by the use of a temporary file. This would require that either the temporary file is plain-text or that the encryption key is revealed to the external viewer, for example, by passing it as a command-line argument. Alternatively the MUA could load into the

Figure 9.4: How Out-Of-Band authentication can be used to mitigate MITM attacks.


user's PC
user's smartphone
man-in-the-middle
Bluetooth pairing
Bluetooth secure connection established
Bluetooth pairing
Bluetooth secure connection established
man-in-the-middle CAN understand transferred data
man-in-the-middle CAN modify data
generate nonce N
show QR code of N
consume QR code using camera
N
use N for message encryption
add MAC to every message
man-in-the-middle CANNOT understand transferred data
man-in-the-middle CAN still modify data
channel is confidential
integrity is verified by MAC
user's PC
user's smartphone
man-in-the-middle

external document viewer within its own process memory space, for example, by means of a shared library. This implies that the plain-text document and plain-text encrypted, although exposed to the external viewer component, are not exposed to the operating system or other applications, as they are bound to the MUA's process memory space. However, it is important to note that an external component loaded in the MUA's address space can access the MUA's process memory space, which would be an introduced security risk.

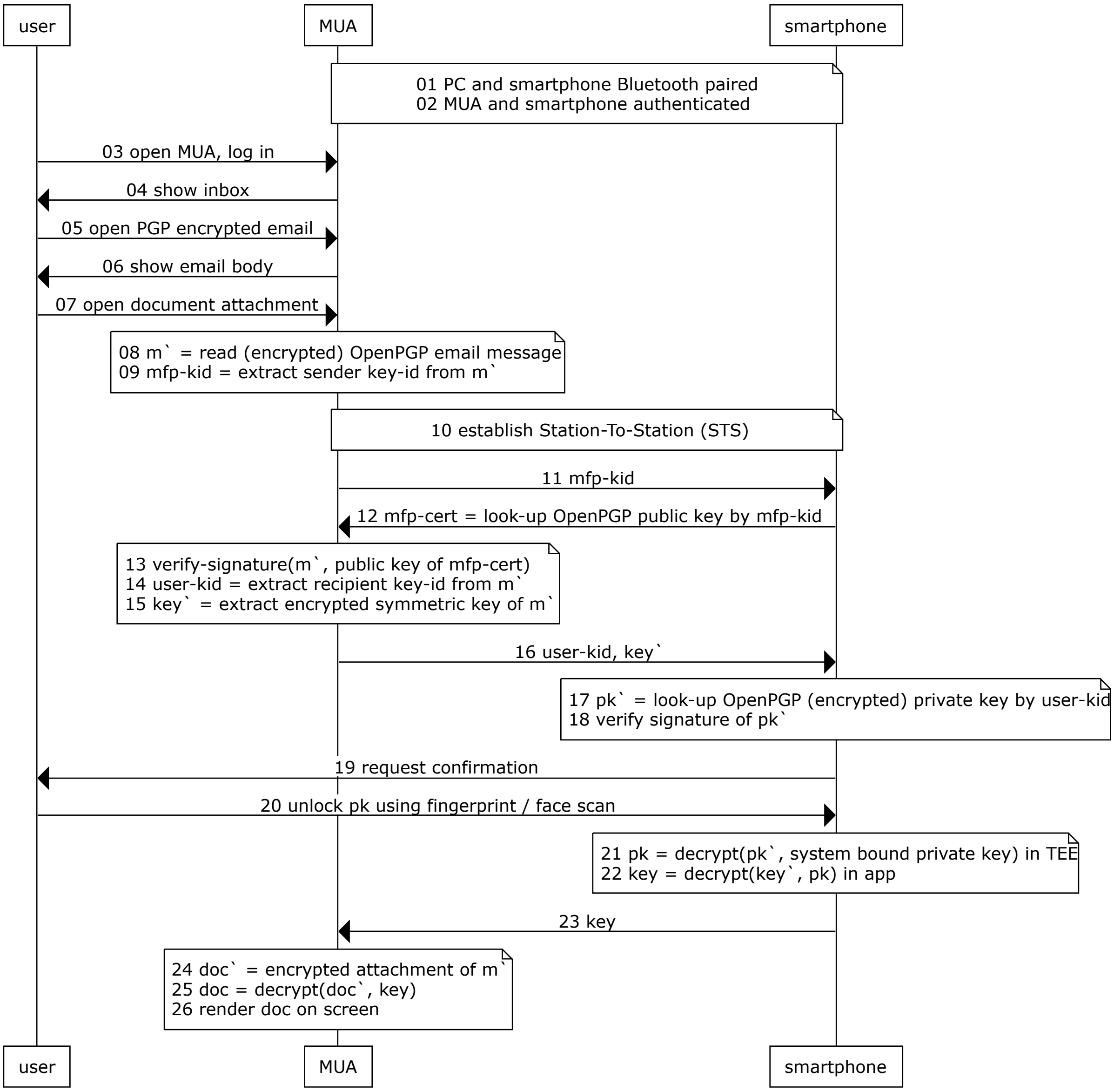
Another option is that the MUA contains a simple built-in document viewer using an open source library, such as `pdfium` [PDF], a library initially created by Foxit and now maintained by Google, or, `pdf.js` [Moz], which is a library created by Mozilla. We argue that this is a more straightforward approach as it obviates the need for installation of an external document viewer that must be configured to work with the MUA. Moreover, a built-in document viewer gives the MUA full control over version management, which mitigates the risk of data leakage. For example, when an external document viewer is updated, it could potentially introduce a vulnerability that could be exploited by an attacker. Tying the MUA add-in together with a built-in document viewer mitigates this risk, as the MUA add-in can be developed in tandem with the document viewer library, giving the developer and security testers full control over the update release process.

## 9.4. Process overview

### 9.4.1. Reading

Figure 9.5 illustrates the process steps for reading the email containing the document. Note that for encryption, after step **RED-10**, AES in GCM mode is used, which provides both confidentiality and integrity. As such, the calculation and verification of MACs is not modelled, as it is not necessary.

Figure 9.5: Process overview of reading a OpenPGP protected document using a smartphone (RED-*).


user
MUA
smartphone
01 PC and smartphone Bluetooth paired
02 MUA and smartphone authenticated
03 open MUA, log in
04 show inbox
05 open PGP encrypted email
06 show email body
07 open document attachment
08 m` = read (encrypted) OpenPGP email message
09 mfp-kid = extract sender key-id from m`
10 establish Station-To-Station (STS)
11 mfp-kid
12 mfp-cert = look-up OpenPGP public key by mfp-kid
13 verify-signature(m`, public key of mfp-cert)
14 user-kid = extract recipient key-id from m`
15 key` = extract encrypted symmetric key of m`
16 user-kid, key`
17 pk` = look-up OpenPGP (encrypted) private key by user-kid
18 verify signature of pk`
19 request confirmation
20 unlock pk using fingerprint / face scan
21 pk = decrypt(pk`, system bound private key) in TEE
22 key = decrypt(key`, pk) in app
23 key
24 doc` = encrypted attachment of m`
25 doc = decrypt(doc`, key)
26 render doc on screen
user
MUA
smartphone

### Process steps of reading the document

**RED-1** When the user uses a MUA or smartphone for the first time, the MUA and smartphone must be paired using the respective operating system's designated BLE pairing methods for both devices. This process typically involves the user to confirm the pairing request on both devices, sometimes by means of a PIN code. This process presupposes that the user has already paired the smartphone with the MUA, using the respective operating system's designated BLE pairing methods for both devices.

**RED-2** Paired BLE devices communicate over a secure channel. However, as established in section 2.3, we fortify the security of the communication channel by means of an additional Out-of-Band authentication step. This step is only performed when the MUA detects first use of the paired BLE connection. This process presupposes that the MUA and smartphone have already established the authentication step. Note that first-time use is found out during our implementation of the Station-To-Station (STS) protocol, as certificates are exchanged that are not yet trusted. See 9.4.2 for more details.

**RED-3** The user opens the MUA and authenticates to the MUA, for example by means of a password or stored credentials or access token on the PC.

**RED-4** The MUA shows the email messages in the user's mailbox.

**RED-5** The user selects an OpenPGP encrypted email message that contains a document.

**RED-6** The MUA opens the email message and shows the user the email body and the names of the attachments. Note that the MFP solely encrypted the attached document, not the email message body. This prevents that the user would have to go through the process of decryption for, both, the email message body and the document. Since the email body output from the MFP is hardly ever confidential, we argue that this is a reasonable trade-off.

**RED-7** The user instructs the MUA to open the encrypted document.

**RED-8** The MUA extracts the OpenPGP metadata from the email message.

**RED-9** The MUA extract the key-id from the email message, which was used for signing the email message. Note that this is the key-id of the MFP's OpenPGP public key.

**RED-10** The MUA and smartphone initiate the STS protocol to establish an authenticated secure channel over BLE. A prerequisite for the STS protocol is that the MUA and smartphone have already established trust in each other's certificates.

Any subsequent communication between the MUA and smartphone is encrypted using the established shared secret. Note that the STS protocol starts with exchanging certificates. When either the MUA or smartphone does not trust the certificate of the other party, the STS protocol is aborted and the process for first-time use is initiated in order to establish trust by authenticating each other's certificates, using the OOB authentication step.

**RED-11** The MUA sends the MFP's key-ID to the smartphone.

**RED-12** The smartphone looks up the public key of the MFP's OpenPGP key pair (mfp-cert) using the key-id. Note that the MFP's public key is stored on the smartphone on first use of the MFP, refer figure 8.2.

**RED-13** The MUA verifies the signature of the email message using the MFP's public key that is has received from the smartphone.

**RED-14** The MUA extracts the key-ID from the email message, which refers to the key-id of the user's OpenPGP key pair used for encrypting the email (user-kid). The key-ID is a unique identifier that is derived from the public key of the user's OpenPGP key pair.

**RED-15** The MUA extracts the encrypted symmetric encryption key from the email message (key‘). Note that the encrypted symmetric encryption key is encrypted using the public key of the user's OpenPGP key pair, for which the private key is stored on the smartphone.

**RED-16** The MUA sends the user's key-id and the encrypted symmetric encryption key to the smartphone.

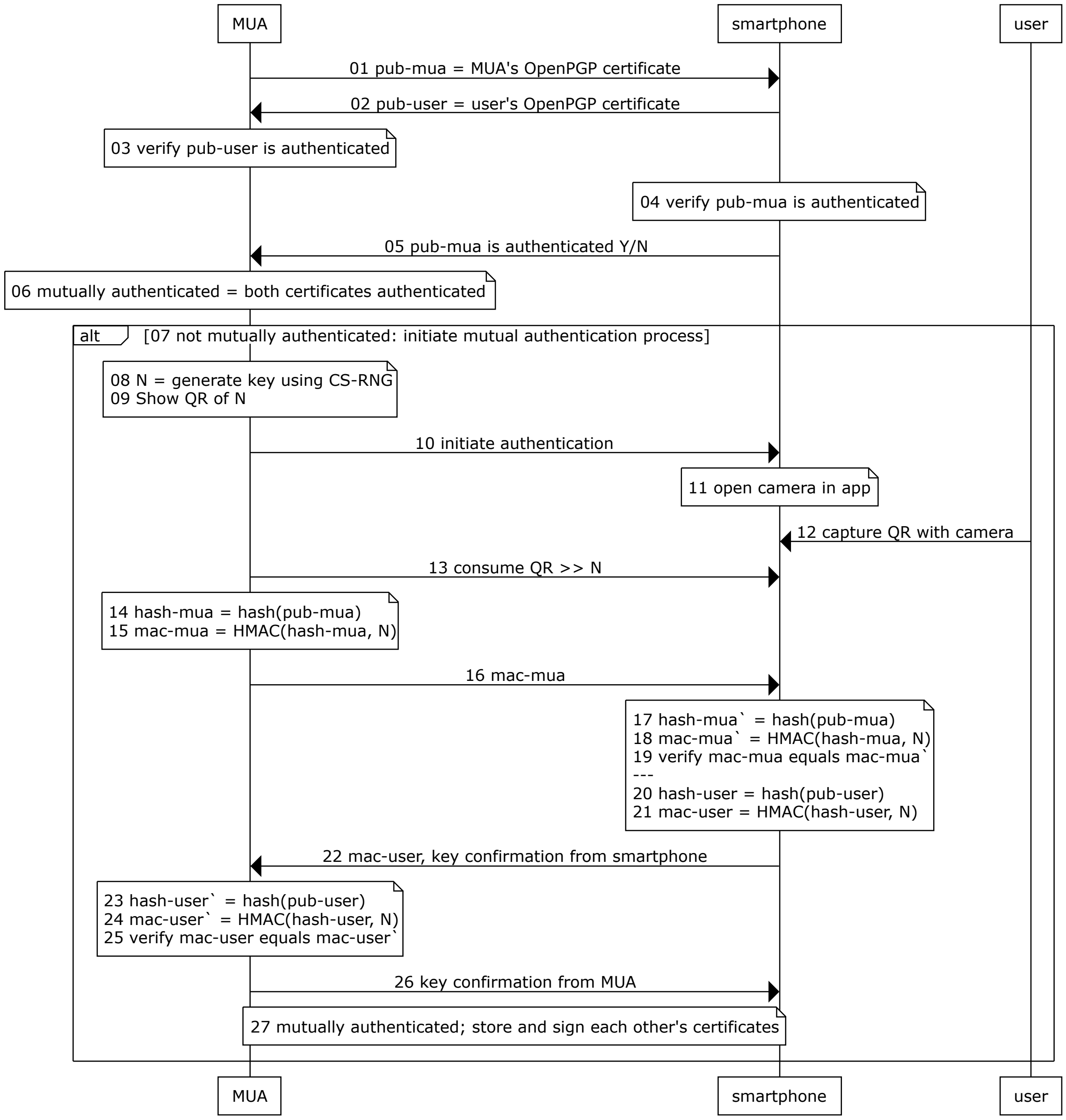
**RED-17** The smartphone looks up the private key of the user's OpenPGP key pair using the provided key-id (pk‘).

**RED-18** The smartphone verifies the signature of pk‘ that is stored along pk‘ using the public key of the user's OpenPGP key pair.

**RED-19** When the signature checks out, the smartphone shows a user notification that the MUA requests access to have the smartphone decrypt the encrypted symmetric encryption key of an email.

**RED-20** The user unlocks their smartphone using biometric authentication, such as a fingerprint or face scan, which, in turn, unlocks the smartphone's secure hardware to use the system bound private key.

Figure 9.6: Out-of-Band authentication using PC’s display and smartphone camera (AUT-*).


MUA
smartphone
user
01 pub-mua = MUA's OpenPGP certificate
02 pub-user = user's OpenPGP certificate
03 verify pub-user is authenticated
04 verify pub-mua is authenticated
05 pub-mua is authenticated Y/N
06 mutually authenticated = both certificates authenticated
alt
[07 not mutually authenticated: initiate mutual authentication process]
08 N = generate key using CS-RNG
09 Show QR of N
10 initiate authentication
11 open camera in app
12 capture QR with camera
13 consume QR >> N
14 hash-mua = hash(pub-mua)
15 mac-mua = HMAC(hash-mua, N)
16 mac-mua
17 hash-mua` = hash(pub-mua)
18 mac-mua` = HMAC(hash-mua, N)
19 verify mac-mua equals mac-mua`
---
20 hash-user = hash(pub-user)
21 mac-user = HMAC(hash-user, N)
22 mac-user, key confirmation from smartphone
23 hash-user` = hash(pub-user)
24 mac-user` = HMAC(hash-user, N)
25 verify mac-user equals mac-user`
26 key confirmation from MUA
27 mutually authenticated; store and sign each other's certificates
MUA
smartphone
user

| | |
|---|---|
| **RED-21** | The smartphone instructs the TEE to decrypt the OpenPGP’s private key (pk‘) using the system bound private key. |
| **RED-22** | The smartphone uses the decrypted private key (pk) to decrypt the encrypted symmetric encryption key (key‘) to obtain the symmetric encryption key (key). |
| **RED-23** | The smartphone sends the symmetric encryption key (key) to the MUA. |
| **RED-24** | The MUA obtains the encrypted attachment (doc‘) from the email message (m‘). |
| **RED-25** | The MUA decrypts the encrypted attachment (doc‘) using the symmetric encryption key (key) to obtain the plain-text attachment (doc). |
| **RED-26** | The MUA renders the document in a built-in document viewer that prevents the document from being copied or stored in an unprotected manner. |

### 9.4.2. Mutual authentication

Before the MUA and smartphone can exchange the document's encryption key, they must mutually authenticate each other to ensure that they are communicating with the intended party. While the Bluetooth pairing process already provides a reasonable assurance of authenticity, we argue that an additional authentication step is necessary to fortify the defense against current and novel attacks. Note that different Bluetooth versions and implementations can be used interchangeably, thereby potentially not mitigating all known attacks, let alone novel attacks.

The absence of mutual authentication is detected during our implementation of the Station-To-Station (STS) protocol, which assumes that the MUA and smartphone have already established trust in each other's certificates, typically by means of a Certificate Authority (CA). However, in our context there is no CA involved, as we use (self-signed) OpenPGP certificates. While the Web of Trust model could be used to establish trust in each other's certificates, we argue that a more direct and practical approach is to use an Out-of-Band authentication step, given that the user already uses a smartphone and computer.

The STS protocol starts with exchanging certificates. When either the MUA or smartphone does not trust the certificate of the other party, the STS protocol is aborted and the process for first-time use is initiated in order to establish trust by authenticating each other's certificates, using the OOB authentication step. This involves generating a secure key (N) displayed as a QR code by the MUA on the PC screen and captured by the smartphone's camera. Both devices then compute and exchange Hash-based Message Authentication Codes (HMACs) derived from their respective certificate hashes and the key (N). They verify each other's HMACs to confirm authenticity. Finally, they send key confirmation messages to each other, and upon successful verification, they mutually authenticate and store and sign each other's certificates, and thereby establishing a trusted relationship for future communications. Figure 9.6 illustrates this process.

**Process steps of Out-of-Band authentication**

**AUT-1** The MUA sends its OpenPGP certificate to the smartphone. This step involves the MUA transmitting its digital certificate, which includes its public key and identity information, to the smartphone for verification. Note that the MUA can create a self-signed certificate on-the-fly, as it is not necessary to have a CA involved, for example, using the computer name and user's email address, opened in the MUA, as identity information, e.g. Johnny@company.com at COMPUTER-NAME. As such, the certificate is bound to the MUA and to a specific email address.

**AUT-2** The smartphone sends the user's OpenPGP certificate to the MUA.

**AUT-3** The MUA verifies whether the user's certificate received from the smartphone is authenticated. This is done by checking if the MUA has a valid signature stored for the fingerprint of the received user's certificate. Note that this signature is created using the MUA's private key, which is stored on the MUA, indicating that the MUA has authenticated the user's certificate previously.

**AUT-4** The smartphone verifies whether the MUA's certificate received from the MUA is authenticated. It does this in the same way as the MUA verifies the user's certificate.

**AUT-5** The smartphone informs the MUA whether its certificate is authenticated (yes or no). The MUA uses this information to determine whether the smartphone has authenticated the MUA's certificate.

**AUT-6** The MUA determines if mutual authentication was previously achieved.

**AUT-7** If (previously) mutually authenticated, the STS protocol continues. If not, the MUA initiates the mutual authentication process. This step is taken if either of the certificates was not previously authenticated. Note that this is the step that deviates from the STS protocol, as the STS protocol assumes that the MUA and smartphone have already established trust in each other's certificates, typically by means of a Certificate Authority (CA).

**AUT-8** The MUA generates a random key using a CS-RNG.

**AUT-9** The MUA displays the generated key N as a QR code on the PC's display.

**AUT-10** The MUA signals the smartphone to initiate the authentication process. This allows the smartphone to prepare for the authentication process by opening the camera in the app to scan the QR code. Note that a service handler on the smartphone that handles the Bluetooth communication could instantiate the app and signal it to initiate the authentication process by automatically opening the camera in the app in the foreground by means of a notification.

