2023 IEEE International Conference on e-Business Engineering (ICEBE) | 979-8-3503-2555-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICEBE59045.2023.00049

Why Johnny was allowed to send this email: How to better provide transparency in email security towards the recipient

Ronald Petrlic
Faculty of Computer Science
Nuremberg Institute of Technology
Nuremberg, Germany
ronald.petrlic@th-nuernberg.de

David Stiegler
Faculty of Computer Science
Nuremberg Institute of Technology
Nuremberg, Germany
stieglerda78912@th-nuernberg.de

***Abstract*—Email security is not transparent at all today. It depends on opportunistic transport encryption: Mail Transfer Agents (MTAs) try to establish a TLS-secured connection before transmitting an email, but if it is not possible, the email is simply transmitted unencrypted. First approaches exist that provide transparency towards the sender of an email before the actual transmission of an email with potentially sensitive content is done. The sender decides whether the connection is "good enough" for the specific content to be sent via email.**

However, recipients of emails still do not have any form of transparency whether incoming emails were properly secured during transport. This leads to situations where recipients complain about received emails (with sensitive content) at data protection authorities (DPAs). In this paper, we come up with an approach to close the transparency gap and inform the recipient about the confidentiality level of an email and that the security mechanisms to protect the data were appropriate.

1. Introduction

Email security lacks *transparency* for users. If users do not employ end-to-end-encryption (with PGP or S/MIME), they normally do not know whether emails are transmitted *protected* or not [1]. This is due to the use of opportunistic transport encryption: The sending mail transfer agent (MTA) tries to establish a secure TLS connection to the receiving MTA using STARTTLS to protect the email to be transmitted. However, if anything goes wrong (e.g., if the receiving MTA does not support TLS or a MITM attacker downgrades the connection), the email is transmitted without any protection [2]. Around 90% of emails are transmitted TLS-protected via Gmail servers, according to Google [3], for example—leaving 10% to be transmitted unprotected. The problem with email security is that users do not know if their emails fall into the 90% or the 10%. [4]

Controllers according to Art. 4 (7) general data protection regulation (GDPR) have the obligation to properly secure their MTAs, so that emails can be transmitted in a secure way (Art. 32 (1) GDPR). The German data protection authorities, for example, have published a guideline with specific requirements for MTA configuration [5]. A company or authority that intends to receive emails with *normal risk* (i.e., emails containing personal data) needs to support only the state-of-the-art TLS versions 1.2 or 1.3 (and no older versions). If the company or authority intends to receive emails with *high risk* (e.g., emails containing health data), it needs to support DANE as well.

However, a minority of controllers follows those obligations. A study of around 4000 German health institutions (hospitals, doctors' offices, etc.) has shown that only 1% of the institutions' mail servers were properly configured to be allowed to receive high risk emails with sensitive medical information. [6]

As a consequence, users who intend to send (sensitive) personal data via email would need to check the quality of the connection towards the receiving MTA (i.e., the server configuration of the receiving MTA) beforehand. Only if the security is good enough with respect to the confidentiality level of the data to be sent, the email can be sent—without violating the privacy requirements. Users, thus, need *transparency* beforehand. Such a solution already exits, as we will show in the related work section.

However, users receiving emails still lack transparency. In this paper, we come up with a solution to this problem.

2. Our Contribution

Transparency is a central principle in the GDPR, explicitly stated in Art. 5.

So far, there is no transparency in terms of the security level of an email transmission towards the recipient of an email, though. In this paper, we come up with a solution to tackle this lack of transparency. This solution provides the following advantages:

- The recipient learns that the sending MTA was properly configured.
- The recipient learns that the sender checked the security details of the connection before email transmission.
- The recipient learns that the sender was, thus, allowed to send the email with the specific confidentiality level, as a consequence.

979-8-3503-2555-3/23/$31.00 ©2023 IEEE
DOI 10.1109/ICEBE59045.2023.00049

Moreover, the most important advantage for the recipient in terms of *accountability*—being also a central principle of the GDPR—is that he can provide evidence that his email server was properly configured and he, thus, was allowed to receive the email with the specific confidentiality level (according to the requirements by the German data protection authorities, for example). Such an evidence can get useful in a data privacy proceeding when the data protection authority checks whether a certain email transmission was legitimate and whether the recipient adhered to the requirements.

