SAE - Secure Application Email Architecture

Francesco Gennai[1], Fabio Sinibaldi[1†], Loredana Martusciello[2], Marina Buzzi[2]

[1] *Institute of Information Science and Technology - ISTI, National Research Council of Italy - CNR*
[2] *Institute for Informatics and Telematics - IIT, National Research Council of Italy - CNR*

Abstract

Email architecture is one of the first Internet services and it is not secure by design. The TLS protocol has been defined to add a security level between email servers. However services relying on email protocols can inherit security vulnerabilities mainly due to DNS security flaws. In this paper a new architecture model is proposed: the Secure Application Email Model, which is able to avoid typical attacks of the Internet email architecture. The proposed architecture tries to offer a secure messaging system built on top of the existing email technologies. The described security layer allows to identify trusted nodes and domain names, without relying on the underlying technologies in order to allow for faster adoption and a higher level of trust. With this scalable solution, the technical effort is concentrated on a set of qualified service providers who have adequate technical skill. The solution is fully compatible with the current Internet email system, without any negative impact from the insecurity of the DNS.

Keywords

Secure Application Email, Certified Electronic Email, DNS, DNSSEC

1. Introduction

In this study, we analyze some of the current weaknesses of the Internet email architecture for the development of trusted email services, which rely on enhanced security features. The current Internet email architecture has some characteristics that could be very attractive for the development of new trusted services (e.g. Certified Email systems, ETSI Registered Email [1]), but also it has critical issues that impose a deep analysis of the Internet email architecture before defining such a service. Any solution that aims to create a trusted community requires two mechanisms: authentication and trusting. The aim of this study is to introduce these concepts at a high abstraction level, to investigate some of the technologies and solutions currently available, analyzing main pros and cons. However these solutions may add a burden on organization managing services, that could be too effort-demanding for small organizations. At this aim, a new low-impact solution at application level, is proposed for the email system. The paper is organized in 5 sections. Section 2 outlines the Internet Email technologies at the time of writing and its inherent security flaws. We then describe how currently proposed solutions (e.g. DNSSEC..) bring some issues, which led us to propose a different approach. Section 3 introduces and discusses the proposed model. A solution of trusting can impact on different protocols and levels. In this paper we focus only on trusting solutions (parties that are a trusted solution) which rely on the email infrastructure and protocols. In section 4 we compare three different security layer designs built on top of the existing email architecture (Naming Conventions, GNS and a 2-Tier Super Peer architecture) and discuss which one seems better fitting to SAE requirements. Last, in section 5 we draw conclusions on the outcomes of the present work and how the ultimate design choices depend on multiple factors. We then conclude our work with considerations on possible related future works.

ITASEC 2023: The Italian Conference on CyberSecurity, May 03–05, 2023, Bari, Italy
†Corresponding author
EMAIL: francesco.gennai@isti.cnr.it (F. Gennai); fabio.sinibaldi@isti.cnr.it (F. Sinibaldi); loredana.martusciello@iit.cnr.it (L. Martusciello); marina.buzzi@iit.cnr.it (M. Buzzi);
ORCID: 0000-0001-6453-0113 (F. Gennai); 0000-0003-1013-6203 (F. Sinibaldi); 0000-0002-9900-7944 (L. Martusciello); 0000-0003-1725-9433 (M. Buzzi)

2. Background

In this section we describe the technological aspects of the Internet Email Architecture at the time of writing, which, in our vision, will constitute the transport layer for SAE. We then describe some critical security aspects of DNS and SMTP. We also analyze how DNSSEC tries to solve some of these security flaws, while introducing a certain level of management complexity that makes its adoption difficult.

2.1. Internet Email Architecture

The basic concepts of electronic mail and protocol are briefly described for the reader's convenience. The Internet email architecture is described in the RFC 5598 [2]. We recall roles and functions of the most important components of the Message Handling Service (MHS) that represents the whole Internet email architecture. For reader convenience, we recall the components included in the message delivery (Figure 1):

- Message User Agent (MUA): the component that creates the email message and submits it to the Message Handling System. The email can be submitted by the Message Submission Agent or by some direct access to the Message Transfer Agent.
- Message Submission Agent (MSA): the component that accepts the email message from the MUA. It is a SMTP server with some special functions related to the submission events [3]. Often the MSA runs on the same node of the MTA.
- Message Transfer Agent (MTA): the component that manages the relays of the email message toward its destinations. The MTA implements both client and server functionality, typically by the SMTP protocol.
- Message Delivery Agent (MDA): the component that delivers the email message to external entities such as the Message Store (MS) or other external applications.

The MSA and the MTA are email servers with differences in some of their specializations. The MDA can run on the same node of the MTA.

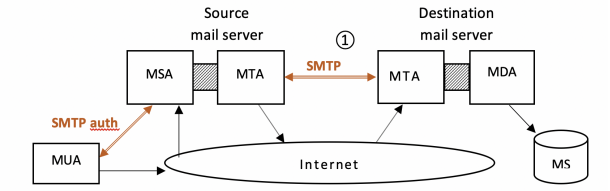

Figure 1: Flow of an email message inside the Internet email architecture.

2.2. Relay between source MTA and destination MTA

As mentioned before, SMTP is the Internet standard that defines formats and protocol (the dialog between client and server) for sending and relaying emails.

The Figure 2 illustrates an example: a client is sending an email to a destination in Internet, it transmits the email to its SMTP server, which relays (❶) each message to the SMTP server for the recipient's domain. This means that messages (email, mailing lists messages) can transit across multiple SMTP relays before getting to their final destination.