**AUT-11** The smartphone opens the camera in the app to scan the QR code. This prepares the smartphone to read the QR code displayed by the MUA.

**AUT-12** The user captures the QR code with the smartphone's camera.

**AUT-13** The smartphone consumes the QR code and extracts the key N from it.

**AUT-14** The MUA computes the hash of its own OpenPGP certificate (hash-mua).

**AUT-15** The MUA computes a HMAC over hash-mua using the shared secret key N (mac-mua).

**AUT-16** The MUA sends the computed HMAC (mac-mua) to the smartphone.

**AUT-17** The smartphone also computes the hash of the MUA's OpenPGP certificate (hash-mua‘).

**AUT-18** Then, the smartphone also computes a HMAC over hash-mua‘ using the shared secret key N (mac-mua‘).

**AUT-19** The smartphone compares the computed mac-mua‘ with the received mac-mua. When they match, it means that the MUA has authenticated itself. If not, the smartphone aborts the authentication process.

**AUT-20** The smartphone computes the hash of its own OpenPGP certificate (hash-user).

**AUT-21** The smartphone computes a HMAC over hash-user using the shared secret key N (mac-user).

**AUT-22** The smartphone sends both the computed HMAC (mac-user) and the explicit message that confirms that the MUA has successfully verified the smartphone's HMAC to the MUA.

**AUT-23** The MUA also computes the hash of the smartphone's OpenPGP certificate (hash-user‘).

**AUT-24** Then, the MUA also computes a HMAC over hash-user‘ using the shared secret key N (mac-user‘).

**AUT-25** The MUA compares the computed mac-user‘ with the received mac-user. When they match, it means that the smartphone has authenticated itself. If not, the MUA aborts the authentication process.

**AUT-26** The MUA sends a key confirmation message to the smartphone. This indicates to the smartphone that the MUA has successfully verified the smartphone's HMAC.

**AUT-27** Both the MUA and smartphone are now mutually authenticated, sign each other's certificates and stores them locally for future communications.

## 9.5. Baseline risk mitigation overview

This section provides an overview of the risk mitigations that the secure scanning process provides with regard to the identified threats for the default digitization process. However, note that the design methodology also introduces new risks, as it introduces new components and processes. Those risks will be identified and discussed in the risk mitigation assessment chapter.

**risk 10**: *The threat that a malicious actor tampers with the email containing the document.*

In the design methodology the MUA verifies the OpenPGP signature of the email message using the MFP's public key. Note that during the digitization process, the user authenticated the MFP's OpenPGP certificate by means of the OOB authentication step. As such, the user can be reasonably assured that the MFP's certificate containing the public key is authentic. Hence, the MUA can be reasonably assured that the email message has not been tampered with.

**risk 11**: *The threat of unauthorized access to the document due to temporary unencrypted files stored on the hard disk as a consequence of viewing the document.*

In the design methodology the MUA incorporates a built-in document viewer that prevents the document from being copied or stored in an unprotected manner. This gives the MUA full control over how the document is rendered and stored in memory. As such, the MUA can prevent the document from being copied or stored in an unprotected manner.

# 10

# Design methodology: sharing

This chapter presents a design methodology: a structured approach aimed at addressing the identified threats and fulfilling the core requirements that have been previously outlined. This methodology serves as the architectural blueprint for developing a system capable of mitigating security risks while balancing stakeholder interests.

The next step in the document's lifecycle is sharing a document with an institution. Note that the context for sharing a document with an institution differs significantly from having the MFP send the document to the individual. This is because institutions typically have an infrastructure in place that can process incoming documents, and, as such, can be extended by facilitating and accepting protected documents coming from their clients (individuals) via email.

Note that, unlike individuals, institutions are bound by stringent regulatory requirements. They are interested in maintaining good reputation and, therefore, ensuring compliance with laws and standards, such as the General Data Protection Regulation (GDPR) and ISO 27000-series, respectively. While this institutional framework aids in protecting individual's data, it primarily pertains to the institution's internal processes. As such, it does not necessarily provide safeguards that would protect the individual from putting their data at risk when sharing them with an institution, in particular when the data are shared via email. However, the institution cannot control the security of the email

Figure 10.1: Envisioned approach for secure document sharing process using email.

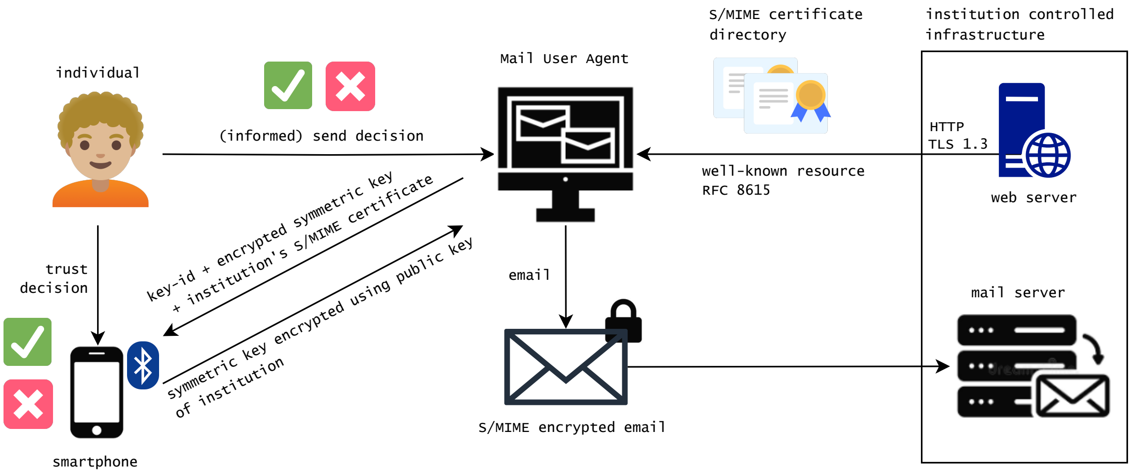
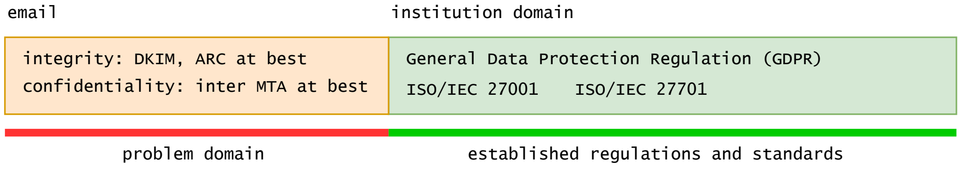

individual
(informed) send decision
Mail User Agent
S/MIME certificate directory
institution controlled infrastructure
HTTP
TLS 1.3
web server
well-known resource
RFC 8615
key-id + encrypted symmetric key + institution's S/MIME certificate
symmetric key encrypted using public key of institution
trust decision
email
mail server
smartphone
S/MIME encrypted email

Figure 10.2: Illustration of the problem domain: while institutions have processes in place to protect the confidentiality and integrity of the document, the email infrastructure used may not be secure.


email
institution domain
integrity: DKIM, ARC at best
confidentiality: inter MTA at best
General Data Protection Regulation (GDPR)
ISO/IEC 27001
ISO/IEC 27701
problem domain
established regulations and standards

infrastructure that individuals use to send emails containing documents.

This chapter proposes a design methodology that addresses this issue, by extending the institution's infrastructure that allows individuals to share documents with the institution via email in a secure manner. We define secure document sharing as the process of sending an email containing a document to an institution in a way that ensures the *confidentiality* and *integrity of the document* attached to the email. We do not provide a means for authentication of the sender, as this is not a strict requirement for the envisioned use case, and, would involve additional complexity. Finally, note that the personal documents an individual shares with an institution may be sent for the purpose of authentication, for example, when sending a copy of a passport to a bank. Figure 10.1 illustrates the envisioned approach for secure document sharing process using email.

For accomplishing secure sharing of documents, several challenges need to be addressed, which include:

- To ensure the confidentiality of the email, the email needs to be encrypted. To this end, the MUA needs to obtain the institution's certificate containing the public key, which it uses for encrypting the email. This key needs to be discovered in an authenticated and secure manner, which is referred to as the *discovery* challenge.
- While some recipients (institutions) may support secure sharing of emails, others may not. The MUA needs to allow the user to make an informed decision about whether to send the email in a protected manner or not, based on the recipient's support for secure sharing of emails. We refer to this as the *informed send decision*.
- Since the email received from the MFP is encrypted and protected by a private key that is only available on the user's smartphone, the MUA needs to communicate with the user's smartphone to perform key operations. With *informed trust decision* we refer to the decision of the user to release the user's private key for performing key operations, based on the recipient's certificate, within the bounds of the smartphone.
- While the recipient's public key covers confidentiality, as it is used for encrypting the email, the integrity of the document needs to be ensured as well. This is typically performed by signing the email using the sender's private key. However, as the sender's private key may not be authenticated, the recipient cannot trust it, and, hence, can also not trust the signature for verifying the integrity of the email.

First, we discuss the problem domain. After that, we address aforementioned challenges, and, finally, we present a process overview.

## 10.1. The problem domain

Figure 10.2 illustrates the problem domain for sharing a document with an institution via email. While the regulations and standards protect the user's privacy and, therefore, the confidentiality of the email with document, the user's email infrastructure may not be secure, which could lead to the email with document being intercepted and tampered with. Since most individuals do not encrypt their emails, refer [RAZS15] and [BLO$^+$15], the document is typically sent in plain text, which implies that the email's confidentiality is protected inter MTA at best, which is not sufficient as each MTA is a potential attack vector. In addition, encrypted communication between MTAs is not a given, as it requires each MTA on the email's path to be properly configured for encrypted communication. For example, when the (opportunistic) STARTTLS protocol is employed the channel would not be encrypted when one party does not support the protocol, or, when a party's certificate has expired. As such, even the protection of the email's *inter MTA* confidentiality is not a given.

As for integrity, the receiving MTA may perform checks on the email, such as checking the DKIM signature, refer RFC 6376 [KCH11], which provides a reasonable assurance that the email has not been tampered with. However, DKIM signatures may break inter MTA, for example, when an MTA modifies the email, such as adding a disclaimer, or, when the MTA relays the email to another MTA. As such, the protection of the email's integrity is also not a given. Finally, ARC is a protocol that could be used to protect the integrity of the email, when the DKIM signature breaks. It allows intermediate MTAs to sign the email, which, in turn, allows the receiving MTA to verify the integrity of the email by following the chain of ARC signatures, refer RFC 8617 [ALBK19]. However, the chain of ARC signatures breaks when an intermediary MTA does not support ARC, or, when ARC is not configured properly. As such, the protection of the email's integrity is still not a given.

Finally, the reliance on the email infrastructure's patchwork of protocols for protecting the email's confidentiality and integrity is not ideal. With regard to confidentiality, the protection is inadequate as it is not end-to-end, but rather inter MTA at best, and, as such, does not protect the email's confidentiality when the email is at rest on the MTA. With regard to integrity, the protection relies on the support and proper configuration of DKIM and ARC, which is not a given.

As for the institution's domain, once the institution has received the email, regulations and standards ensure that the institution's internal processes are secure, and, as such, the confidentiality and integrity of the document are protected.

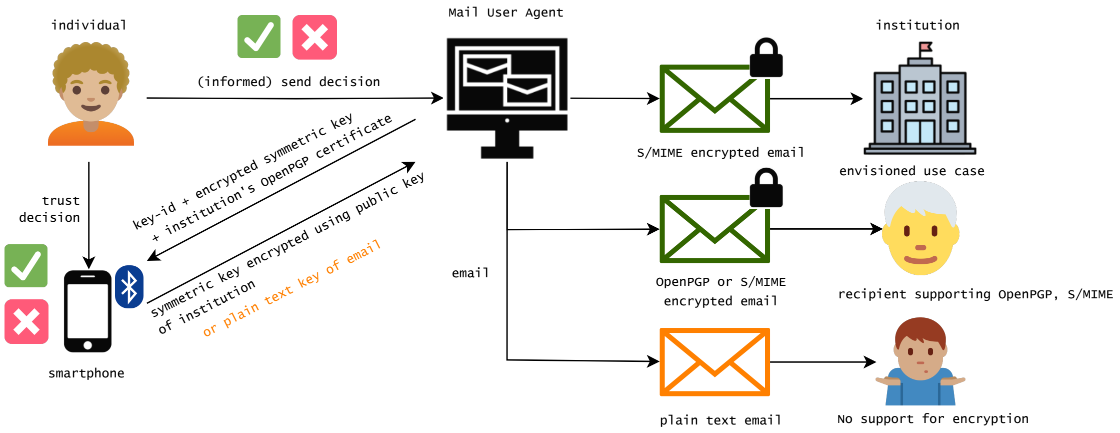
## 10.2. Alternative use cases

While our envisioned use case is sharing a document with an institution while protecting confidentiality and integrity, we also consider the use case where the user shares a document with other parties, as we argue that the user should not be restricted to our envisioned use case. For example, the user may deliberately choose to share a document with a party that does not support secure sharing of emails. This can be accomplished by having the MUA provide default OpenPGP and S/MIME functionality for sending emails, for when secure sharing, as proposed, is not available. Figure 10.3 illustrates these alternative use cases. For the remainder of this chapter, we mainly focus on the envisioned use case, and, where applicable, we also consider the alternative use cases, however, without going into much detail. Note that we use the term *alternative use cases* to refer to the classical OpenPGP and S/MIME use cases, or not using such a secure standard at all (plain text email), while our envisioned use case is referred to as *secure sharing*.

## 10.3. The discovery challenge

The secure sending process starts with the user's intention to forward the (encrypted) email containing the document to the institution. For this, the user opens the email in the MUA, clicks the 'Forward' button and types in the recipient's email address. It is at this point that the MUA could discover whether the recipient institution supports secure sharing of emails or not.

Figure 10.3: Use cases for sharing an email containing a previously digitized document.



individual
(informed) send decision
Mail User Agent
institution
trust
decision
key-id + encrypted symmetric key
+ institution's OpenPGP certificate
symmetric key encrypted using public key
of institution
or plain text key of email
smartphone
email
S/MIME encrypted email
envisioned use case
OpenPGP or S/MIME
encrypted email
recipient supporting OpenPGP, S/MIME
plain text email
No support for encryption

**Institution: S/MIME versus OpenPGP**

Institutions are likely to favor S/MIME over OpenPGP due to its hierarchical trust model, which allows for a clear and unmistakable chain of trust, established through a CA. Moreover, a CA can issue certificates with varying levels of trust. Extended Validation certificates (EV certificates) are an example of this, where the CA performs thorough verification of the requester's identity, which enhances trust even further. OpenPGP does not have such a hierarchical trust model and does not support the notion of EV certificates. Instead, OpenPGP relies on a web of trust, where users can sign each other's public keys to establish trust, or, on a decentralized trust model, where users can trust a public key based on the key's fingerprint. While OpenPGP may be more suitable for individuals, the S/MIME approach is more suitable for institutions. Therefore, we choose to use S/MIME for the envisioned use case, as S/MIME exerts more control over the trust model, which is more suitable for institutions.

**Discovery of the institution's S/MIME certificate**

A method for obtaining the institution's public key is to utilize CERT records in the Domain Name System (DNS). In the DNS, a CERT record is a resource record that can hold a certificate, and is defined in RFC 4398 [Jos06]. When the user types in the institution's email address, the MUA can query the DNS based on the domain part of the entered email address for a CERT record, which holds the institution's certificate. Used in combination with Domain Name System Security Extensions (DNSSEC), which safeguards the integrity of the DNS records, this method ensures that the institution's public key is obtained in a trustworthy manner. However, S/MIME certificates are typically bound to a specific email address, which implies that an institution may have multiple certificates that each need a CERT record in the DNS. We argue that this method is not practical, as an institution may have many email addresses, and, as such, many certificates that periodically need to be renewed.

Another method is to introduce a novel Well-Known URI, refer RFC 8615 [Not19], which could be associated with a directory of S/MIME certificates for a domain. For example, the well-known resource `https://company.com/.well-known/smime` could be used to discover the directory of S/MIME certificates for the domain 'company.com'. While such a directory is mostly static, it needs to be updated from time to time when a certificate is added, renewed or revoked. Having a website serve this resource, as opposed to the DNS, has the advantage that it is easier to update the directory, for example, by means of an automated process. As for authenticity and integrity, note that the well-known resource is served over Hypertext Transfer Protocol Secure (HTTPS), which ensures the confidentiality and integrity of the directory of S/MIME certificates, and, that TLS certificate of the domain provides authenticity of the directory, through the CA. In addition, when the S/MIME certificate is an EV certificate, the CA has performed thorough verification of the requester's identity, which enhances trust even further, when the CA is trusted by the user's operating system. We argue that the latter approach - using a new Well-Known URI - is the better option as it aligns more closely with established standards. Moreover, it provides strong guarantees for authenticity and integrity of the institution's S/MIME certificate that the MUA obtains from the directory.

**Data format of the well-known resource**

The well-known resource could be a JavaScript Object Notation (JSON) document that contains information about the institution, such as the institution's name, logo, address, and a dictionary of email addresses and their corresponding S/MIME certificates. For example:

```
{
    "meta": {
        "type": "smime",
        "version": "1.0",
        "organization": "Company␣Bank",
        "address": "Company␣Street␣1,␣1234␣AB,␣Amsterdam",
        "logo": "https://company.com/logo.png"
    },
    "data": [
        "applications@company.com": "-----BEGIN␣CERTIFICATE-----\nMIIF...\n-----END␣CERTIFICATE-----",
        "inquiries@company.com": "-----BEGIN␣CERTIFICATE-----\nMIIF...\n-----END␣CERTIFICATE-----"
    ]
}
```

The type of the well-known resource is 'smime', which indicates that the directory contains S/MIME certificates. Version 1.0 indicates the version of the well-known resource. This would allow for future extensions, while maintaining backwards compatibility. The organization, address and logo fields can be used by the MUA to present information about the institution to the user for making the sending decision. Finally, 'data' is a dictionary of email addresses and their corresponding active S/MIME certificates.

**IANA registry**

To ensure that the resource is served from a well-known location, we propose to register the well-known resource with the Internet Assigned Numbers Authority (IANA) registry for well-known URIs[1]. This involves drafting a specification for the well-known resource, which is then reviewed by the IANA and, if approved, added to the registry. We argue that this a worthwhile effort, as it would allow for a standardized approach for the automatic discovery of S/MIME certificates for a given email address, which could be generally adopted by MUAs. The actual registration of the well-known resource is considered future work.

## 10.4. Protecting integrity

When the user forwards the email using the MUA, the MUA needs to modify the email before submitting it to the MTA to ensure that the email is sent in a protected manner in a way that the recipient can decrypt the email, extract the document and verify the integrity of the document.

The confidentiality of the email is ensured by encrypting the email using the institution's public key. This is rather straightforward, as the MUA can simply encrypt the email using the institution's public key, which it obtained from the well-known resource, in the standard way of encrypting emails using S/MIME. Protecting the integrity of the document, however, is more challenging, as it is assumed that the user does not have an S/MIME certificate that can be used for signing the email. Although, the user does have an OpenPGP certificate with private key stored on the user's smartphone, this certificate is not authenticated, and, as such, the recipient cannot trust it. So how can the end-to-end

[1] See https://www.iana.org/assignments/well-known-uris/well-known-uris.xhtml (retrieved at 2021-10-14).

integrity of the document be ensured, despite the fact that the user does not have an authenticated certificate?

**Use user's OpenPGP certificate for signing the S/MIME email**

One option is to have the MUA sign the email using the user's OpenPGP certificate and somehow authenticate the user's certificate to the institution. However, this would mix S/MIME (used for encryption) and OpenPGP (used for the signature), which is not desirable, as both techniques rely on different trust models and are not interoperable.