3. Related Work

KÖRBER ET AL. [7] present a solution that allows the sender to select from predefined confidentiality levels for emails in a user-friendly way. Each confidentiality level is linked to technical security requirements that must be met when establishing an SMTP connection from the source MTA to the target MTA. The solution focuses on sending emails with sensitive content and assumes that the email has one sender, one recipient, and the recipient does not forward the email. Transparency is provided to the sender of an email, but not to the recipient.

Mandatory transport encryption enables emails to be delivered to a server only if the connection used to transmit the email meets certain encryption properties. Properties can be, for example, specific TLS versions (e.g., only TLS 1.2 or 1.3) or concrete cipher suites. Transparency towards the recipient of an email is ensured by the fact that only those emails can be delivered whose connection used for transmission fulfills the properties. Mandatory transport encryption is offered by some email providers, like Posteo [8], [9] and mailbox.org [10]. They offer a TLS guarantee for receiving emails. The providers' email servers accept incoming SMTP connections only if they are TLS-secured with TLS 1.2 or 1.3. However, the configuration of mandatory transport encryption prevents the delivery of emails that are transmitted with weak or no encryption. Although this is not desirable from a security point of view, it is for some practical reasons. Mandatory transport encryption follows an "all-or-nothing"-strategy, applied to all emails (not only sensitive emails).

End-to-end encryption with PGP [11] or S/MIME [12] achieves partial transparency for the recipient: Even if an email is transmitted without transport encryption, transparency (for the content) is achieved as the recipient learns that the content was transmitted encrypted by PGP or S/MIME. However, this does not allow a statement to be made about the metadata, which is still important even for E2E encryption [13].

4. Concept

4.1. Preliminaries: Checking the recipient MTA's server configuration

The solution by KÖRBER ET AL. [7] allows the sender to specify a confidentiality level for an email. The sending MTA checks whether the receiving MTA is properly configured, i.e., the quality of the communication channel is good enough; if this is the case, the email is transmitted.

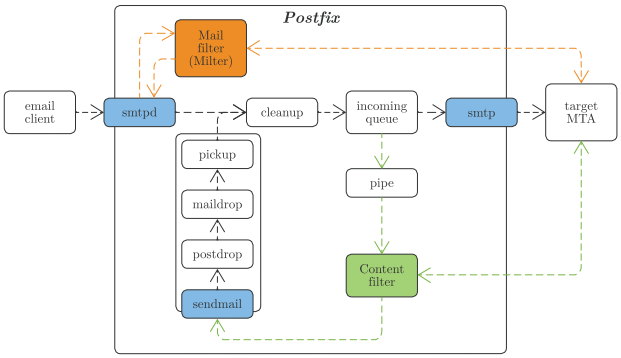

Figure 1: Postfix Architecture (Figure according to [7])

Users who want to send an email specify a keyword, enclosed by curly brackets in the email's subject. If the email to be sent includes sensitive content like health or financial data, for example, the user chooses the confidentiality level *secure*.[1] After composing the email it can be submitted to the sender's MTA. In order to enable wide distribution of the concept, the authors implemented the approach for the open source MTA *Postfix*. The modular architecture of Postfix enables the development of extensions for further processing of emails. Figure 1 shows the processing of an email through the Postfix system.

A content filter is used in the solution to perform the check of the target MTA's settings. If the requirements with regards to the confidentiality level are met, the email will be delivered; otherwise the delivery will be stopped. If the confidentiality level *secure* is used, for example, it means that the target MTA needs to provide state-of-the-art TLS-secured SMTP connections, currently TLS 1.2 and 1.3 [14], with a valid X.509 certificate, and SMTP via DANE[2]. By using a filter, the script is executed for each email. The filter script is written in the common scripting language Bash and can be found on Github: https://github.com/email-security/scripts. The procedure of the script is shown in Figure 2.

The script can be flexibly extended with new keywords. Therefore, it can cover different regulatory or area-specific requirements.