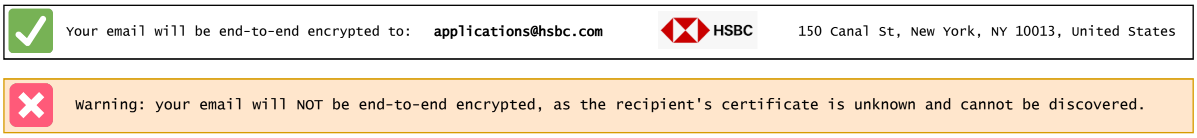
**Use self-signed S/MIME certificate for signing the S/MIME email**

Another option is to have the MUA generate a self-signed S/MIME certificate for the user, which it uses for signing the email, and somehow authenticate that certificate to the institution. However, while using self-signed certificates in internal networks or for testing purposes could be appropriate, using them in production environments, and, over the internet, is discouraged due to their lack of inherent trust from a CA. Even when a self-signed certificate would be authenticated, for example, on a peer-to-peer basis with the institution, this method still lacks the widespread trust and validation provided by a CA. Moreover, this approach would also increase administrative overhead and complexity in managing and verifying these certificates.

**Rely on AEAD for protecting integrity**

A third option is to not protect the integrity of the email, but rather protect the integrity of the document by means of (authenticated) encryption using the institution's public key. While this approach may seem counter-intuitive, we contend that this is the next best option in the absence of an authenticated certificate. This is because the institution can still verify the integrity of the document, even though the email itself is not signed. While this approach would not provide authentication, it would allow the recipient to verify the integrity of the document, which is the primary concern. As such, in the case that the legitimate document of the user arrives at the institution, the institution can verify the integrity of the document, and, therefore, detect tampering. Although, while a malicious actor could, for example, replace the document with a malicious document, without the institution being able to detect this (other than finding that the integrity of the malicious document checks out), we argue that still the confidentiality and integrity of the user's genuine document is protected, as the malicious actor cannot decrypt the document or tamper with it without the institution being able to detect this. With respect to the latter it is important to note that Authenticated Encryption with Associated Data (AEAD) should be employed, such as AES in Galois/Counter Mode (AES-GCM), which provides both confidentiality and integrity (and authenticity of non-encrypted data) in a single cryptographic primitive. This is because other encryption modes, such as AES in Cipher Block Chaining mode (AES-CBC), do not provide integrity, and, as such, would allow a malicious actor to tamper with the document without the institution being able to detect this. This approach achievable, as S/MIME, refer RFC 5751 [TR10], uses the Cryptographic Message Syntax (CMS), refer RFC 5652 [Hou09] for encrypting emails, which supports AEAD, refer RFC 5084 [Hou07]. The approach is also viable, as it does not necessitate customized software at the institution's side, as default S/MIME software implementations should generate an error when the ciphertext is tampered with. So while the approach lacks sender authentication – just like ordinary email, it does provide confidentiality and integrity of the document, which is the primary concern, refer **requirement 7**. We argue that unless the user has an authenticated certificate, this is the next best option for protecting the integrity of the document, and an improvement over the status quo (sending an email with document without end-to-end encryption and end-to-end integrity protection).

Figure 10.4: Example of banners in the user interface of the MUA when the recipient does and does not, respectively, support secure sharing.



✓ Your email will be end-to-end encrypted to: **applications@hsbc.com** HSBC 150 Canal St, New York, NY 10013, United States

✗ Warning: your email will NOT be end-to-end encrypted, as the recipient's certificate is unknown and cannot be discovered.

## 10.5. Informed decision-making in email security

Our methodology aims to provide the user with the ability to make an informed decision by automatically discovering whether the institution supports secure sharing of emails, and, if so, presenting this information to the user prior to sending the email in the MUA.

As sharing the document requires the use of the private key stored on the user's smartphone, the MUA needs to communicate with the user's smartphone to perform key operations. We refer to this as the *informed trust decision*, where the smartphone prompts the user to release the private key for performing key operations, based on the recipient's certificate, within the bounds of the smartphone.

### 10.5.1. The informed send decision

We envision that the MUA presents the user with the information whether the recipient supports secure sharing of emails by means of showing a banner in the user interface of the MUA. See figure 10.4 for an example of banners that a MUA could present to the user, once it has discovered whether there is a certificate available for the recipient. The banner could be shown as soon as the user finishes entering the recipient's email address.

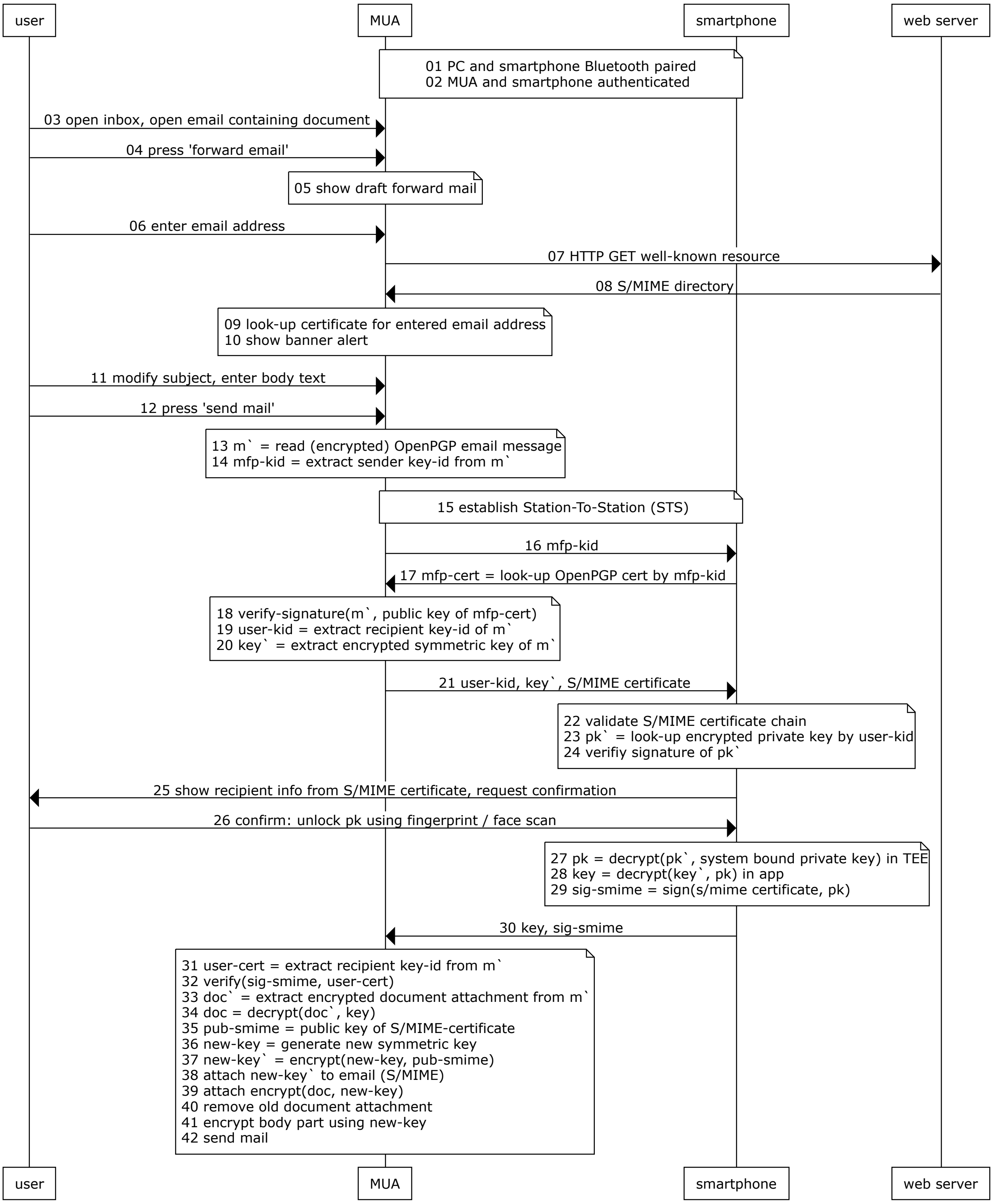
The banner could be accompanied by a button that allows the user to view the recipient's certificate, for example, the EV certificate of the institution obtained from the well-known resource, or, the OpenPGP certificate of the recipient obtained from the user's OpenPGP Keyring, or, the S/MIME certificate of the recipient, obtained from the operating system's certificate store.

### 10.5.2. The informed trust decision

When the user forwards the email using the MUA, the MUA needs to communicate with the user's smartphone to perform key operations, such as decrypting the symmetric encryption key, or, converting the encryption of the symmetric key from the MFP to the institution's public key, for forwarding the email to, respectively, a recipient that *does not* or *does* support secure sharing of emails. The smartphone should provide the user sufficient information about the recipient's certificate, such as the level of trust, for example, is it trusted, revoked, expired, or, is it a valid EV certificate, etc. This allows the user to make an *informed trust decision* about whether to release the private key for performing key operations, based on the recipient's certificate.

Both, the informed send decision and the informed trust decision, are important for minimizing the risk that the user accidentally sends the email without encryption or integrity protection, or, that the user accidentally releases the private key for performing key operations to a fraudulent recipient.

Figure 10.5: Process overview of sharing (SAR-*).


user
MUA
smartphone
web server
01 PC and smartphone Bluetooth paired
02 MUA and smartphone authenticated
03 open inbox, open email containing document
04 press 'forward email'
05 show draft forward mail
06 enter email address
07 HTTP GET well-known resource
08 S/MIME directory
09 look-up certificate for entered email address
10 show banner alert
11 modify subject, enter body text
12 press 'send mail'
13 m` = read (encrypted) OpenPGP email message
14 mfp-kid = extract sender key-id from m`
15 establish Station-To-Station (STS)
16 mfp-kid
17 mfp-cert = look-up OpenPGP cert by mfp-kid
18 verify-signature(m`, public key of mfp-cert)
19 user-kid = extract recipient key-id of m`
20 key` = extract encrypted symmetric key of m`
21 user-kid, key`, S/MIME certificate
22 validate S/MIME certificate chain
23 pk` = look-up encrypted private key by user-kid
24 verifiy signature of pk`
25 show recipient info from S/MIME certificate, request confirmation
26 confirm: unlock pk using fingerprint / face scan
27 pk = decrypt(pk`, system bound private key) in TEE
28 key = decrypt(key`, pk) in app
29 sig-smime = sign(s/mime certificate, pk)
30 key, sig-smime
31 user-cert = extract recipient key-id from m`
32 verify(sig-smime, user-cert)
33 doc` = extract encrypted document attachment from m`
34 doc = decrypt(doc`, key)
35 pub-smime = public key of S/MIME-certificate
36 new-key = generate new symmetric key
37 new-key` = encrypt(new-key, pub-smime)
38 attach new-key` to email (S/MIME)
39 attach encrypt(doc, new-key)
40 remove old document attachment
41 encrypt body part using new-key
42 send mail
user
MUA
smartphone
web server

## 10.6. Process overview

### Process steps of sharing a document

**SAR-1** When the user uses a MUA or smartphone for the first time, the MUA and smartphone must be paired using the respective operating system's designated BLE pairing methods for both devices. This process typically involves the user to confirm the pairing request on both devices, sometimes by means of a PIN code. This process presupposes that the user has already paired the smartphone with the MUA, using the respective operating system's designated BLE pairing methods for both devices.

**SAR-2** Paired BLE devices communicate over a secure channel. However, as established in section 2.3, we fortify the security of the communication channel by means of an additional Out-of-Band authentication step. This step is only performed when the MUA detects first use of the paired BLE connection. This process presupposes that the MUA and smartphone have already established the authentication step. Note that first-time use is found out during our implementation of the STS protocol, as certificates are exchanged that are not yet trusted. See 9.4.2 for more details.

**SAR-3** The user opens their inbox and selects the email containing the document that the user wants to share with an institution.

**SAR-4** The user clicks the 'Forward' button in the MUA.

**SAR-5** The MUA shows a draft email with the document attached in which the user can modify the email's subject and enter the body text, along with the recipient's email address. Note that the MFP solely encrypted the document, and, as such, the email body is not encrypted. Therefore, the user does not yet have to interact with the smartphone.

**SAR-6** The user enters the recipients email address and moves on to the next step: entering the body text.

**SAR-7** The MUA uses the domain part of the entered email address to discover the existence of a well-known resource that contains a directory S/MIME certificates associated with the domain.

**SAR-8** The web server serves the resource to the MUA.

**SAR-9** The MUA parses the resource and extracts the S/MIME certificate for the specific email address from the dictionary. When the S/MIME certificate is found, the MUA validates the certificate, by checking the certificate's metadata, by traversing the certificate chain to the root certificate to ensure that the certificate is valid and by checking the certificate revocation list. Note that the latter requires the MUA to have access to the internet.

**SAR-10** The MUA shows a banner in the user interface of the MUA that indicates that the recipient supports secure sharing of emails by means of showing a green checkmark, refer figure 10.4.

**SAR-11** The user finishes entering the body text.

**SAR-12** The checks the banner stating whether the recipient supports secure sharing of emails and clicks the 'send email' button. When the recipient does not support secure sharing of emails, the MUA should show a warning dialog to the user that the email will be sent in plain text, explicitly asking the user whether to proceed. This allows the user to make an *informed send decision* about whether to send the email in a protected manner or not.

**SAR-13** The MUA interprets the email and assigns the email message to variable m'.

**SAR-14** The MUA extract the OpenPGP key-id from the email message. This key-id refers to the OpenPGP certificate of the MFP that was used for encrypting the document.

**SAR-15** The MUA and smartphone initiate the STS protocol to establish an authenticated secure channel over BLE. Refer to **RED-10** for more details on this step.

**SAR-16** The MUA sends the MFP's key-id to the smartphone.

**SAR-17** The smartphone looks up the public key of the MFP's OpenPGP key pair (mfp-cert) using the key-id. Note that the MFP's public key is stored on the smartphone on first use of the MFP, refer figure 8.2.

**SAR-18** The MUA verifies the signature of the email message using the MFP's public key that is has received from the smartphone.

**SAR-19** The MUA extracts the key-id from the email message, which refers to the key-id of the user's OpenPGP key pair used for encrypting the email (user-kid). The key-id is a unique identifier that is derived from the public key of the user's OpenPGP key pair.

**SAR-20** The MUA extracts the encrypted symmetric encryption key from the email message (key'). Note that the

| | |
|---|---|
| | encrypted symmetric encryption key is encrypted using the public key of the user’s OpenPGP key pair, for which the private key is stored on the smartphone. |
| **SAR-21** | The MUA sends the user’s key-id, together with the encrypted symmetric encryption key and the institution’s S/MIME certificate to the smartphone. |
| **SAR-22** | The smartphone verifies the validity of the institution’s S/MIME certificate by checking its metadata, by traversing the certificate chain to the root certificate to ensure that the certificate is valid and by checking the certificate revocation list. Note that the latter requires the smartphone to have access to the internet. For added security, the smartphone could also check whether the certificate is an EV certificate, which would provide the user with a higher level of trust. |
| **SAR-23** | The smartphone looks up the private key of the user’s OpenPGP key pair using the provided key-id (pk‘). |
| **SAR-24** | The smartphone verifies the signature of pk‘ that is stored along pk‘ using the public key of the user’s OpenPGP key pair. |
| **SAR-25** | The smartphone presents a confirmation dialog to the user about whether to release the private key of the user’s OpenPGP key pair for performing key operations, based on the institution’s S/MIME certificate. The dialog contains information about the institution’s S/MIME certificate, such as the level of trust – is it trusted, revoked, expired, or, is it a valid EV certificate? Note that it is important that the default setting of the smartphone app is to not allow the release of the private key, in the case that something is wrong with the institution’s S/MIME certificate, aligning with the principle of privacy-by-default. |
| **SAR-26** | The user confirms the release of the system bound private key by means of biometric authentication, such as a fingerprint or face scan. The system bound private key refers to the private key stored in the smartphone’s secure hardware. This step refers to the *informed trust decision*. |
| **SAR-27** | The smartphone decrypts the user’s OpenPGP private key (pk‘) using the unlocked system bound private key in the smartphone’s secure hardware, the Trusted Execution Environment (TEE). |
| **SAR-28** | The smartphone decrypts the document’s symmetric encryption key (key‘) using the user’s OpenPGP private key (pk). |
| **SAR-29** | The smartphone creates a signature of the institution’s S/MIME certificate (sig-smime) using the user’s OpenPGP private key (pk). |
| **SAR-30** | The smartphone sends, both, the plaintext symmetric encryption key (key) and the signature of the institution’s S/MIME certificate (sig-smime) to the MUA. |
| **SAR-31** | The MUA obtains the user’s OpenPGP certificate (user-cert) from the user’s OpenPGP key ring using the recipient’s key-id from the email message (m‘). Note that, in the context of the original email message m‘, ’recipient’ refers to the user. Also note that the user’s certificate is stored within the MUA’s OpenPGP key ring on first use of the MUA, refer figure 8.2. |
| **SAR-32** | The MUA verifies the signature of the institution’s S/MIME certificate (sig-smime) using the user’s OpenPGP certificate (user-cert). This proofs to the MUA that the user has released the private key of the user’s OpenPGP key pair for performing key operations, in relation to the institution’s S/MIME certificate. |
| **SAR-33** | The MUA reads the encrypted document attachment (doc‘) from the email message m‘. |
| **SAR-34** | The MUA decrypts the document attachment (doc‘) using the plaintext symmetric encryption key (key). |
| **SAR-35** | The MUA extracts the public key that is stored in the institution’s S/MIME certificate (pub-smime). |
| **SAR-36** | The MUA generates a random symmetric encryption key (new-key) using the PC’s CS-RNG. This key will be used for symmetrically encrypting the email message for the institution. |
| **SAR-37** | The MUA encrypts new-key using the institution’s public key (pub-smime). |
| **SAR-38** | The MUA attaches the encrypted key (new-key‘) to the email message, conforming to the S/MIME standard. |
| **SAR-39** | The MUA encrypts the descrypted document attachment (doc) using the new-key and adds it to the email message, conforming to the S/MIME standard. |
| **SAR-40** | The MUA removes the old document attachment (doc‘) from the email message. |
| **SAR-41** | The MUA encrypts the email body part using new-key and replaces it in the email message, conforming to the S/MIME standard. |
| **SAR-42** | The MUA sends the email message to the MTA. |

## 10.7. Baseline risk mitigation overview

This section provides an overview of the risk mitigations that the process of sending an email to an institution provides with regard to the identified threats for the default digitization process. However, note that the design methodology also introduces new risks, as it introduces new components and processes. Those risks will be identified and discussed in the risk mitigation assessment chapter.

---

**risk 2**: *The threat that a user accidentally misspells their email address, which could lead to information disclosure and loss of information.*

---

This threat also applies to the process of sending an email to an institution. The user could misspell the recipient's email address, which could lead to the email being sent to a fraudulent recipient. However, the risk is mitigated by means of the *informed send decision* and the *informed trust decision*, which warn the user when the recipient does not support secure sharing of emails.

---

**risk 6**: *The threat that an unauthorized actor could alter or manipulate the contents of the email, as it passes from the MFP through the sender MTA to the recipient's MTA, potentially changing its information or inserting malicious contents.*

---

Although the email is not signed due to the absence of the user's S/MIME certificate, the risk is mitigated by employing AES in Galois/Counter Mode (AES-GCM) for encrypting the email, which is an Authenticated Encryption with Associated Data (AEAD) mode of operation and supported by the S/MIME standard, refer [Hou07] and [SRT19]. This provides both confidentiality and integrity in a single cryptographic primitive, ensuring that any tampering with the email is detected by the institution while decrypting the email.

Note that since we do not sign the email, it could be possible for a malicious actor to *add* malicious contents to the email, which would not be detected by the institution. Although this is not a threat to the confidentiality and integrity of the user's genuine document, it could be a threat to the institution. However, we argue that this is not a new concern, as the institution is already vulnerable to this threat when receiving ordinary emails, and, as such, this is not a new risk introduced by our methodology.

---

**risk 8**: *The threat that an unauthorized actor obtains the email containing the digitized document due to unencrypted traffic between the MFP and MTA, between MTAs and between MTA and MUA.*

---

This risk is mitigated by means of encrypting the email, which provides an additional layer of confidentiality protection. As such, the email is still protected when typical email security measures are not in effect, such as the absence of TLS as a consequence of the opportunistic nature of STARTTLS.

---

**risk 9**: *The threat that an unauthorized actor obtains the email containing the digitized document due to the email passing through multiple MTAs.*

---

This risk is mitigated by means of encrypting the email, which protects the email's confidentiality when the email is at rest on the MTA.

# 11

# Design threat model

This chapter presents the threat model for the proposed design. While the design mitigates the threats identified in the baseline threat model, refer chapter 6, it also introduces new threats, as a consequence of the design choices made. These threats are presented in this chapter.

## 11.1. Newly introduced threats

### 11.1.1. Use of an MFP for secure digitization

As our design enhances the security of the MFP, it is important to consider the new threats that arise from this decision.

#### Spoofing

The MFP itself cannot be easily spoofed, as the MFP is a physical device while the user interacts with the MFP directly.

#### Tampering

Tampering with the MFP is a serious security concern, perhaps even more pronounced in the context of the proposed design.

**design risk 1** **The threat that an attacker alters the MFP's firmware.**

An attacker may manipulate the MFP to install malicious firmware, potentially stealing data and breaching the confidentiality of documents. While this risk already exists without our design, our design aggravates the risk, as the MFP is now used for secure digitization. Therefore, we model this risk in the design threat model rather than in the baseline threat model.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | Malicious firmware could circumvent the security mechanisms of the MFP, and steal data from the MFP, which could compromise the confidentiality of documents. |
| *Reproducibility* | High | The attacker can reproduce the attack on multiple MFPs that are vulnerable to the attack of uploading malicious firmware. |

| DREAD element | Risk level | Description |
|---|---|---|
| *Exploitability* | Low | The attacker needs physical access to the MFP to upload malicious firmware. Moreover, the attacker needs to know the exact model of the MFP, and the firmware version, and, build a malicious firmware image that is compatible with the MFP. This is not a trivial task. Note that many MFPs have security mechanisms in place to prevent uploading malicious firmware, such as requiring a signed firmware image, that would require the attacker to have access to the private key of the MFP manufacturer. Nonetheless, it could be possible to circumvent these security mechanisms, for example, by exploiting a vulnerability in the firmware update mechanism. |
| *Affected users* | High | Once the attack is successful, all users of the MFP that are using the MFP for secure digitization are potentially affected. |
| *Discoverability* | Medium | An attacker can easily discover the attack, as the attacker can recognize the MFP model from the outside, as models are typically printed on the outside of the MFP. Moreover, the attacker could discover the firmware version by navigating the MFP's user interface, but only, if the MFP is not protected against this type of unauthorized access. |

**Conclusion design risk 1**
The identified risk of firmware tampering in MFPs is notably significant, especially in the context of our design which involves secure digitization. The potential for an attacker to install malicious firmware raises serious concerns about data theft and the compromise of document confidentiality. This risk is exacerbated in our design due to the enhanced use of the MFP.

The MFP manufacturer's commitment to security is essential for mitigating this risk, which is expressed by regular firmware updates that address security vulnerabilities, and, by ensuring that the firmware update mechanism itself is secure.

### Repudiation

Anonymous users can use the MFP for secure digitization, which has implications for repudiation, such as the inability to identify the user that performed the secure digitization, or, when it matters, the inability to identify the user that breached the MFP's security. However, the risk of repudiation is not new, as it already exists without our design, as anonymous users could utilize the MFP for regular scan-to-email functionality.

### Information disclosure

**design risk 2** **The threat that an attacker intercepts the NFC communication between the MFP and the smartphone.**

An attacker capable of intercepting the NFC communication between the MFP and the smartphone could, for example, eavesdrop on the communication, alter the communication, redirect the communication, or, block the communication.

| **DREAD element** | **Risk level** | **Description** |
|---|---|---|
| *Damage potential* | Medium | As NFC communication is encrypted, the attacker is unable to eavesdrop on the communication, unless the attacker is able to perform a MITM attack. In that case the attacker could establish a secure channel with, both, the MFP and the smartphone, and, intercept the communication for eavesdropping, altering or blocking the communication.<br>When the attacker blocks the communication, the user is unable to use the MFP for secure digitization. As a consequence the user may resort to regular scan-to-email functionality, which is less secure, potentially compromising the document's confidentiality.<br>When the attacker eavesdrops on the communication, the attacker learns the user's OpenPGP certificate, as this is transmitted from the smartphone to the MFP. Although the public certificate is hardly a secret, the attacker could use the public certificate to send encrypted emails to the user, which the user may interpret as legitimate emails from the MFP, which is spoofing. |
| *Reproducibility* | High | An attacker capable of performing a MITM attack on NFC communication can reproduce the attack on multiple MFPs that are vulnerable to the attack. |
| *Exploitability* | Low | While MITM attacks on NFC communication are possible, they are not trivial to perform due to NFC's inherent characteristics. For example, physical barriers and proximity may be limiting factors for the attacker. Moreover, the attacker needs to have specialized equipment to perform the attack, which is an additional limiting factor. |
| *Affected users* | High | Once the attack is successful, all users of the MFP that are using the MFP for secure digitization are potentially affected. |
| *Discoverability* | High | An attacker can easily discover the attack, as the attacker can easily recognize an NFC capable device. |

**Conclusion design risk 2**

The risk of MITM attacks on the NFC communication between the MFP and smartphone is significant as it could compromise the confidentiality of documents, and, it could be used for spoofing. However, the risk is mitigated by the low exploitability of such attacks. NFC's physical and technical constraints, along with the need for specialized equipment, serve as substantial barriers to potential attackers. Therefore, while the risk is significant, it is not likely to be exploited in practice.

#### Denial of service

Denial of service is already addressed as a potential consequence of MITM attacks on NFC communication, refer to risk **design risk 2**.

#### Elevation of privilege

Elevation of privilege is not applicable in this context.

### 11.1.2. Use of a smartphone for storing private keys

As our design is based on the use of a smartphone for storing private keys and managing access to them, it is important to consider the new threats that arise from this decision.

### Spoofing

**design risk 3** **The threat that an attacker spoofs the Mail User Agent (MUA) to obtain key material from the user's smartphone.**

An attacker may spoof the MUA to trick the user into performing actions that the user did not intend to perform, such as trick the user into releasing the private key on their smartphone.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | When the attacker is able to successfully spoof the MUA and trick the user into releasing the private key on their smartphone, then the attacker can decrypt an email, which compromises the confidentiality of the email's contents. |
| *Reproducibility* | High | Once an attacker knows how to spoof the MUA, the attacker can reproduce the attack on multiple MUAs. |
| *Exploitability* | Low | The MUA communicates with the smartphone over Bluetooth. The design requires, both, the user's PC running the MUA, and the user's smartphone to be paired together. In addition to that, the user's smartphone and MUA mutually authenticate each other after pairing, on first use through an out-of-band channel. This makes it difficult for an attacker to spoof the MUA, as the attacker would have to obtain access to the user's PC. |
| *Affected users* | Low | As the attack is targeted, the attacker can only affect the user of the MUA that the attacker is spoofing. |
| *Discoverability* | Low | An attacker cannot easily discover the attack, due to the secure communication and mutual authentication between the user's smartphone and the MUA. Since the attack requires access to the user's PC, which is a significant barrier, it's less likely that an attacker can easily identify or exploit this vulnerability without substantial effort or specific knowledge about the user's system. |

**Conclusion design risk 3**

Although, the risk of spoofing the MUA is significant, as it could compromise the confidentiality of emails, the risk is mitigated by the low exploitability of such attacks.

### Tampering

**design risk 4** **The threat that an attacker tampers with the Mail User Agent (MUA) software to collect key material or plaintexts of protected emails.**

An attacker may tamper with the MUA software to collect key material or plaintexts of protected emails.

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | High | When the attacker is able to successfully tamper with the MUA software, the attacker could collect key material or plaintexts of protected emails, which compromises the confidentiality of the email's. |
| *Reproducibility* | High | Once an attacker knows how to tamper with the MUA software, the attacker can reproduce the attack on multiple MUAs. |
| *Exploitability* | Medium | A limiting factor for an attacker to tamper with the MUA software is that the attacker would have to obtain access to the user's PC. In addition, the attacker would have to know how to tamper with the MUA software, which is not trivial. However, once the attacker has access to the user's PC, the attacker can tamper with the MUA software, which poses a significant risk. |
| *Affected users* | Low | As the attack is targeted, the attacker can only affect the user – or users – of the MUA that the attacker is tampering with. |
| *Discoverability* | Low | An attacker cannot easily discover the attack, as the attacker would have to obtain access to the user's PC, which is a significant barrier. |

**Conclusion design risk 4**
The risk of tampering with the MUA software is significant, as the MUA works with secret key material of documents, obtained from the user's smartphone. The notion that the user would have to approve the release of the private key on the smartphone for each request would not mitigate a stealthy attack that collects the plaintext symmetric document keys or decrypted emails. While the risk is mitigated by the low exploitability of such attacks, the risk is still significant. Additional security measures that protect the MUA software from tampering, such as the use of digital code signing and physical security measures, should be employed to further mitigate the risk.

### Repudiation

Repudiation is not applicable in this context, as unauthorized access and data compromise are the primary concerns.

### Information disclosure

Information disclosure is a significant concern in this context, as the MUA works with secret key material of documents, obtained from the user's smartphone. The risk that the MUA leaks the secret key material is significant, as it could compromise the confidentiality of emails. This risk is addressed under **design risk 4**.

### Denial of service

Denial of service is not a direct concern in this context, as unauthorized access and data compromise are the primary concerns.

### Elevation of privilege

Elevation of privilege is not applicable in this context, as there is only regular user access between the MUA and the smartphone.

### 11.1.3. Security of aged encrypted emails

#### Information disclosure

**design risk 5** **The threat that an attacker is able to decrypt aged encrypted emails as a consequence of obsolete encryption algorithms, key sizes and advancements in computing power and cryptanalysis.**

| DREAD element | Risk level | Description |
|---|---|---|
| *Damage potential* | Medium | The attacker is able to decrypt ages encrypted documents, which compromises the confidentiality of the email's contents, e.g. the document. However, the age of the document may be a limiting factor to the damage potential, as the document may no longer be relevant, although, advances in cryptanalysis, on the other hand, may significantly reduce the time required to decrypt the document. |
| *Reproducibility* | High | The attacker can reproduce the attack on multiple emails that the attacker has access to. |
| *Exploitability* | Low | Emails are typically stored on the Mail Delivery Agent (MDA) and the user's Mail User Agent (MUA). An attacker would have to obtain access to these systems to obtain the encrypted emails, before the attacker can attempt to decrypt the emails. |
| *Affected users* | Medium | An attacker that obtained access to an MDA could potentially obtain access to all emails of all users of the MDA, while an attacker that obtained access to a user's MUA could potentially obtain access to all emails of that user. |
| *Discoverability* | Low | An attacker can only discover the attack when the attacker has access to the encrypted emails. |

**Conclusion design risk 5**
The security of aged encrypted emails is a concern, as the confidentiality of the email's contents may be compromised in the future. However, mitigating factors may be the age of the document, as the document may no longer be relevant and how well to MDA and MUA are secured.

## 11.2. Comparing the baseline and design threat model

In this section we compare the baseline and design threat model, and, assess to what extent the newly introduced threats weigh up against the mitigated threats.

The baseline model concentrates on threats that exist in the current process of digitization and sharing of documents. The design model, on the other hand, focuses on threats that are introduced by the proposed design, which mitigates many of the threats in the baseline model. While the baseline model is typically concerned with threats that are more basic in nature, the design model shifts focus to more sophisticated threats that have become more prominent due to the mitigation of the baseline threats, as a consequence of incorporating security measures in the design. These measures include the use of encryption, authentication and authorization. The added complexity of these measures, however, introduces new threats, such as the risk of tampering with the MFP's firmware, or, the risk of spoofing the MUA. The design threat model addresses these threats, and, proposes additional security measures to mitigate these threats.

# 12

# Conclusions

This chapter concludes the thesis. It begins with the results which concerns to main research question on how to protect the confidentiality of documents in the document's lifecycle.

## 12.1. Summary

This section summarizes the methodology, results and contributions yielded by this thesis.

### 12.1.1. Summary of methodology

We adopted a systematic approach, beginning with the creation of a typical process model that outlined the steps for digitizing a document using an MFP and emailing it to an institution. This was followed by the development of a baseline threat model to identify threats to document confidentiality, employing the STRIDE model for structure and the DREAD model for risk assessment. Simultaneously, we identified key stakeholders and their interests in the digitization and sharing of documents. This informed our analysis of requirements for a privacy-enhancing system aimed at protecting the document's confidentiality and integrity. Our next step involved developing a system design methodology for each process – digitizing, reading, and sharing, again, with the objective of safeguarding confidentiality and integrity. This process included evaluating various methods, selecting the most appropriate ones and integrating these into a comprehensive process model. After establishing the design methodologies, we conducted a reassessment of the established threats, ensuring that the proposed design methodologies were effective in mitigating or, at least, reducing the identified threats.

### 12.1.2. Summary of the threat model

While the baseline threat model focuses on general risks like unauthorized data access, the design model addresses more complex threats associated with advanced security technologies, as a consequence of the design choices made. These include exploiting weaknesses in modern Multi-Function Printers, safeguarding against sophisticated attacks on smartphone-based key exchanges, and ensuring the integrity of encrypted emails over time. These threats are more complex and less likely to be exploited in practice, but they are still significant and should be considered when designing a secure system.

### 12.1.3. Summary of contributions

The main contributions of this thesis are:

- A proposed design that enables individuals to seamlessly digitize a document using an MFP to (their) email, while maintaining the document's confidentiality. The proposed design uses the individual's smartphone to store and protect secret keys.
- A proposed design that enables individuals to forward an email containing a document to an institution, while maintaining the document's confidentiality by means of seamless end-to-end encryption. The proposed design uses the individual's smartphone to perform cryptographic operations for encrypting and decrypting the email.
- A proof-of-concept prototype for the secure digitization to email proposed design, which further demonstrates the feasibility of the approach.

Other contributions:

- We have conveyed the idea of a novel usage scenario for OpenPGP that incorporates the use of a smartphone app to perform cryptographic operations, whereby the secret key, stored on the smartphone, is protected by the smartphone's secure hardware, and, as such, cannot leave the smartphone[1]. This contrasts sharply with the traditional usage scenario, where the secret key is stored on the computer. It allows for flexibility as the user could manage their keys centrally on their smartphone, and use them on multiple MUAs, such as on multiple devices, e.g. a computer or tablet.
- We have conveyed the idea of seamless end-to-end encryption of emails in which is a potential configuration-free method for encrypting emails, transparent to the user. It works by utilizing a proposed Well-Known URI that enables a MUA to discover an institution's S/MIME certificate based on the domain part of the institution's email address, for a given email address that the user entered.

### 12.1.4. Generalizability

Although somewhat constrained by the specific focus on individuals, MFPs, email and institutions, the results and contributions of this thesis are generalizable to other contexts. For example, the concept of automatic authenticated certificate discovery by means of a Well-Known URI could also be applied in corporate settings where workers could seamlessly exchange end-to-end encrypted emails with other organizations.

## 12.2. Future work

A part of the document life cycle that we did not address is the processing of documents by institutions. This is a complex topic that requires a thorough understanding of the institution's context and legal requirements. For example, it may be unacceptable for an institution to work with user-enforced Digital Rights Management (DRM) technologies, as it may be in conflict with the institution's legal obligations. Future work could focus on developing a design methodology for secure processing of documents by institutions, which aids institutions in complying with laws, regulations and standards, while maintaining the document's confidentiality. This future work could be done by a joint effort of students from different disciplines, such as law, business administration and computer

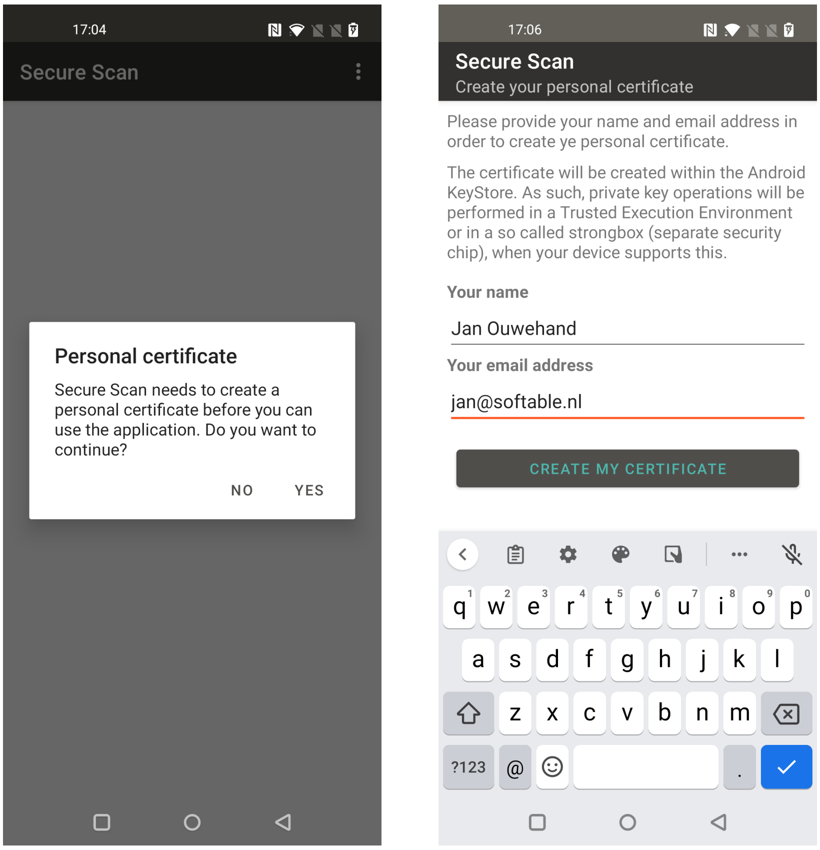
[1] We refer the system bound key that is stored in the smartphone's secure hardware, which can only be used for cryptographic operations within the smartphone's Trusted Execution Environment (TEE). Note that we purposely decided to store OpenPGP private keys in encrypted form outside the TEE, as a trade-off between security and flexibility.

science.

The proposed methods for authentication between the MFP and the smartphone, and between the smartphone and the MUA, could be significantly improved by connecting the proposed design onto the IRMA framework [AvdBH+17]. This would obviate the need for the specific authentication steps, providing the user with a more seamless experience. This future work could be done by a student in collaboration with the IRMA team.

# A

# PROTOTYPE SCREENSHOTS



Personal certificate

Secure Scan needs to create a personal certificate before you can use the application. Do you want to continue?