4.2. Adding transparency for the recipient

The recipient of an email should be able to evaluate if the transmission of an email was legitimate in terms of security and privacy requirements. The solution described in the previous section only offers transparency to the sender and currently is not able to involve recipients and inform them. We extend the aforementioned solution and enable

1. The keyword *normal* is default even if no keyword is provided.
2. This is in accordance to the requirements as stated by the German DPAs for emails with sensitive content [5].

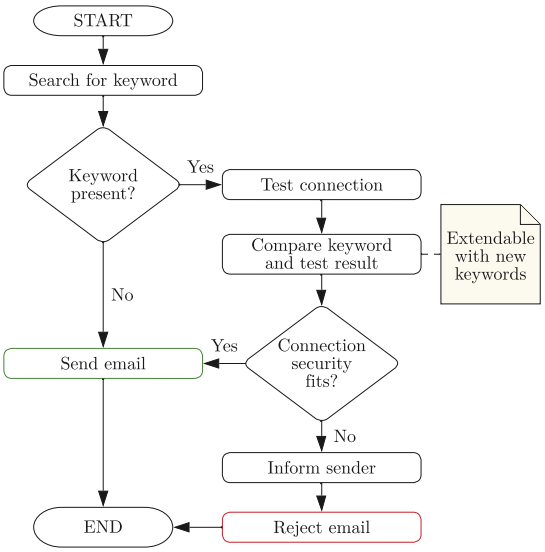

Figure 2: Sequence of verification when sending an email (Figure according to [7]).

transmitting the confidentiality levels and the associated technical information between each MTA on the route and the confidentiality level originally chosen by the sender to the recipient. All the information is then presented in a way the recipient can easily understand.

4.3. Requirements

The requirements to the concept are defined as follows:

1) The sender of an email should be able to specify a confidentiality level for the content of the email to ensure an appropriately secured connection.
2) The quality of the connection to the next MTA (SMTP relay or already the target MTA) should be tested by the sending MTA before the actual transmission of the email—only if the quality is good enough with respect to the confidentiality level as specified by the user, the email should be forwarded. Every other MTA on the route may do this testing procedure to ensure overall transparency of the transmission.
3) The information about the specified confidentiality level and the quality of each MTA-to-MTA connection should be forwarded by the current MTA to the target MTA—the information being integrity-protected to ensure the recipient gets the unadulterated transparency information.
4) The recipient should have transparency about a received email and the confidentiality level specified by the sender. The requirement also includes transparency about the confidentiality levels of each part of the connection if more than two participating MTAs are involved.
5) Transparency can only be granted if the information above is conveyed to the recipient properly; thus, the common access point for users to emails—an email client (mail user agent (MUA))—needs to provide an opportunity to meet this requirement.

4.4. Design decisions

A way for the recipient to retrieve connection details (TLS information and cipher used) with small effort would be to read the header of an email. The *Received:* header field sometimes contains details about the used encryption and can be used to trace the route of an email through several hosts [15]. But the field is not standardized and may be modified multiple times during transport. For this reason, some MTAs like Postfix even do not write this information to the *Received:* field by default [16]. Not only because of that, but also to improve the usability for less technically experienced users, using the *Received:* field does not meet our requirements. It is necessary to transmit connection details in a different way. This could include sending them directly within the email body or as an attachment. Sending this information within the email body allows presenting the connection details in a more human-readable format than in the header, but may confuse the user, as it is not clear which part belongs to the message and which part contains technical information. This could be solved by sending the details as an attachment, but as attachments sometimes are considered malicious, users might distrust them. Finally, we decided to extend the email header by two new non-standard fields, where one contains the sender's desired confidentiality level and the other one carries all connection details such as TLS version or DANE usage. Each MTA that implements our concept will add such a header field to report the confidentiality level to its successor. The recipient's client will then be able to evaluate the entire transmission (as stated in requirement 5).

4.5. Role of the sending MTA

The sender's MTA gathers connection data to its successing MTA as explained in section 4.1. In Figure 3, step *1.* shows the tasks of the solution by KÖRBER ET AL. [7]. We directly build on this solution by continuing with step *2.* and *3.* (see Figure 3).