Secure Scan

Create your personal certificate

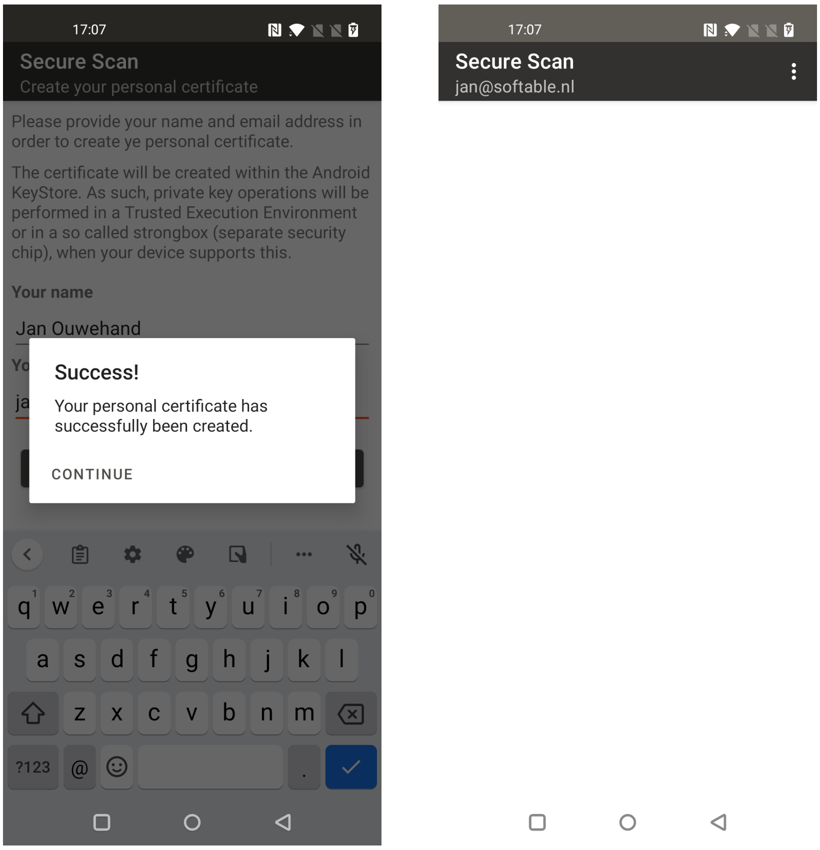
Please provide your name and email address in order to create ye personal certificate.

The certificate will be created within the Android KeyStore. As such, private key operations will be performed in a Trusted Execution Environment or in a so called strongbox (separate security chip), when your device supports this.

Your name: Jan Ouwehand

Your email address: jan@softable.nl

Figure A.1: First time start (left) and creating certificate (right)



17:07

Secure Scan

Create your personal certificate

Please provide your name and email address in order to create ye personal certificate.

The certificate will be created within the Android KeyStore. As such, private key operations will be performed in a Trusted Execution Environment or in a so called strongbox (separate security chip), when your device supports this.

Your name

Jan Ouwehand

Success!

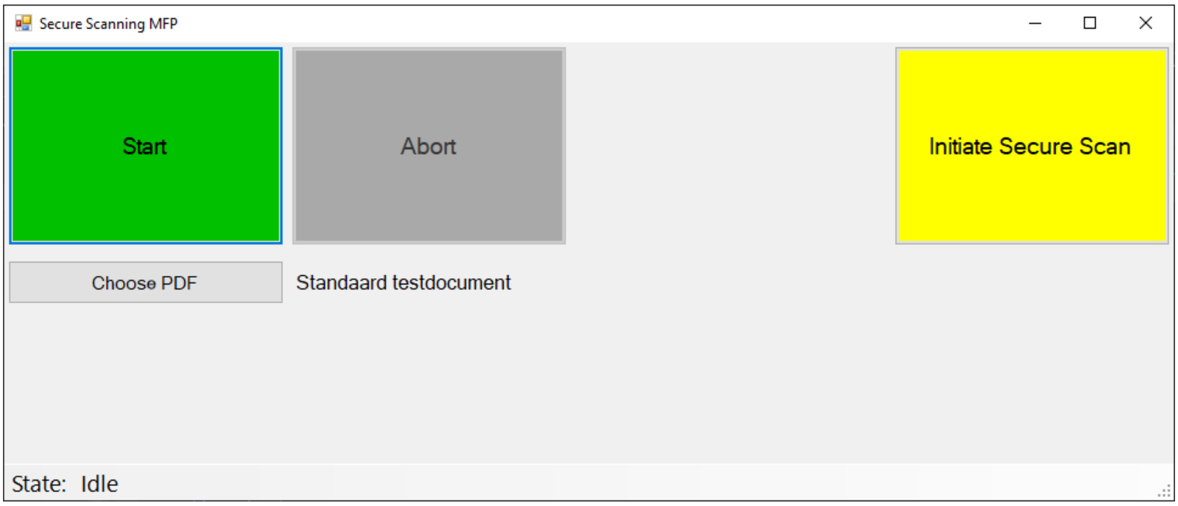
Your personal certificate has successfully been created.

17:07

Secure Scan

jan@softable.nl

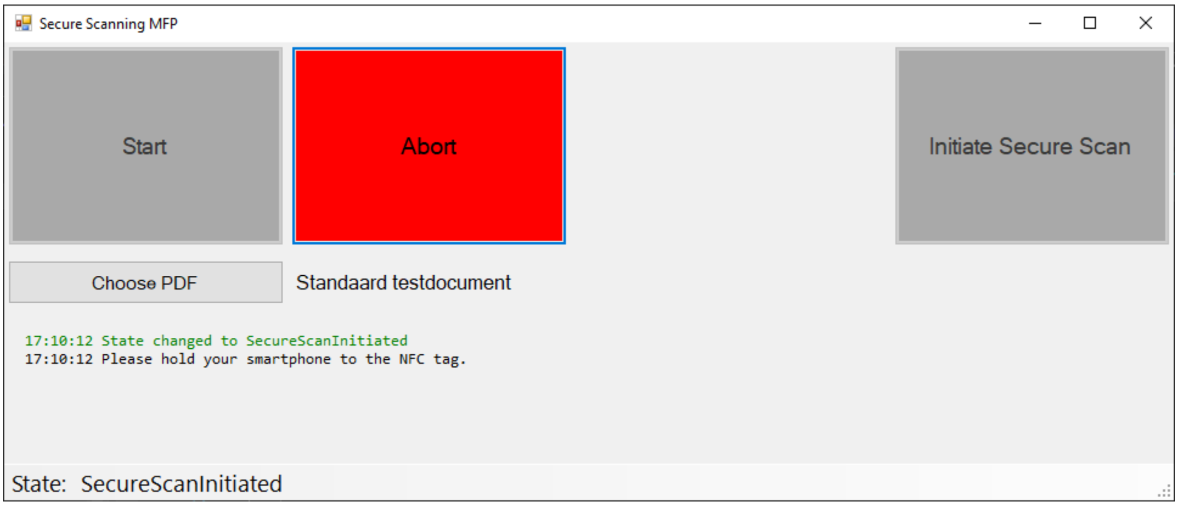
Figure A.2: Certificate created (left) and idle (right)



Standaard testdocument

State: Idle

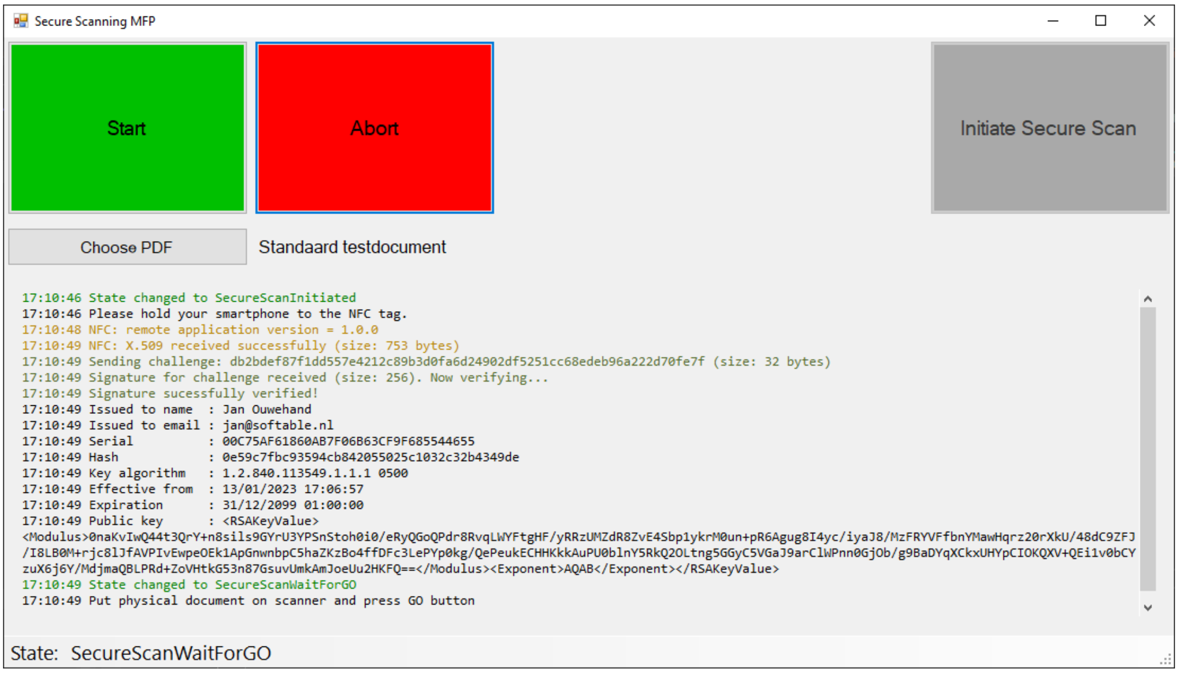
Figure A.3: MFP prototype, idle state



```
17:10:12 State changed to SecureScanInitiated
17:10:12 Please hold your smartphone to the NFC tag.
```

State: SecureScanInitiated

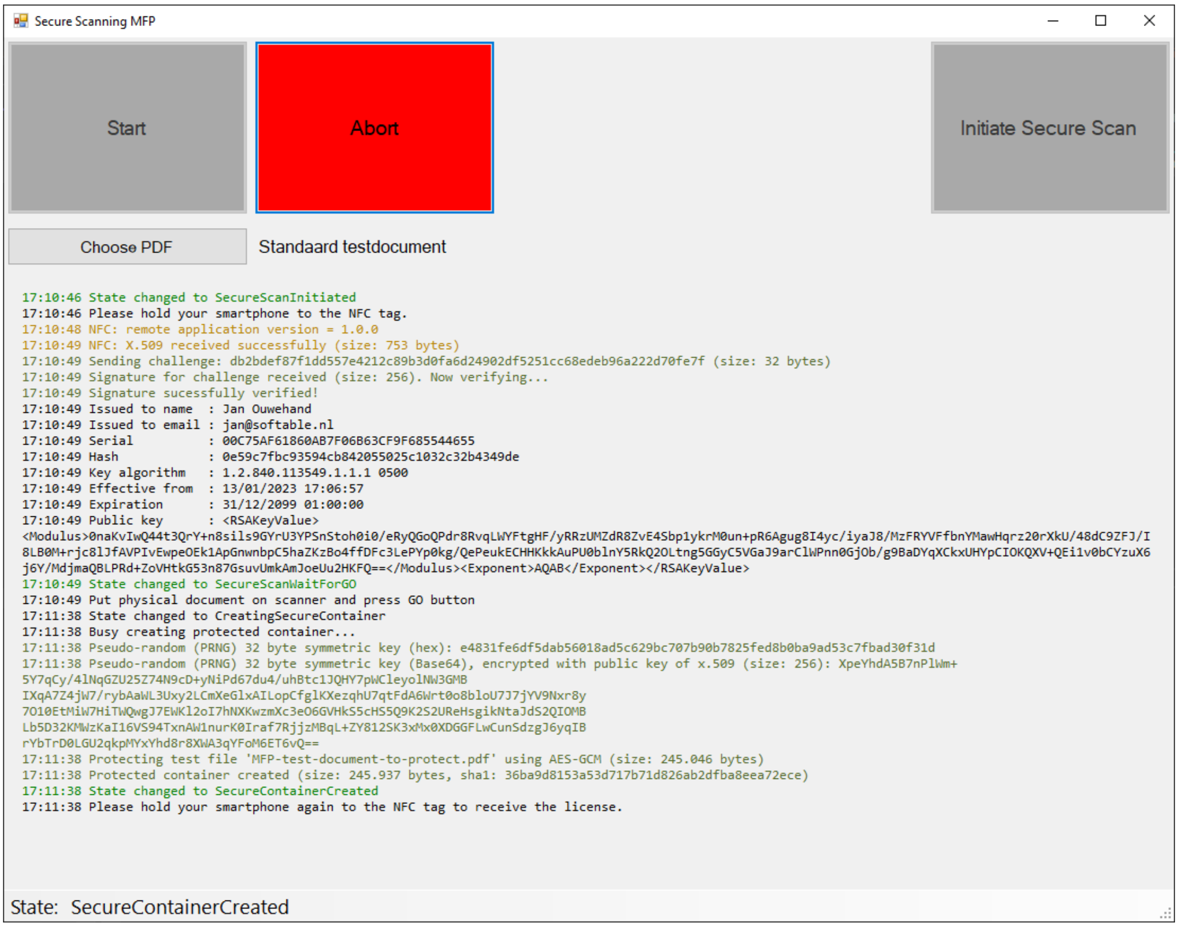
Figure A.4: MFP prototype, secure scan initiated



```
17:10:46 State changed to SecureScanInitiated
17:10:46 Please hold your smartphone to the NFC tag.
17:10:48 NFC: remote application version = 1.0.0
17:10:49 NFC: X.509 received successfully (size: 753 bytes)
17:10:49 Sending challenge: db2bdef87f1dd557e4212c89b3d0fa6d24902df5251cc68edeb96a222d70fe7f (size: 32 bytes)
17:10:49 Signature for challenge received (size: 256). Now verifying...
17:10:49 Signature sucessfully verified!
17:10:49 Issued to name  : Jan Ouwehand
17:10:49 Issued to email : jan@softable.nl
17:10:49 Serial          : 00C75AF61860AB7F06B63CF9F685544655
17:10:49 Hash            : 0e59c7fbc93594cb842055025c1032c32b4349de
17:10:49 Key algorithm   : 1.2.840.113549.1.1.1 0500
17:10:49 Effective from  : 13/01/2023 17:06:57
17:10:49 Expiration      : 31/12/2099 01:00:00
17:10:49 Public key      : <RSAKeyValue>
<Modulus>0naKvIwQ44t3QrY+n8sils9GYrU3YPSnStoh0i0/eRyQGoQPdr8RvqLWYFtgHF/yRRzUMZdR8ZvE4Sbp1ykrM0un+pR6Agug8I4yc/iyaJ8/MzFRYVFfbnYMawHqrz20rXkU/48dC9ZFJ
/I8LB0M+rjc8lJfAVPIvEwpeOEk1ApGnwnbpC5haZKzBo4ffDFc3LePYp0kg/QePeukECHHKkkAuPU0blnY5RkQ2OLtng5GGyC5VGaJ9arClWPnn0GjOb/g9BaDYqXCkxUHYpCIOKQXV+QEi1v0bCY
zuX6j6Y/MdjmaQBLPRd+ZoVHtkG53n87GsuvUmkAmJoeUu2HKFQ==</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>
17:10:49 State changed to SecureScanWaitForGO
17:10:49 Put physical document on scanner and press GO button
```

State: SecureScanWaitForGO

Figure A.5: MFP prototype, pubkey received from smartphone through NFC

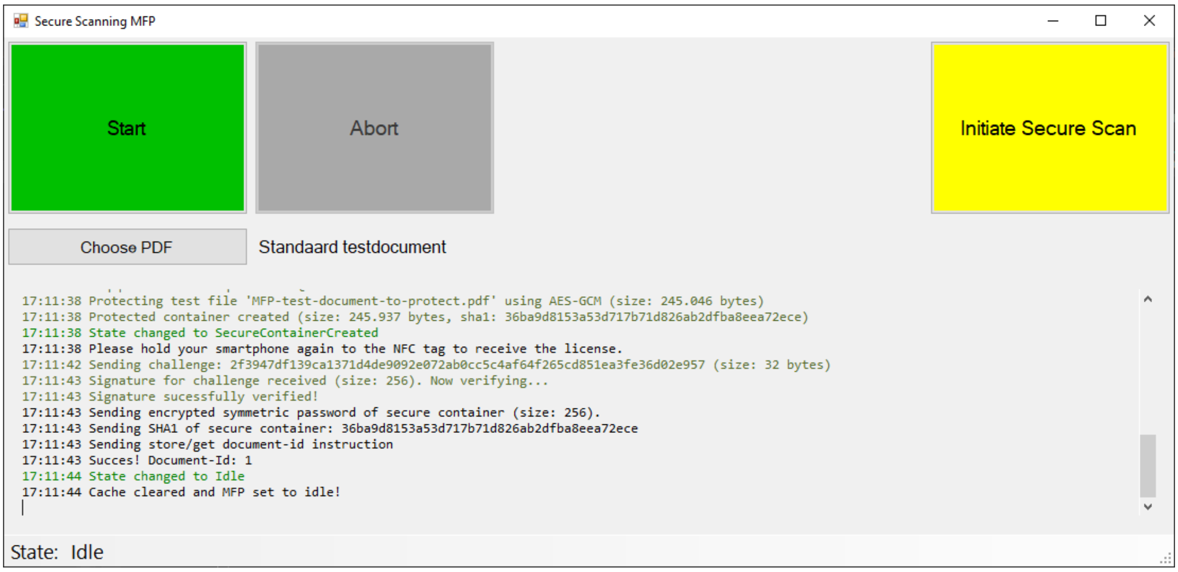


Standaard testdocument

```
17:10:46 State changed to SecureScanInitiated
17:10:46 Please hold your smartphone to the NFC tag.
17:10:48 NFC: remote application version = 1.0.0
17:10:49 NFC: X.509 received successfully (size: 753 bytes)
17:10:49 Sending challenge: db2bdef87f1dd557e4212c89b3d0fa6d24902df5251cc68edeb96a222d70fe7f (size: 32 bytes)
17:10:49 Signature for challenge received (size: 256). Now verifying...
17:10:49 Signature sucessfully verified!
17:10:49 Issued to name  : Jan Ouwehand
17:10:49 Issued to email : jan@softable.nl
17:10:49 Serial          : 00C75AF61860AB7F06B63CF9F685544655
17:10:49 Hash            : 0e59c7fbc93594cb842055025c1032c32b4349de
17:10:49 Key algorithm   : 1.2.840.113549.1.1.1 0500
17:10:49 Effective from  : 13/01/2023 17:06:57
17:10:49 Expiration      : 31/12/2099 01:00:00
17:10:49 Public key      : <RSAKeyValue>
<Modulus>0naKvIwQ44t3QrY+n8sils9GYrU3YPSnStoh0i0/eRyQGoQPdr8RvqLWYFtgHF/yRRzUMZdR8ZvE4Sbp1ykrM0un+pR6Agug8I4yc/iyaJ8/MzFRYVFfbnYMawHqrz20rXkU/48dC9ZFJ/I
8LB0M+rjc8lJfAVPIvEwpeOEk1ApGnwnbpC5haZKzBo4ffDFc3LePYp0kg/QePeukECHHKkkAuPU0blnY5RkQ2OLtng5GGyC5VGaJ9arClWPnn0GjOb/g9BaDYqXCkxUHYpCIOKQXV+QEi1v0bCYzuX6
j6Y/MdjmaQBLPRd+ZoVHtkG53n87GsuvUmkAmJoeUu2HKFQ==</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>
17:10:49 State changed to SecureScanWaitForGO
17:10:49 Put physical document on scanner and press GO button
17:11:38 State changed to CreatingSecureContainer
17:11:38 Busy creating protected container...
17:11:38 Pseudo-random (PRNG) 32 byte symmetric key (hex): e4831fe6df5dab56018ad5c629bc707b90b7825fed8b0ba9ad53c7fbad30f31d
17:11:38 Pseudo-random (PRNG) 32 byte symmetric key (Base64), encrypted with public key of x.509 (size: 256): XpeYhdA5B7nPlWm+
5Y7qCy/4lNqGZU25Z74N9cD+yNiPd67du4/uhBtc1JQHY7pWCleyolNW3GMB
IXqA7Z4jW7/rybAaWL3Uxy2LCmXeGlxAILopCfglKXezqhU7qtFdA6Wrt0o8bloU7J7jYV9Nxr8y
7O10EtMiW7HiTWQwgJ7EWKl2oI7hNXKwzmXc3eO6GVHkS5cHS5Q9K2S2UReHsgikNtaJdS2QIOMB
Lb5D32KMWzKaI16VS94TxnAW1nurK0Iraf7RjjzMBqL+ZY812SK3xMx0XDGGFLwCunSdzgJ6yqIB
rYbTrD0LGU2qkpMYxYhd8r8XWA3qYFoM6ET6vQ==
17:11:38 Protecting test file 'MFP-test-document-to-protect.pdf' using AES-GCM (size: 245.046 bytes)
17:11:38 Protected container created (size: 245.937 bytes, sha1: 36ba9d8153a53d717b71d826ab2dfba8eea72ece)
17:11:38 State changed to SecureContainerCreated
17:11:38 Please hold your smartphone again to the NFC tag to receive the license.
```

State: SecureContainerCreated

Figure A.6: MFP prototype, document scanned

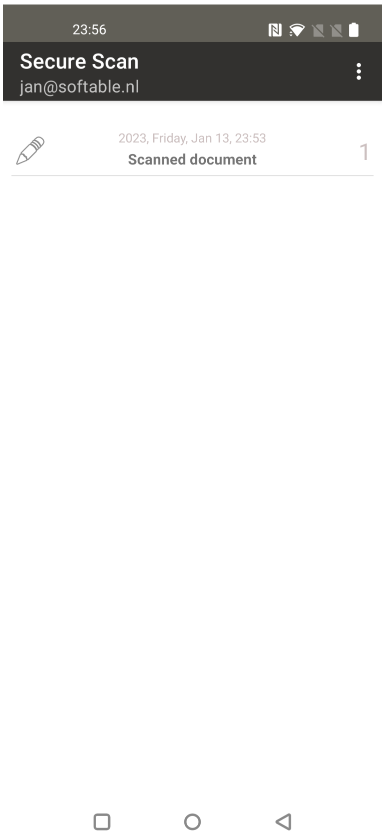


Standaard testdocument

```
17:11:38 Protecting test file 'MFP-test-document-to-protect.pdf' using AES-GCM (size: 245.046 bytes)
17:11:38 Protected container created (size: 245.937 bytes, sha1: 36ba9d8153a53d717b71d826ab2dfba8eea72ece)
17:11:38 State changed to SecureContainerCreated
17:11:38 Please hold your smartphone again to the NFC tag to receive the license.
17:11:42 Sending challenge: 2f3947df139ca1371d4de9092e072ab0cc5c4af64f265cd851ea3fe36d02e957 (size: 32 bytes)
17:11:43 Signature for challenge received (size: 256). Now verifying...
17:11:43 Signature sucessfully verified!
17:11:43 Sending encrypted symmetric password of secure container (size: 256).
17:11:43 Sending SHA1 of secure container: 36ba9d8153a53d717b71d826ab2dfba8eea72ece
17:11:43 Sending store/get document-id instruction
17:11:43 Succes! Document-Id: 1
17:11:44 State changed to Idle
17:11:44 Cache cleared and MFP set to idle!
```

State: Idle

Figure A.7: MFP prototype, document hash and key communicated with smartphone

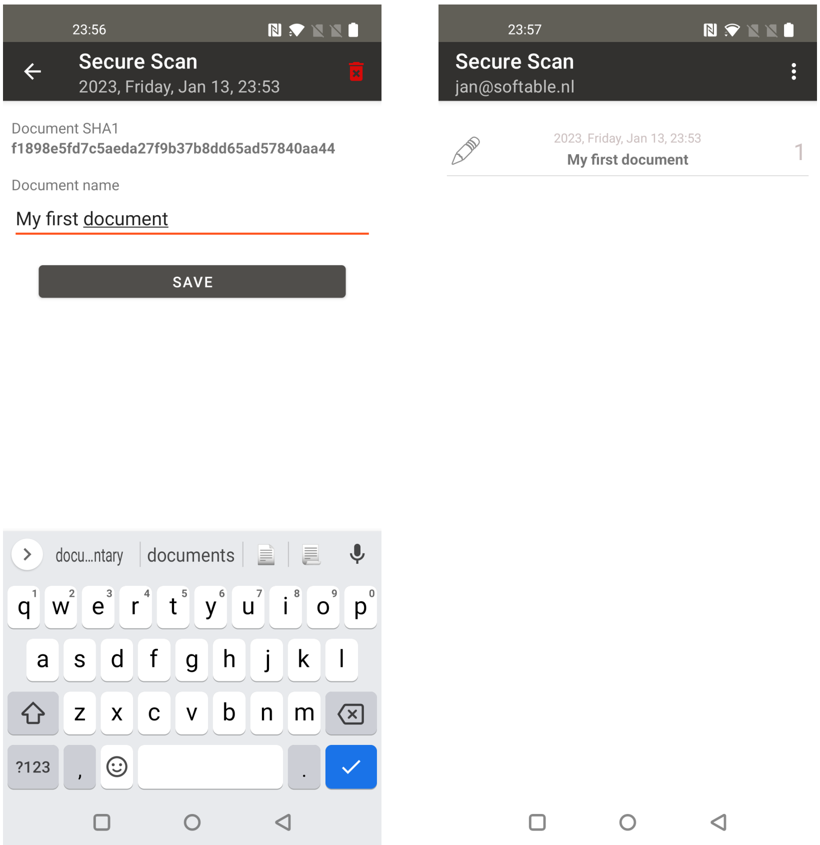


23:56

Secure Scan
jan@softable.nl

2023, Friday, Jan 13, 23:53
Scanned document
1

Figure A.8: Document on smartphone app



Secure Scan
2023, Friday, Jan 13, 23:53

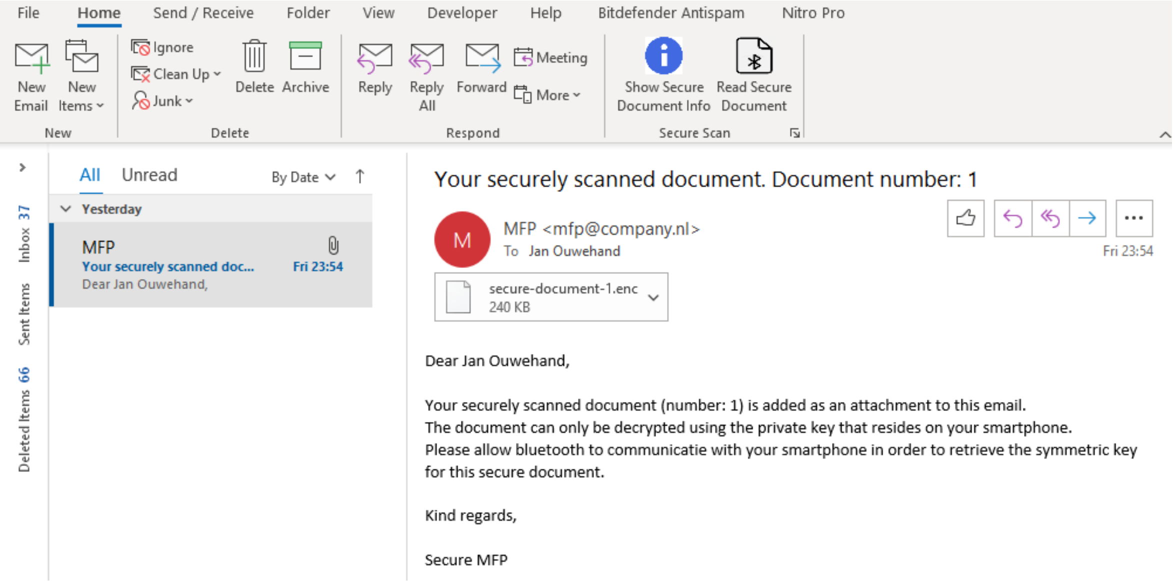
Document SHA1
f1898e5fd7c5aeda27f9b37b8dd65ad57840aa44

Document name
My first document

Secure Scan
jan@softable.nl

2023, Friday, Jan 13, 23:53
My first document 1

Figure A.9: Change name of document



MFP
Your securely scanned doc... Fri 23:54
Dear Jan Ouwehand,

Your securely scanned document. Document number: 1

MFP <mfp@company.nl>
To Jan Ouwehand
Fri 23:54

secure-document-1.enc
240 KB

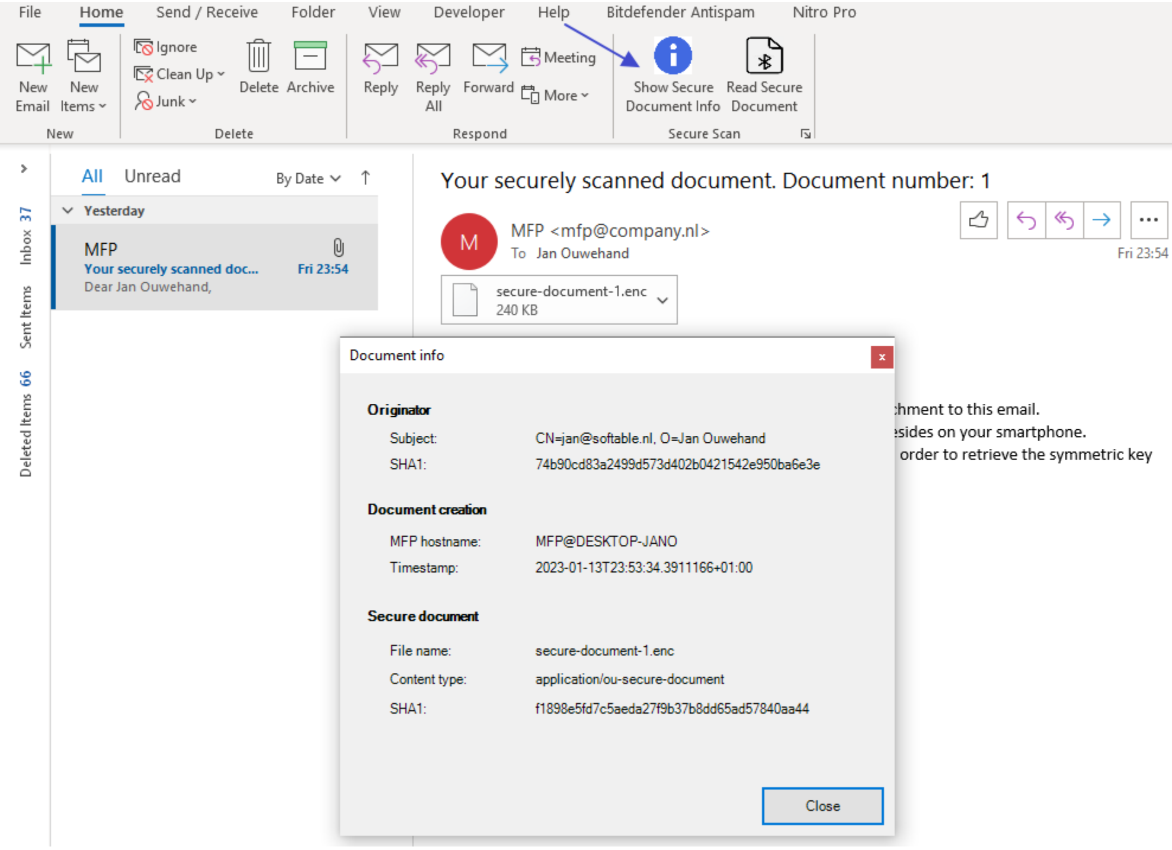
Dear Jan Ouwehand,

Your securely scanned document (number: 1) is added as an attachment to this email.
The document can only be decrypted using the private key that resides on your smartphone.
Please allow bluetooth to communicatie with your smartphone in order to retrieve the symmetric key for this secure document.

Kind regards,

Secure MFP

Figure A.10: Mail-client, received document



MFP
Your securely scanned doc... Fri 23:54
Dear Jan Ouwehand,

Your securely scanned document. Document number: 1

MFP <mfp@company.nl>
To Jan Ouwehand
Fri 23:54

secure-document-1.enc
240 KB

…hment to this email.
…sides on your smartphone.
… order to retrieve the symmetric key

Document info

**Originator**

Subject: CN=jan@softable.nl, O=Jan Ouwehand
SHA1: 74b90cd83a2499d573d402b0421542e950ba6e3e

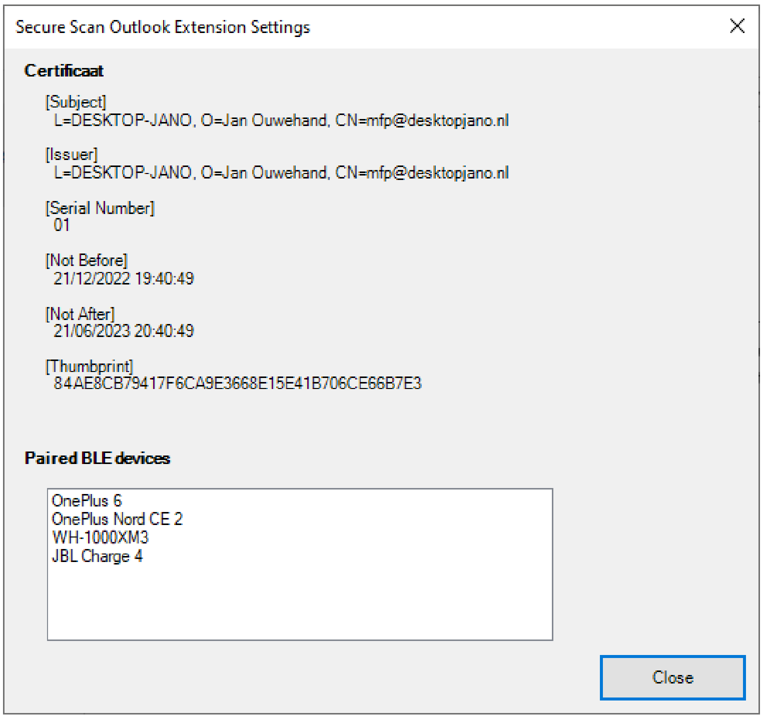
**Document creation**

MFP hostname: MFP@DESKTOP-JANO
Timestamp: 2023-01-13T23:53:34.3911166+01:00

**Secure document**

File name: secure-document-1.enc
Content type: application/ou-secure-document
SHA1: f1898e5fd7c5aeda27f9b37b8dd65ad57840aa44

Figure A.11: Mail-client, show properties of secure document



Secure Scan Outlook Extension Settings

**Certificaat**

[Subject]
L=DESKTOP-JANO, O=Jan Ouwehand, CN=mfp@desktopjano.nl

[Issuer]
L=DESKTOP-JANO, O=Jan Ouwehand, CN=mfp@desktopjano.nl

[Serial Number]
01

[Not Before]
21/12/2022 19:40:49

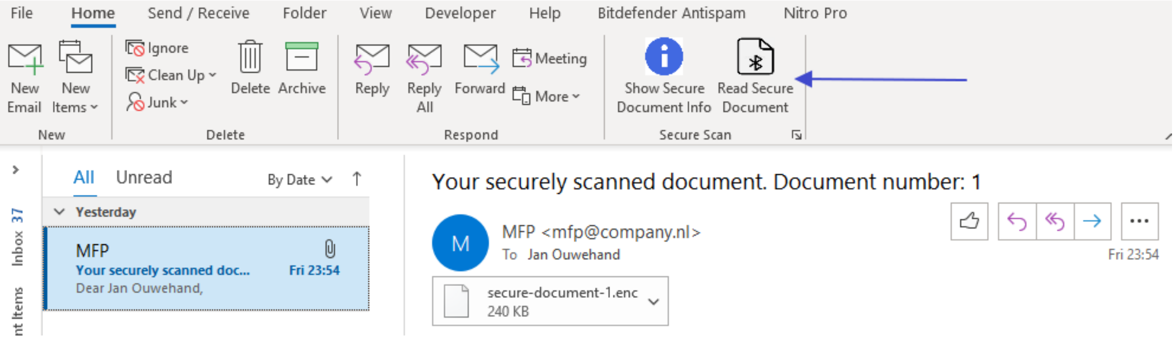
[Not After]
21/06/2023 20:40:49

[Thumbprint]
84AE8CB79417F6CA9E3668E15E41B706CE66B7E3

**Paired BLE devices**

OnePlus 6
OnePlus Nord CE 2
WH-1000XM3
JBL Charge 4

Figure A.12: Mail-client, secure document info



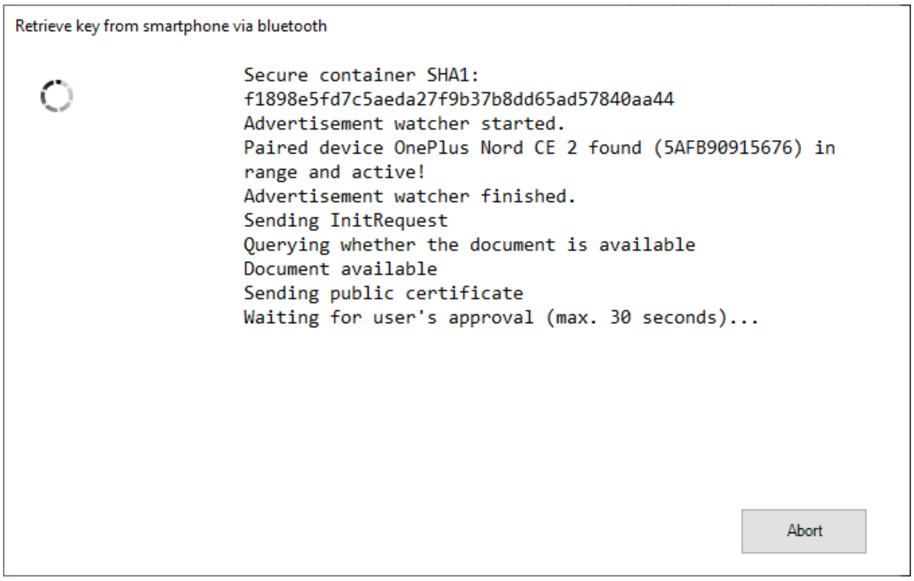
MFP
Your securely scanned doc... Fri 23:54
Dear Jan Ouwehand,