The connection data is transmitted via the email header. Therefore, we need to define user-defined header fields, which can be seen in Figure 3 as *X-Transparency-Info* and *X-Confidentiality*. Defining custom fields is allowed but must be declared with the prefix *X-* [17]. Requirement 3 requires that the sender's desired confidentiality level (depending on the quality of the TLS connection) should be transmitted to the recipient. This information will be stored once in the *X-Confidentiality* field. Furthermore the requirement states that the actual technical details about the connection should also be transmitted to the recipient. This includes information

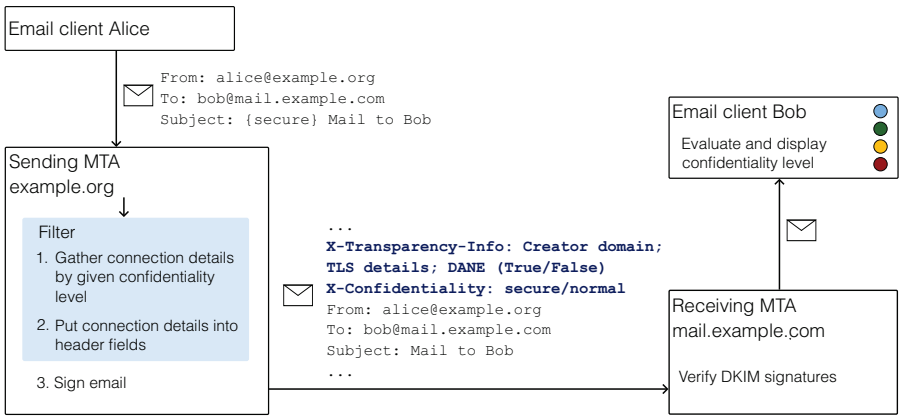

Figure 3: Overview of the concept

about the TLS version and DANE usage, which is stored in the *X-Transparency-Info* field, as well as the domain of the MTA that created this field to enable associating the determined confidentiality level to the corresponding checked part of the route. To fulfill this requirement and protect the header field from changes/manipulation, the last step before transmitting the email is to sign the data. As mentioned above, some parts of an email may be modified during transport (e.g., the *Received:* field). Thus, it is important to choose those parts of the email that do not change the signature to keep it verifiable by the receiving MTA. We decided to use DKIM (DomainKeys Identified Mail) [18] for signing. DKIM allows signing of specific parts of an email such as our two user-defined fields, the *From:*, *To:*, *Subject:* fields or even the message itself. For our purposes it is sufficient to protect the two user-defined header fields.

4.6. Role of the receiving MTA

The receiving MTA only needs to verify the signed parts of the incoming email according to the standard [18]. So no further adoptions need to be done to Postfix, except for properly setting up DKIM.

The verification result, which the receiving MTA writes to the email header after the signature verification, is important for the evaluation within the receiving email client.

To sign a message with DKIM and let a receiving MTA verify the signature, asymmetric encryption is used. The required public key for verification by the receiving MTA is stored as a TXT resource record in the DNS of the sending domain. The verifying MTA queries the record and uses the public key for verification. Then it creates a separate header field [19] and adds the verification result (or if more signatures exist it will verifiy each of them) to it.

4.7. Extending the email client

As requested in requirement 5, the confidentiality of the transmission needs to be presented appropriately to the user. To make this easy to grasp, the confidentiality level will be depicted as a kind of ”traffic light” with four states:

TABLE 1: The four *traffic light* levels and their prerequisites

Traffic light	Prerequisite
Blue	DANE and TLS 1.2/TLS 1.3
Green	TLS 1.3 or TLS 1.2
Yellow	TLS 1.1 or TLS 1.0
Red	SSL or unencrypted

If the sending user specified the confidentiality level *secure*, for example, and the communication channel was good enough (i.e., TLS 1.2 or TLS 1.3 with DANE), the email client of the recipient shows the blue light—showing the recipient that everything is fine. This traffic light is also used for the more detailed view on the transmission. Each MTA on the route which has performed a connection test (see requirement 2) will add confidentiality information to the email and, thus, it is possible to trace back the transport way and show the recipient all partial MTA-to-MTA connections and their confidentiality level.