Your securely scanned document. Document number: 1

MFP <mfp@company.nl>
To Jan Ouwehand
Fri 23:54

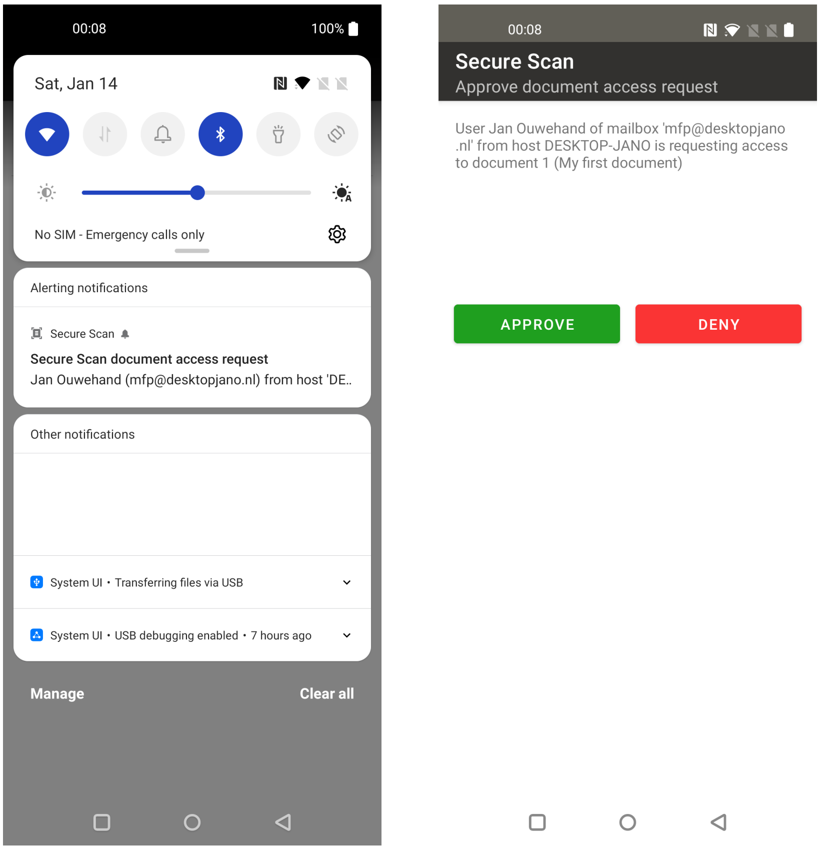
secure-document-1.enc
240 KB

Figure A.13: Mail-client, read document


Retrieve key from smartphone via bluetooth

```
Secure container SHA1:
f1898e5fd7c5aeda27f9b37b8dd65ad57840aa44
Advertisement watcher started.
Paired device OnePlus Nord CE 2 found (5AFB90915676) in
range and active!
Advertisement watcher finished.
Sending InitRequest
Querying whether the document is available
Document available
Sending public certificate
Waiting for user's approval (max. 30 seconds)...
```

Figure A.14: Mail-client, communicating with smartphone



00:08
100%
Sat, Jan 14
No SIM - Emergency calls only
Alerting notifications
Secure Scan

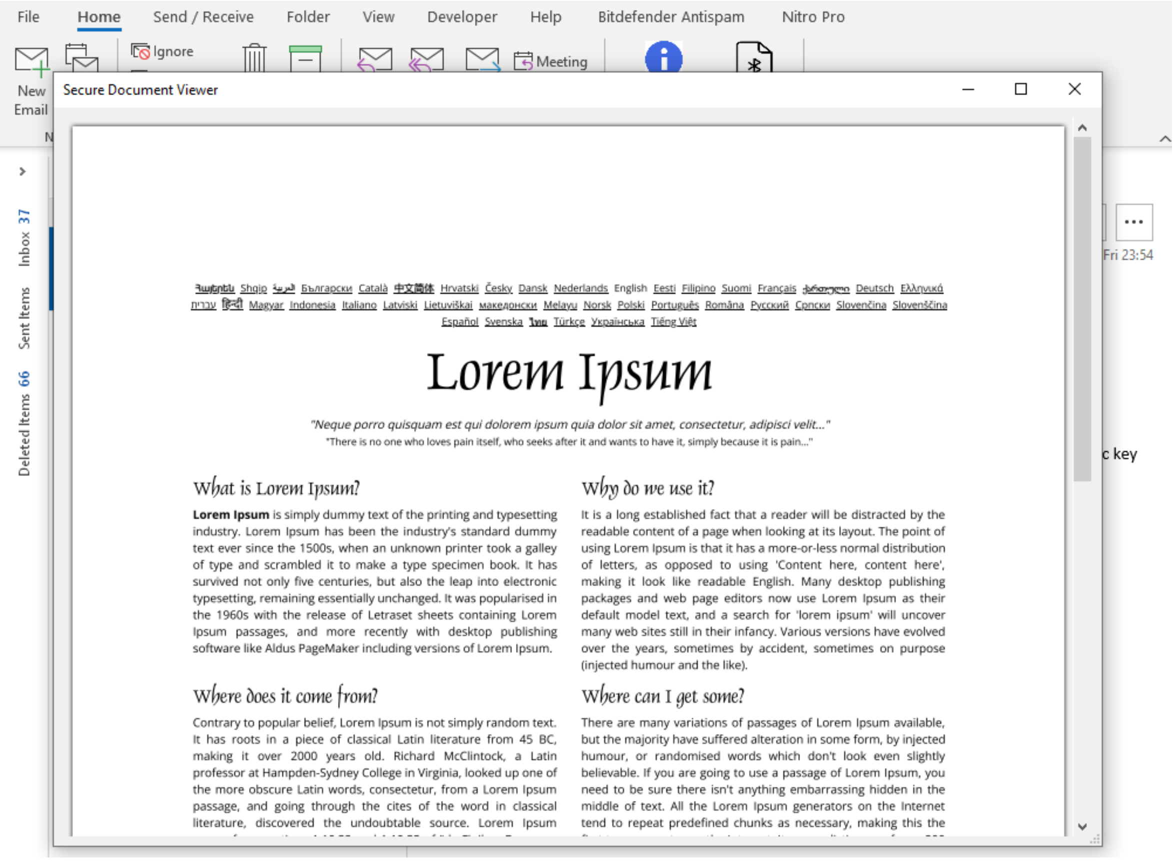
Secure Scan document access request

Jan Ouwehand (mfp@desktopjano.nl) from host 'DE..

Other notifications
System UI • Transferring files via USB
System UI • USB debugging enabled • 7 hours ago
Manage
Clear all
00:08
Secure Scan
Approve document access request

User Jan Ouwehand of mailbox 'mfp@desktopjano.nl' from host DESKTOP-JANO is requesting access to document 1 (My first document)

Figure A.15: Approve access request on smartphone



File
Home
Send / Receive
Folder
View
Developer
Help
Bitdefender Antispam
Nitro Pro
Ignore
Meeting
New Email
Inbox 37
Sent Items
Deleted Items 66
Fri 23:54
c key
Secure Document Viewer
Հայերեն Shqip العربية Български Català 中文簡體 Hrvatski Česky Dansk Nederlands English Eesti Filipino Suomi Français ქართული Deutsch Ελληνικά
עברית हिन्दी Magyar Indonesia Italiano Latviski Lietuviškai македонски Melayu Norsk Polski Português Româna Pусский Српски Slovenčina Slovenščina
Español Svenska ไทย Türkçe Українська Tiếng Việt

# Lorem Ipsum

*"Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit..."*

"There is no one who loves pain itself, who seeks after it and wants to have it, simply because it is pain..."

## What is Lorem Ipsum?

**Lorem Ipsum** is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

## Where does it come from?

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going through the cites of the word in classical literature, discovered the undoubtable source. Lorem Ipsum

## Why do we use it?

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).

## Where can I get some?

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the

Figure A.16: Mail-client, rendering secure document

# BIBLIOGRAPHY

[ACC⁺21] Sajeda Akter, Sriram Chellappan, Tusher Chakraborty, Taslim Arefin Khan, Ashikur Rahman, and A. B. M. Alim Al Islam. Man-in-the-middle attack on contactless payment over nfc communications: Design, implementation, experiments and detection. *IEEE Transactions on Dependable and Secure Computing*, 18(6):3012–3023, 2021. 41

[ACD⁺22] Gorjan Alagic, David Cooper, Quynh Dang, Thinh Dang, John M. Kelsey, Jacob Lichtinger, Yi-Kai Liu, Carl A. Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, and Daniel Apon. Nist ir 8413, status report on the third round of the nist post-quantum cryptography standardization process, 2022-07-05 04:07:00 2022. 11

[ADFS⁺23] Nidal Al-Dmour, Umer Farooq, Irfan Sarwar, Muhammad waseem Iqbal, Muhammad Aqeel, Wasim Khan, and Hussam Al Hamadi. Cyber security threats on multifunctional devices and mitigation techniques. pages 1–6, 03 2023. 12

[ALBK19] Kurt Andersen, Brandon Long, Seth Blank, and Murray Kucherawy. The Authenticated Received Chain (ARC) Protocol. RFC 8617, July 2019. 26, 65

[And23a] Android Developers. Data backup overview. https://developer.android.com/guide/topics/data/backup, 2023. Accessed: 2023-12-10. 47

[And23b] Android Developers. Key/value backup. https://developer.android.com/guide/topics/data/keyvaluebackup, 2023. Accessed: 2023-12-10. 47

[Ant23] Daniele Antonioli. Bluffs: Bluetooth forward and future secrecy attacks and defenses. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, page 636–650, New York, NY, USA, 2023. Association for Computing Machinery. 9

[AvdBH⁺17] Gergely Alpár, Fabian van den Broek, Brinda Hampiholi, Bart Jacobs, Wouter Lueks, and Sietse Ringers. Irma : practical , decentralized and privacy-friendly identity management using smartphones. 2017. 13, 82

[ban23] How many people have smartphones worldwide, Nov 2023. Accessed: Nov 22, 2023. 38

[BBJ⁺23] Leon Botros, Merel Brandon, Bart Jacobs, Daniel Ostkamp, Hanna Schraffenberger, and Marloes Venema. Postguard: Towards easy and secure email communication. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–6, 2023. 13

[BDK⁺18] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018, London, United Kingdom, April 24-26, 2018*, pages 353–367. IEEE, 2018. 11, 40

[BHK⁺19] Daniel J. Bernstein, Andreas Hülsing, Stefan Kölbl, Ruben Niederhagen, Joost Rijneveld, and Peter Schwabe. The sphincs$^+$ signature framework. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pages 2129–2146. ACM, 2019. 11

[BJ20] Dragana Bolcic-Jankovic. *Doctor-Patient Communication by eMail: Trends, Determinants, and Effects of Digital Disparities on eMail Use and the Association between eMail Use and Quality of Communication in Health Care.* PhD thesis, University of Massachusetts Boston, 2020. v

[BLO+15] Zinaida Benenson, Gabriele Lenzini, Daniela Oliveira, Simon Parkin, and Sven Uebelacker. Maybe poor johnny really cannot encrypt: The case for a complexity theory for usable security. In Anil Somayaji, Paul C. van Oorschot, Mohammad Mannan, and Rainer Böhme, editors, *Proceedings of the 2015 New Security Paradigms Workshop, NSPW 2015, Twente, The Netherlands, September 8-11, 2015*, pages 85–99. ACM, 2015. 2, 12, 65

[bri23] 8 staggering statistics: Physical security technology adoption, 2023. Accessed: Nov 22, 2023. 38

[BS18] J. Botha and S. Solms. Security threats and measures on multifunctional devices. 06 2018. 8, 12

[CA/23] CA/Browser Forum. Baseline requirements for the issuance and management of publicly-trusted s/mime certificates. https://cabforum.org/wp-content/uploads/CA-Browser-Forum-SMIMEBR-1.0.1.pdf, August 2023. Copyright 2023 CA/Browser Forum, licensed under the Creative Commons Attribution 4.0 International license. 45

[Can23] Canon Device Security - Cybersecurity. https://csa.canon.com/internet/portal/us/csa/products/softwaresolutions/security/cybersecurity/device-security, 2023. Accessed: 2023-12-08. 12

[CPST22] Matthias Cäsar, Tobias Pawelke, Jan Steffan, and Gabriel Terhorst. A survey on bluetooth low energy security and privacy. *Comput. Networks*, 205:108712, 2022. 9

[DBC+14] Anupam Das, Joseph Bonneau, Matthew C. Caesar, Nikita Borisov, and Xiaofeng Wang. The tangled web of password reuse. In *Network and Distributed System Security Symposium*, 2014. 38

[Dig23] DigiCert. Create a csr (certificate signing request). https://www.digicert.com/kb/csr-creation.htm, 2023. Accessed: 2023-11-22. 39

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(1):238–268, 2018. 11

[DMR10] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. Password strength: An empirical analysis. In *2010 Proceedings IEEE INFOCOM*, pages 1–9, 2010. 38

[EPFB13] Maximilian Engelhardt, Florian Pfeiffer, Klaus Finkenzeller, and Erwin Biebl. Extending iso/iec 14443 type a eavesdropping range using higher harmonics. In *Smart SysTech 2013; European Conference on Smart Objects, Systems and Technologies*, pages 1–8, 2013. 40

[FDC+07] Hal Finney, Lutz Donnerhacke, Jon Callas, Rodney L. Thayer, and Daphne Shaw. OpenPGP Message Format. RFC 4880, November 2007. 40

[FH07] Dinei Florencio and Cormac Herley. A large-scale study of web password habits. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 657–666, New York, NY, USA, 2007. Association for Computing Machinery. 38

[GE21] Craig Gidney and Martin Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum*, 5:433, April 2021. 10

[Goo23a] Google. How google retains data we collect, privacy and terms, google. https://policies.google.com/technologies/retention, 2023. [Accessed 13-11-2023]. 10

[Goo23b] Google. Privacy & data - google safety center. https://safety.google/privacy/data/, 2023. Accessed: 2023-12-10. 47

[Goo23c] Google Cloud. Customer-supplied encryption keys. https://cloud.google.com/docs/security/encryption/customer-supplied-encryption-keys, 2023. Accessed: 2023-12-10. 47

[Goo23d] Google Cloud. Key management deep dive, 2023. Accessed: 2023-12-10. 47

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219. ACM, 1996. 11

[HGB19] Margareta Heidt, Jin P. Gerlach, and Peter Buxmann. Investigating the security divide between SME and large companies: How SME characteristics influence organizational IT security investments. *Inf. Syst. Frontiers*, 21(6):1285–1305, 2019. 8

[Hof02] Paul E. Hoffman. SMTP Service Extension for Secure SMTP over Transport Layer Security. RFC 3207, February 2002. 28

[Hou07] Russ Housley. Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS). RFC 5084, November 2007. 68, 73

[Hou09] Russ Housley. Cryptographic Message Syntax (CMS). RFC 5652, September 2009. 68

[HPA23] HP Access Control (AC) Scan (PRO). https://h20195.www2.hp.com/v2/GetDocument.aspx?docname=4AA7-8044EeW, 2023. Accessed: 2023-12-08. 12

[IBM22] IBM. Ibm unveils 400 qubit-plus quantum processor and next-generation ibm quantum system two. https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two, 2022. [Accessed 13-11-2023]. 10

[IBM23] IBM. Ibm, charting the course to 100,000 qubits, ibm quantum roadmap. https://research.ibm.com/blog/100k-qubit-supercomputer, 2023. [Accessed 13-11-2023]. 10

[Jos06] Simon Josefsson. Storing Certificates in the Domain Name System (DNS). RFC 4398, March 2006. 66

[JvdBdR16] Erik Poll Jordi van den Breekel, Diego A. Ortiz-Yepes and Joeri de Ruiter. EMV in a nutshell. Technical report, KPMG; IBM Research Zurich; Radboud University Nijmegen, 6 2016. https://www.cs.ru.nl/~erikpoll/publications/EMVtechreport.pdf. 40

[KCH11] Murray Kucherawy, Dave Crocker, and Tony Hansen. DomainKeys Identified Mail (DKIM) Signatures. RFC 6376, September 2011. 25, 65

[Kle08] Dr. John C. Klensin. Simple Mail Transfer Protocol. RFC 5321, October 2008. 28

[KMLS17] Rafiullah Khan, Kieran McLaughlin, David Laverty, and Sakir Sezer. Stride-based threat modeling for cyber-physical systems. In *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–6, 2017. 14

[Kob18] Nadim Kobeissi. An analysis of the protonmail cryptographic architecture. *IACR Cryptol. ePrint Arch.*, page 1121, 2018. 13

[Kyr23] D. Kyrlynn. Quantumzeitgeist, us government increases funding for quantum computing research for 2023. https://quantumzeitgeist.com/us-government-increases-funding-for-quantum-computing-research-for-2023/, 2023. [Accessed 13-11-2023]. 10

[L'E12] Pierre L'Ecuyer. *Random number generation*. Springer, 2012. 38

[Lin99] Gunnar Lindberg. Anti-Spam Recommendations for SMTP MTAs. RFC 2505, February 1999. 24

[MD21] Nicky Mouha and Morris Dworkin. Nist ir 8319, review of the advanced encryption standard, 2021-07-23 04:07:00 2021. 11

[Moz] Mozilla. https://github.com/mozilla/pdf.js. [Accessed 13-11-2023]. 56

[MP20] Shiri Melumad and Michel Tuan Pham. The smartphone as a pacifying technology. *Journal of Consumer Research*, 47(2):237–255, 2020. 38

[MRR$^+$18] Daniel Margolis, Mark Risher, Binu Ramakrishnan, Alex Brotman, and Janet Jones. SMTP MTA Strict Transport Security (MTA-STS). RFC 8461, September 2018. 28

[MS19] Mark Mamchenko and Alexey Sabanov. Exploring the taxonomy of usb-based attacks. In *2019 Twelfth International Conference "Management of large-scale system development" (MLSD)*, pages 1–4, 2019. 38

[MSR$^+$17] Ghulam Mujtaba, Liyana Shuib, Ram Gopal Raj, Nahdia Majeed, and Mohammed Ali Al-Garadi. Email classification research trends: Review and open issues. *IEEE Access*, 5:9044–9064, 2017. v

[Nat20] National Institute of Standards and Technology (NIST). NIST Special Publication 800-57 Part 1 Revision 5. NIST Special Publication 800-57 Part 1 Rev. 5, National Institute of Standards and Technology, 2020. 45, 46

[Not19] Mark Nottingham. Well-Known Uniform Resource Identifiers (URIs). RFC 8615, May 2019. 66

[NS23] Mathew Nicho. and Ibrahim Sabry. Bypassing multiple security layers using malicious usb human interface device. In *Proceedings of the 9th International Conference on Information Systems Security and Privacy - ICISSP*, pages 501–508. INSTICC, SciTePress, 2023. 54

[NYE17a] Nir Nissim, Ran Yahalom, and Yuval Elovici. Usb-based attacks. *Computers & Security*, 70:675–688, 2017. 38

[NYE17b] Nir Nissim, Ran Yahalom, and Yuval Elovici. Usb-based attacks. *Comput. Secur.*, 70:675–688, 2017. 54

[OAAA23] Israel Oludayo Ogundele, Agnes Kikelomo Akinwole, Adeniran Adedeji Adebayo, and Adewale Ayodeji Aromolaran. A review of smartphone security challenges and prevention. *International Research Journal of Innovations in Engineering and Technology (IRJIET)*, 7(5):234–245, May 2023. 9

[OWA23] OWASP. Threat modeling process. https://owasp.org/www-community/Threat_Modeling_Process, 2023. Accessed: 2023-12-10. 14

[PDF] PDFium. https://pdfium.googlesource.com/pdfium. [Accessed 13-11-2023]. 56

[PIBS21] Damian Poddebniak, Fabian Ising, Hanno Böck, and Sebastian Schinzel. Why TLS is better without STARTTLS: A security analysis of STARTTLS in the email context. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 4365–4382. USENIX Association, August 2021. 4

[PR19] Jithu Philip and Merin Raju. A formal overview of application sandbox in android and ios with the need to secure. 2019. http://gnanaganga.inflibnet.ac.in:8080/jspui/handle/123456789/1066. 46

[Quo23] Quocirca. The print security landscape 2023. Excerpt report: Xerox, May 2023. Print security trends in the US and Europe. Available at https://www.xerox.com/downloads/usa/en/services/report/quocirca-print-security-2023-xerox-excerpt.pdf. 9

[RAZS15] 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. *CoRR*, abs/1510.08555, 2015. 12, 65

[RFW15] Andrew Regenscheid, Larry Feldman, and Gregory Witte. Nist special publication 800-88, revision 1: Guidelines for media sanitization, 2015-02-05 2015. 42, 49, 50

[SBN+21] Deepraj Soni, Kanad Basu, Mohammed Nabeel, Najwa Aaraj, Marcos Manzano, and Ramesh Karri. *FALCON*, pages 31–41. Springer International Publishing, Cham, 2021. 11

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997. 10

[Sma14] Smart Card Alliance - Mobile and NFC Council. Host Card Emulation (HCE) 101. White Paper MNFCC-14002, Smart Card Alliance - Mobile and NFC Council, August 2014. https://www.iqdevices.us/pdfFiles/HCE-101-WP-FINAL-081114-clean.pdf. 40