To show the functionality of the concept, we developed a prototype of a client extension. Many companies use Microsoft Outlook as email client [20]. This is a solid basis for implementing the traffic light functionality. Both, Outlook desktop client as well as the web version, support the development of custom plugins, known as Add-Ins [21]. Figure 4 shows the user interface with our plugin in use.

5. Evaluation

Both information added to an email, the sender-defined confidentiality level (*X-Confidentiality*) and the gathered connection details (*X-Transparency-Info*) are not classified as confidential, as everybody can easily check the connection to a destination MTA by using command line tools like OpenSSL. We are more interested in checking how good our approach protects both fields’ integrity to give the recipient an unadulterated view on transparency of the transmission. To evaluate the approach regarding

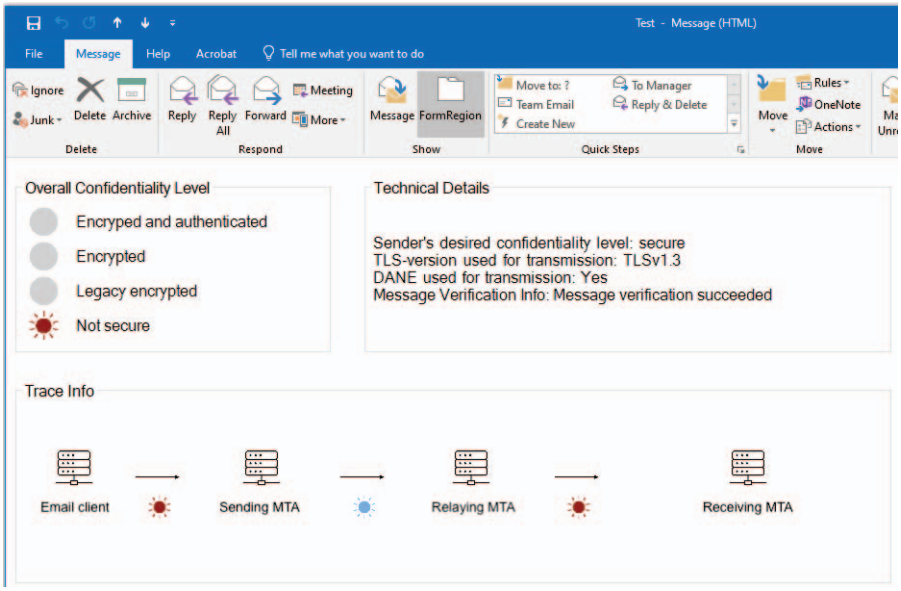

Figure 4: Prototype design of the user interface of the plugin implemented as an Outlook Form Region (separate view within an active email item window). It consists of three views, each of them represents a different level of detail. The upper left section shows the confidentiality level of the overall transport connection whereas the block below depicts the confidentiality level of each single part between the MTAs on the route. The upper right box shows more details about the connection such as the sender-defined confidentiality level or whether DANE was used or not.

integrity, we assume that the attacker sticks to the rules (like a semi-honest adversary [22]). On the one hand, an attacker makes effort to gain as much information as possible from an email, whereas on the other hand, rules are not being broken deliberately. We are now going to describe two different kinds of attacks (passive and active) and how the developed concept can deal with them.

Passive attacks: An attacker may have access to every network device on the transport way to sniff traffic [23], for example, on a router. If the sender decides to use the keyword *secure*, the email will be transmitted encrypted and, thus, is not readable to any attacker sitting in between sender and receiver. The attacker will only see the IP packet and its source and destination address whereas the confidential SMTP packet itself is encrypted with TLS and the attacker cannot read it.

The only way to read an encrypted email during transport is for the attacker to get direct access to an MTA, because this is where the transport encryption ends. Therefore, the sender must assume that MTAs treat processed emails confidentially [24]. An attacker gaining access to an MTA could violate the confidentiality of an email but not their integrity, which ensures that a recipient gets the email unadulterated. Nevertheless the recipient will not be informed about such an incident, because the DKIM signature was not affected by the attack. In its current development state, the solution described in chapter 4 cannot detect such an attack but it should be pointed out that having access to an MTA would raise a major issue to the overall security of the email infrastructure.