[SMS18] Da-Zhi Sun, Yi Mu, and Willy Susilo. Man-in-the-middle attacks on secure simple pairing in bluetooth standard v5.0 and its countermeasure. *Personal and ubiquitous computing*, 22(1):55–67, 2018. 9, 53

[SRT19] Jim Schaad, Blake C. Ramsdell, and Sean Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification. RFC 8551, April 2019. 73

[sta22a] Global smartphone penetration per capita since 2005, 2022. Accessed: Nov 22, 2023. 38

[Sta22b] Niels Starren. Johnny can encrypt? a usability study of irmaseal. 2022. 13

[TFH21] Dennis Tatang, Robin Flume, and Thorsten Holz. Extended abstract: A first large-scale analysis on usage of mta-sts. In Leyla Bilge, Lorenzo Cavallaro, Giancarlo Pellegrino, and Nuno Neves, editors, *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 361–370, Cham, 2021. Springer International Publishing. 4

[TK18] Süleyman Taşkin and Ecir Ugur Kucuksille. Recovering data using mft records in ntfs file system. *Academic Perspective Procedia*, 1:448–457, 11 2018. 42

[TR10] Sean Turner and Blake C. Ramsdell. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751, January 2010. 68

[UoC23] UoC. University of chicago joins global partnerships to advance quantum computing. https://news.uchicago.edu/story/university-chicago-joins-global-partnerships-advance-quantum-computing, 2023. [Accessed 13-11-2023]. 10

[WSG$^+$22] Chuhan Wang, Kaiwen Shen, Minglei Guo, Yuxuan Zhao, Mingming Zhang, Jianjun Chen, Baojun Liu, Xiaofeng Zheng, Haixin Duan, Yanzhong Lin, et al. A large-scale and longitudinal measurement study of {DKIM} deployment. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1185–1201, 2022. 25

[WT99] Alma Whitten and J. Doug Tygar. Why johnny can't encrypt: A usability evaluation of PGP 5.0. In G. Winfield Treese, editor, *Proceedings of the 8th USENIX Security Symposium, Washington, DC, USA, August 23-26, 1999*. USENIX Association, 1999. 12

[Xer23] Document Scanning and Storage - Enterprise Content Management. https://www.xerox.com/en-ie/services/enterprise-content-management/document-scanning-and-storage, 2023. Accessed: 2023-12-08. 12

# Glossary

**2FA** Two-Factor Authentication. A method of authentication that requires two different factors, such as a password and a one-time password. 39

**AEAD** Authenticated Encryption with Associated Data. A type of encryption that provides confidentiality, integrity, and authenticity. The difference between integrity and authenticity in this context is that integrity only protects the integrity of the encrypted data, whereas authenticity also protects the integrity of the associated data. The associated data, in turn, is data that is not encrypted, but that is authenticated. 68, 73

**AES** Advanced Encryption Standard. A popular symmetric encryption algorithm used to encrypt data. 8, 55

**AES-CBC** AES in Cipher Block Chaining mode. A mode of operation for the AES encryption algorithm that provides confidentiality. CBC mode does not provide authenticity and integrity, and is therefore not an AEAD mode of operation. 68

**AES-GCM** AES in Galois/Counter Mode. A mode of operation for the AES encryption algorithm that provides authenticated encryption. 68, 73

**Android** An operating system for mobile devices developed by Google. 9

**API** Application Programming Interface. A set of functions and procedures that allow the creation of applications that access the features or data of an operating system, application, or other service. 39

**ARC** Authenticated Received Chain. A method for preserving email authentication results across intermediary hops in the email delivery path. 26, 30, 65

**BLC** Bluetooth Classic, often just referred to as Bluetooth, is the original Bluetooth technology, designed for continuous wireless connection and data streaming between devices, such as in phone calls and music streaming. vi, 9, 54

**BLE** Bluetooth Low Energy (BLE) is a power-conserving variant of Bluetooth wireless technology, designed for short-range communication and ideal for applications requiring periodic or intermittent data transfer with minimal energy consumption. 9, 54

**BLE advertising** A method used by BLE devices to broadcast information to other devices in the vicinity. 55

**BLE scanning** A method used by BLE devices to discover other devices in the vicinity. 55

**Bluetooth** A wireless technology that allows devices to communicate with each other over short distances. 9, 40

**CA** Certificate Authority. An entity that issues digital certificates. Often for a fee, as a CA will verify the identity of the certificate holder before issuing a certificate. 39, 60, 66

**certificate** A digital document that contains and binds a public key to an entity, such as a person, e.g. email address, or an organization. A certificate may refer to an X.509 certificate used for web sites or S/MIME, refer PKIX, or to an OpenPGP certificate, refer to RFC 4880. 8, 35, 37

**CMS** Cryptographic Message Syntax (CMS), using ASN.1 notation, is a standard designed for digitally signing and encrypting messages and certificates, emphasizing interoperability across different systems and protocols for secure data transmission and authentication. 68

**confidentiality** In the context of this thesis confidentiality pertains to safeguarding the confidentiality of scanned documents, ensuring that access is strictly controlled and limited to authorized entities. 34, 35

**cryptoanalysis** The study and practice of deciphering encrypted communication. This is associated with cryptanalytic attacks, which exploit weaknesses in cryptographic algorithms or implementations. 10

**cryptographic primitive** A low-level cryptographic function, such as a hash function or a block cipher. 45, 68

**CRYSTALS-Dilithium / ML-DSA** CRYSTALS-Dilithium, also known as ML-DSA, is a digital signature algorithm in the field of post-quantum cryptography, currently undergoing standardization by NIST, known for its resistance to quantum computer attacks, and based on asymmetric cryptographic principles. 11

**CRYSTALS-KYBER / ML-KEM** CRYSTALS-KYBER, also known as ML-KEM, is a public key encryption (PKE) algorithm in the field of post-quantum cryptography, currently in the process of standardization by NIST, recognized for its quantum-resistant attributes and based on asymmetric cryptographic principles. 11, 40

**CSR** Certificate Signing Request. A message sent from an applicant to a certificate authority to apply for a digital identity certificate. Creating a CSR involves generating a key pair, which is used to sign the CSR. 39

**CS-RNG** Cryptographically Secure Random Number Generator. A random number generator that is suitable for use in cryptography. 36, 38, 48, 60, 72

**design methodology** A collection of methods discussed and evaluated for their potential to solve a specific problem. This approach involves examining, comparing, and understanding different techniques or strategies to identify the most feasible solution. 5

**DH** Diffie-Hellman key exchange. A popular method for two parties to agree on a shared secret over an insecure channel. 10

**DHE** Diffie-Hellman Ephemeral. A method for two parties to agree on a shared secret over an insecure channel. The ephemeral variant of Diffie-Hellman uses a new key pair for each key exchange. 41

**discrete logarithmic problem** A problem that involves finding the exponent $x$ in the equation $g^x = y$, where $g$ and $y$ are known. 10

**DKIM** DomainKeys Identified Mail. A method for associating a domain name with an email message, allowing a person, role, or organization to claim some responsibility for the message. DKIM also provides a mechanism for verifying that the message has not been altered in transit. 7, 25, 30

**DMS** Document Management System. A system used to store, manage, and track electronic documents and images of paper documents. 19

**DNS** Domain Name System. A hierarchical and decentralized naming system for computers, services, or other resources connected to the Internet or a private network. The DNS can be used to store certificates using CERT records. 66

**DNSSEC** Domain Name System Security Extensions. A set of extensions to the DNS that provide origin authentication of DNS data, data integrity, and authenticated denial of existence. 66

**DREAD risk assessment model** A risk assessment model that is used to assess the risk of a threat. The DREAD risk assessment model is based on five factors: Damage potential, Reproducibility, Exploitability, Affected users, and Discoverability. DREAD can be used in combination with the STRIDE threat model to assess the risk of a threat. 14

**DSA** Digital Signature Algorithm. An obsolete digital signature algorithm that is based on the discrete logarithmic problem. 10

**ECC** Elliptic Curve Cryptography. A popular asymmetric or public-key encryption algorithm that is based on the discrete logarithmic problem. 10, 40

**economy of mechanism** A principle that states that the design of a system should be as simple as possible to reduce the number of potential vulnerabilities. 9

**EV certificate** Extended Validation certificate. A type of certificate that provides the highest level of authentication. EV certificates are often used by banks and other financial institutions. 66, 67, 69, 72

**FALCON** A post-quantum digital signature algorithm. 11

**FIDO** Fast Identity Online. An open standard for passwordless authentication. 38

**firmware** Software that is embedded in a hardware device, providing the necessary instructions for how the device communicates with other computer hardware. 8

**FTP** File Transfer Protocol. A protocol used to transfer files between computers. 8

**full disk encryption** A type of encryption that encrypts all data on a disk, including the operating system. Prominent examples of full disk encryption software are BitLocker on Windows and FileVault on macOS. 42

**GCM** Galois/Counter Mode. A mode of operation for block ciphers that provides authenticated encryption. 11, 55

**GDPR** General Data Protection Regulation. A regulation in EU law on data protection and privacy in the European Union and the European Economic Area. 3, 63

**GMail** An email service provided by Google. 10

**Grover's algorithm** A quantum algorithm that can be used to find the solution to a search problem in $O(\sqrt{N})$ time, where $N$ is the number of possible solutions. Grover's algorithm can be used to break symmetric encryption algorithms and hash functions. 11

**HCE** Host Card Emulation. A technology that allows mobile devices to emulate smart cards and perform contactless transactions using NFC. 40, 48

**HMAC** Hash-based Message Authentication Code. A type of message authentication code (MAC) calculated using a cryptographic hash function in combination with a secret key. 60, 61

**HTTP** Hypertext Transfer Protocol. A protocol used by web browsers to request web pages from web servers. 8

**HTTPS** Hypertext Transfer Protocol Secure. An extension of the HTTP protocol that uses TLS to provide end-to-end security for applications that communicate over a network. Currently, TLS 1.3 is the latest version of TLS. 66

**IANA** Internet Assigned Numbers Authority. A department of ICANN, a nonprofit organization that is responsible for coordinating the maintenance and procedures of several databases related to the namespaces and numerical spaces of the internet, ensuring the network's stable and secure operation. 67

**IBE** Identity-Based Encryption. A type of public-key encryption in which the public key of a user is some unique information about the user, such as an email address or a phone number. 13

**institution** An institution, in the context of this thesis, refers to an official organization that provides a service to the public, such as a bank, insurance company or government agency. An institution is the recipient of the a scanned document. 34, 35

**integer factorization problem** A problem that involves finding the prime factors of an integer $n$. 10

**integrity** In the context of this thesis integrity pertains to ensuring that the content of scanned documents is not altered or tampered with. 34, 35

**iOS** An operating system for mobile devices developed by Apple. 9

**IRMA** I Reveal My Attributes. A user-friendly app that allows users to manage their identities. The name has been rebranded to Yivi. 13, 82

**ISO 27000-series** A series of standards that provide best practices and recommendations for information security management. 3, 63

**JSON** JavaScript Object Notation. A lightweight data-interchange format. 67

**KEM** Key Encapsulation Mechanism. A cryptographic primitive that allows two parties to agree on a shared secret over an insecure channel. 46

**MAC** Message Authentication Code. A short piece of information that is used to authenticate a message. 11, 56

**MDA** Mail Delivery Agent. A program that accepts email messages from MTAs and delivers them to MUAs. 8, 21, 79

**MFP** Multi-Function Printer. A device that can print, copy, scan, and fax. v, 2, 8, 34–36

**MITM** Man-In-The-Middle. An attack where an attacker intercepts and modifies communications between two parties. 9, 76

**MSA** Mail Submission Agent (MSA): A server component that receives outgoing emails from an email client (MUA) and relays them to a Mail Transfer Agent (MTA) for delivery. 7, 20

**MTA** Mail Transfer Agent. A program that accepts email messages from MSAs or other MTAs and routes them towards their final destination, which can be another MTA or an MDA. 2, 7, 18, 20, 21, 49, 65

**MTA-STS** Mail Transfer Agent Strict Transport Security. A mechanism that allows a mail server to declare that it should only be contacted using secure SMTP connections. 28, 29

**MUA** Mail User Agent. A program used to read and send email messages, which is used by end users. Also known as an email client. Examples include Microsoft Outlook, Mozilla Thunderbird, and Apple Mail. Note that the MFP that is used to send email messages is also an MUA. v, 7, 8, 18, 34, 35, 52, 77, 79

**NFC** Near Field Communication. A technology that allows two devices to communicate with each other when they are in close proximity. 40

**NIST** National Institute of Standards and Technology, a U.S. federal agency that develops and promotes measurement, standards, and technology. NIST can be considered a frontrunner in the field of encryption and cybersecurity standards. NIST's guidelines and frameworks are also globally recognized and used as benchmarks for best practices in cybersecurity. Especially NIST's Special Publication 800 series is widely used in the field of cybersecurity. 11

**OOB authentication** Out-of-Band authentication. A method of authentication that uses a separate communication channel to verify identity. 42, 55

**OpenPGP** RFC 4880. OpenPGP is a non-proprietary protocol for encrypting email communication and ensuring its authenticity through digital signatures, utilizing public key encryption (PKE) for secure data exchange. Note that PGP is not the same as OpenPGP as, unlike OpenPGP, PGP is not a standard and refers to the original encryption program developed by Phil Zimmermann in 1991. 2, 13, 37, 39, 52

**OpenPGP key-id** A short identifier for an OpenPGP key. The key-id is the last 64 bits of the fingerprint of the key. It is used to identify a key in a keyring, for example when performing key lookups. 45

**OpenPGP Keyring** An OpenPGP Keyring is a collection of cryptographic keys used in OpenPGP for encryption, decryption, and authentication. It typically includes both public keys, which are shared and used for encrypting messages and verifying signatures, and private keys, which are confidential and used for decrypting messages and creating signatures. This keyring, essential for secure communication, is managed by OpenPGP software, allowing users to easily handle and access their keys. 69

**OS** Operating System. A program that manages the hardware and software resources of a computer. 9

**Osprey** A quantum computer developed by D-Wave Systems. The Osprey quantum computer is a state-of-the-art computing system designed to perform complex calculations at unprecedented speeds using the principles of quantum mechanics. 10

**Outlook.com** An email service provided by Microsoft. 10

**post-quantum algorithm** A cryptographic algorithm that is resistant to attacks by quantum computers. 40

**privacy** In the context of this thesis privacy pertains to safeguarding the confidentiality of scanned documents, ensuring that access is strictly controlled and limited to authorized entities. 3, 4

**publish and subscribe** A messaging pattern where senders and receivers of messages are loosely coupled and do not need to know each other's identity. Senders publish messages and receivers subscribe to a topic to receive messages. 55

**QR code** A type of matrix barcode that can be scanned using a smartphone camera. QR codes can be usewd to store arbitrary data, such as cryptographic keys. 36

**quantum computing** A type of computing that uses quantum-mechanical phenomena to perform operations on data. 10

**RAM** Random Access Memory. A type of computer memory that can be accessed randomly, meaning that any byte of memory can be accessed without touching the preceding bytes. A charactieristic of RAM is that it is volatile, meaning that its contents are lost when it loses power. 42

**RSA** Rivest-Shamir-Adleman. A popular asymmetric or public-key encryption algorithm that is based on the integer factorization problem. 10, 40

**RSSI** Received Signal Strength Indicator. A measurement of the power present in a received radio signal. It can be used to estimate the distance between two devices. 55

**S/MIME** Secure/Multipurpose Internet Mail Extensions. A standard that extends the format of email messages to support encryption and digital signatures. vi, 2, 8, 35, 39

**seamless end-to-end encryption** A type of end-to-end encryption that is transparent to the user, which means that the user does not have to take a specific action to encrypt the data. For example, when a user sends an email message, the message is automatically encrypted before it is sent. 3, 34, 35

**secret key** A cryptographic key that is kept secret and used for encryption and decryption. A secret key may refer either to a symmetric key or a private key. 39

**secure digitization** Secure digitization is the process of converting physical documents into digital documents in a secure manner, ensuring that confidentiality and integrity are preserved. 15, 35–37

**Shor's algorithm** A quantum algorithm that can be used to find the prime factors of an integer $N$ in $O((\log N)^3)$ time. Shor's algorithm can be used to break public-key encryption algorithms. 10

**smartphone bound public key** A public key that is bound to a smartphone, meaning that the private key which belongs to the public key cannot be used outside of the smartphone's secure hardware. 48

**SMB** The Server Message Block (SMB) protocol is a widely used network file sharing protocol allowing applications on a computer to read and write to files and request services from server programs in a computer network.. 8

**SMTP** Simple Mail Transfer Protocol. A protocol used to send email messages. 2, 8, 28

**SPF** Sender Policy Framework. An email authentication method that allows the owner of a domain to specify which mail servers are authorized to send email messages from that domain. 30

**SPHINCS+** A post-quantum digital signature algorithm. 11

**SSP** Simple Secure Pairing. A method used by Bluetooth devices to establish a secure connection with each other. 9

**STARTTLS** STARTTLS is an extension to plain text communication protocols, allowing a plain text connection to be upgraded to an encrypted connection. Note that this differs from employing SMTP over TLS, which first establishes an encrypted connection and then starts the SMTP protocol. 2, 28, 30, 65, 73

**stream-based encryption** Stream-based encryption involves encrypting data as it flows, typically in a file stream, where data is encrypted block by block. This method is efficient for large data volumes, as it minimizes RAM usage by not requiring the entire dataset to be loaded into memory. The key advantage is that data written to disk is always in an encrypted state, ensuring no plaintext data is stored on the disk. v, 43

**STRIDE threat model** A threat model that categorizes threats into six categories: Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege. The STRIDE threat model is used to identify threats to a system. 14

**STS protocol** The Station-To-Station (STS) protocol is a cryptographic key agreement protocol that enables two parties to establish a shared secret over an unsecured communication channel. It integrates the Diffie-Hellman key exchange mechanism with digital signatures, ensuring authenticity and mitigating man-in-the-middle attacks. A notable aspect of STS is its reliance on trusted certificates, typically validated by a Certificate Authority (CA), to authenticate the communicating parties. 58, 60, 71

**TEE** Trusted Execution Environment. A secure area of a main processor. A TEE guarantees that code and data loaded inside it are protected with respect to confidentiality and integrity. v, 39, 72, 81

**Telnet** A network protocol used to provide bidirectional, text-based communication over the internet, primarily for accessing remote servers and network devices. 8

**TLS** Transport Layer Security. A cryptographic protocol that provides end-to-end security for applications that communicate over a network. 2, 13, 30, 66, 73

**TOFU** Trust On First Use. A security model that assumes that the first time a user connects to a server, the server's public key is valid and can be trusted. 42, 48

**TOTP** Time-based One-Time Password. A type of one-time password that is valid for a certain amount of time. 39

**TPM** Trusted Platform Module. A dedicated, tamper-resistant hardware device designed to securely generate, store, and manage cryptographic keys and perform cryptographic operations to protect sensitive data and operations. 36, 38

**trust boundary** A trust boundary marks the point at which the level of trust or the security context changes within a system. For example, a trust boundary is crossed when data is transferred from a secure network to an insecure network. 20

**TTP** Trusted Third-Party. A third-party that is trusted by both parties in a transaction to facilitate the transaction. 13

**Web of Trust** A decentralized trust model in which users vouch for the authenticity of other users. In the context of this thesis, the Web of Trust refers to the OpenPGP Web of Trust. 40

**Well-Known URI** A Uniform Resource Identifier (URI) that is used to point to a resource that is associated with a host or domain. For example, the well-known URI `https://example.com/.well-known/transparency-statement.json` points to the transparency statement of the domain `example.com`. vi, 16, 66, 81

**Wi-Fi** A wireless echnology that allows devices to communicate with each other over limited distances using radio waves. 40