Active attacks: Active attackers do not follow the rules and are more malicious regarding the semi-honest model because they try to manipulate email data, which is a serious threat to their integrity. An attacker could try to suppress the clear-text-transmitted STARTTLS command [2] in the checking phase (filter step 1 in Figure 3). An email labeled as *normal* would be sent in clear-text but the keyword *normal* is written to the header and a recipient could see that the connection was not secure compared to the connection test results. An attempt to manipulate the two header fields carrying confidentiality information would result in a failed verification of the DKIM signature. This would be recognized by the Outlook Add-In which evaluates the *Authentication-Results* fields, each carrying the verification result to a corresponding DKIM signature. If one or more of these fields contain *dkim=fail* the email will be considered as if it was unsigned and—to provide transparency—shows a reason message to the user. Similar to manipulating header fields, the total suppression of DKIM signatures would also be recognized by the Add-In, which then marks the entire connection as unsigned with the red light.

5.1. Restrictions

Our concept neither considers the connection parts between the sender and the sender's MTA nor the parts between the recipient's MTA and the recipient. Therefore, additional client functionality would be required to also

check the confidentiality of these parts. This would require additional software components on the client such as OpenSSL or tools like *testssl.sh* [25] and would result in a higher effort and deeper technical understanding for the end user.

In case there are forwarding MTAs on the route to the receiving MTA, every forwarding MTA needs to implement the concept to ensure transparency along the entire route. Otherwise transparency could only be granted between MTAs that implement the concept and their next hops. To make this clear to the recipient, the client extension evaluates every part of the MTA-to-MTA connections and if no connection test could be performed by an MTA, the traffic light on such a section will be shown in red. With every MTA implementing the concept, transparency will gradually evolve. Therefore, it is not just an "all-or-nothing" approach but would make a contribution to transparency step by step, just like mapping an until then unknown road network.

6. Conclusion and Outlook

The recipient of an email learns about the confidentiality level of a received email as specified by the sender. This allows the recipient to act accordingly—i.e. take appropriate security measures.

Our approach is based on standard DKIM and, thus, can easily be deployed in practice without introducing any further mechanisms.

Our concept can be extended with additional features. For example, the plugin at the receiver side sees the confidentiality level of an email and could inform the recipient if he/she intends to forward the email that proper security measures need to be taken into account, i.e. that the connection towards the recipient should be checked according to our proposed concept.

References

[1] R. Petrlic, "The general data protection regulation: From a data protection authority's (technical) perspective," *IEEE Security Privacy*, vol. 17, no. 6, pp. 31–36, 2019.

[2] Z. Durumeric, D. Adrian, A. Mirian, J. Kasten, E. Bursztein, N. Lidzborski, K. Thomas, V. Eranti, M. Bailey, and J. A. Halderman, "Neither snow nor rain nor mitm... an empirical analysis of email delivery security," in *Proceedings of the 2015 Internet Measurement Conference*, 2015, pp. 27–39.

[3] Google LLC, "Email encryption in transit," accessed: 2023-10-01. [Online]. Available: https://transparencyreport.google.com/safer-email/overview

[4] L. Dessani and R. Petrlic, "E-Mail-Sicherheit auf dem Prüfstand," *Springer Datenschutz und Datensicherheit*, vol. 45, pp. 534–540, 2021.

[5] Orientierungshilfe der Konferenz der unabhängigen Datenschutzaufsichtsbehörden des Bundes und der Länder vom 27. Mai 2021, "Maßnahmen zum Schutz personenbezogener Daten bei der Übermittlung per E-Mail." [Online]. Available: https://www.datenschutzkonferenz-online.de/media/oh/20210616_orientierungshilfe_e_mail_verschluesselung.pdf

[6] C. Lange, T. Chang, M. Fiedler, and R. Petrlic, "An Email a Day Could Give Your Health Data Away," in *Proceedings of the Data Privacy Management, Cryptocurrencies and Blockchain Technology ESORICS 2022 International Workshops, DPM 2022 and CBT 2022*, 2023.

[7] M. Körber, L. Dessani, and R. Petrlic, "MTA extension for user-friendly enforcement of mandatory TLS encryption," in *Proceedings of the 37th International Conference on Advanced Information Networking and Applications (AINA-2023)*, 2023.

[8] Posteo e.K., "What is the tls-sending guarantee and how do i activate it?" accessed: 2022-06-22. [Online]. Available: https://posteo.de/en/help/activating-tls-sending-guarantee

[9] ——, "What is the tls-receiving guarantee and how do i activate it?" accessed: 2022-06-22. [Online]. Available: https://posteo.de/en/help/activate-tls-receiving-guarantee

[10] Heinlein Hosting GmbH, "Ensuring e-mails are sent securely," accessed: 2022-06-22. [Online]. Available: https://kb.mailbox.org/en/private/e-mail-article/ensuring-e-mails-are-sent-securely

[11] H. Finney, L. Donnerhacke, J. Callas, R. L. Thayer, and D. Shaw, "OpenPGP Message Format," RFC 4880, Nov. 2007. [Online]. Available: https://www.rfc-editor.org/info/rfc4880

[12] B. C. Ramsdell, "S/MIME Version 3 Message Specification," RFC 2633, Jun. 1999. [Online]. Available: https://www.rfc-editor.org/info/rfc2633

[13] D. Poddebniak, C. Dresen, J. Müller, F. Ising, S. Schinzel, S. Friedberger, J. Somorovsky, and J. Schwenk, "Efail: Breaking S/MIME and OpenPGP Email Encryption using Exfiltration Channels," in *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, 2018, p. 549–566.

[14] K. Moriarty and S. Farrell, "Deprecating TLS 1.0 and TLS 1.1," RFC 8996, Mar. 2021. [Online]. Available: https://www.rfc-editor.org/info/rfc8996

[15] R. T. Braden, "Requirements for Internet Hosts Application and Support." [Online]. Available: https://www.rfc-editor.org/info/rfc1123

[16] W. Z. Venema, "Postfix configuration parameters." [Online]. Available: http://www.postfix.org/postconf.5.html

[17] D. Crocker, "Standard for the format of ARPA Internet Text Messages." [Online]. Available: https://www.rfc-editor.org/rfc/rfc822

[18] D. Crocker, T. Hansen, and M. Kucherawy, "RFC 5672: DomainKeys Identified Mail (DKIM) Signatures." [Online]. Available: https://www.rfc-editor.org/rfc/rfc6376.html

[19] M. Kucherawy, "Message Header Field for Indicating Message Authentication Status." [Online]. Available: https://www.rfc-editor.org/rfc/rfc8601.html

[20] Nielsen Company und empower GmbH, "Die große Office-Software Studie," 11 2020. [Online]. Available: https://www.empowersuite.com/de/wissen/office-studie-deutschland

[21] J. Hart, G. Hogenson, L. L. Cannon, D. Mabee, G. Warren, N. Schonning, T. G. Lee, and M. Jones, "Get started programming VSTO Add-ins." [Online]. Available: https://learn.microsoft.com/en-us/visualstudio/vsto/getting-started-programming-vsto-add-ins?view=vs-2022

[22] Y. Lindell, *How to Simulate It – A Tutorial on the Simulation Proof Technique*. Cham: Springer International Publishing, 2017, pp. 277–346. [Online]. Available: https://doi.org/10.1007/978-3-319-57048-8_6

[23] R. Barnes, B. Schneier, C. F. Jennings, T. Hardie, B. Trammell, C. Huitema, and D. Borkmann, "Confidentiality in the Face of Pervasive Surveillance: A Threat Model and Problem Statement," RFC 7624, Aug. 2015. [Online]. Available: https://www.rfc-editor.org/info/rfc7624

[24] J. Davies, *Implementing SSL / TLS Using Cryptography and PKI*. John Wiley & Sons, Incorporated. [Online]. Available: https://ebookcentral.proquest.com/lib/thnuernberg/detail.action?docID=706899

[25] D. Wetter, "testssl.sh," GitHub, September 2022. [Online]. Available: https://github.com/drwetter/testssl.sh