The SMTP protocol is not secure by definition. To enable encrypted transmission of messages over the SMTP the SMTP extension STARTTLS has been defined by IETF [4]. However, the TLS application is optional. This means that there is an amount of email traffic in clear text, thus vulnerable to be intercepted by malicious actors.

Specifically there are two types of attacks on the security level of a SMTP connection:

- STARTTLS downgrade attacks
- DNS hijacking attacks

The result of the first type of attack is a SMTP connection in clear text, also when the destination server supports TLS. The result of a DNS hijacking attack is that the email is delivered to the attacker email server (which will relay the message to the original destination server, so nobody will notice the unusual traffic). As originally conceived, SMTP does not support the confidentiality of messages in transit or authenticating messages upon receipt. Due to these issues, passive observers can read message content on the wire, and active attackers can alter or spoof messages. To fix this security gap, the IETF has developed protocol extensions, such as STARTTLS, DKIM [5], DMARC [6], SPF [7], DANE [8], and MTA-STS [9] to encrypt SMTP sessions, authenticate parties and to avoid TLS downgrade attacks. Specifically:

- mechanisms for email source authentication are represented by SPF, DKIM, DMARC.
- solutions for email destination authentication and to avoid downgrade attacks include DNSSEC and MTA-STS.

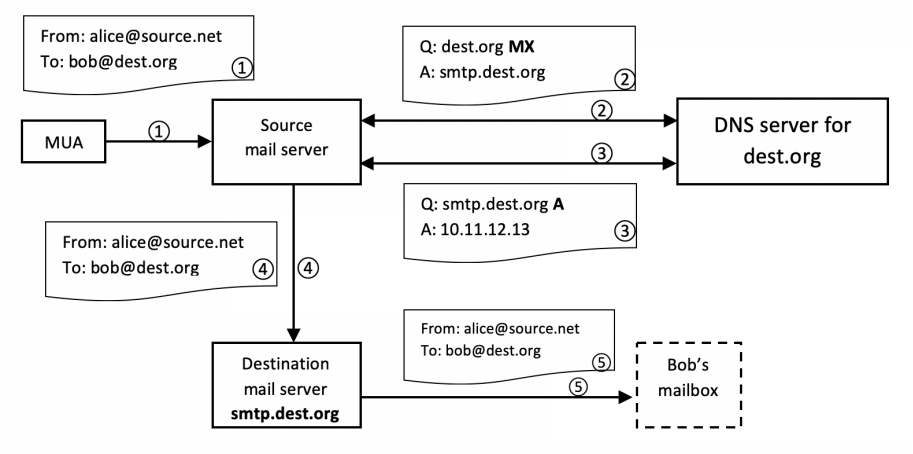

Figure 2: SMTP Protocol—A client sends outgoing email by connecting to its organization's local SMTP server (❶). The local server performs a DNS lookup for the email exchange (MX) record of the destination.com domain, which contains the hostname of the destination's SMTP server, in this case smtp.dest.org (❷). The sender's server then performs a second DNS lookup for the destination server's IP address (❸), establishes a connection, and relays the message (❹). The message is delivered to the recipient mailbox (❺).

It is important to observe that there is not a strong and currently widely applied mechanism to authenticate the recipient server, which is discovered by the sender email server exploiting a specific DNS record (MX record). A service developed on top of the Internet email architecture should be validated against the properties of the Internet email architecture.

In the next section the DNS and DNSSEC are introduced.

2.3. DNS and DNSSEC

We noted how the resolution of the target email server relies on the MX record on DNS. However this is quite problematic since DNS is vulnerable to various attacks [10][11]. Domain Name System Security Extensions (DNSSEC) are a suite of extension specifications by the Internet Engineering Task Force (IETF). DNSSEC tries to bring security to DNS by implementing a hierarchical Private Key Infrastructure in which each zone signs the public keys of all its children zones. This design is simple in theory but lots of challenges make it difficult to adopt [12][13][14], while still presenting some security flaws and overall costs [13][15]. DNSSEC is in fact more demanding in terms of cross-domain coordination since each parent and child pair need to manage both DNS server updates but also periodic key changes. These requirements become really challenging when dealing with lots of

different administrative domains, resulting in lots of misconfigurations [13][16]. It is also worth noting that the high number of human driven activities can be a threat per se, since such activities tend to be more error-prone than automated ones [13]. Another critical aspect of DNS and DNSSEC is the underlying trusted computing base (TBD) [17]. In DNS and DNSSEC this is hard to define because CNAME and DNAME records might involve completely different domains like what happens with collateral damage from DNS Censorship. When a chained name resolution enters a censored zone, it triggers the censorship mechanism to react affecting communication beyond the censored networks [18]. Another challenge in designing a trust mechanism is given by the vast adoption of cloud computing. Cloud computing advantages are well known and the market interest has continued to grow in the last decade. However, a survey from Fujitsu reports that more than 80% of cloud customers worry about security issues [19]. The currently mainly approach is based on a Service Level Agreement (SLA) but this doesn't practically satisfy trust requirements. For instance, in 2016 CloudFare suffered a major data leakage affecting 2 millions websites including Uber and 1password. In 2017 Microsoft Azure failures affected related services for 8 hours, and in the same year Amazon Web Services were affected by a security breach that exposed personal information of about 200 millions of US voters. In order to overcome these challenges, trust mechanisms are put in place both for identifying a node and to evaluate its trust status by analyzing its behavior. In section 4 we describe how SAE uses a security layer on top of the existing architecture in order to grant a certain level of security regardless of the underlying infrastructure trustability.

2.4. MTA-STS

Another security extension currently in use is the Mail Transfer Agent Strict Transport Security [20], proposed by IETF. It allows MTAs to publish a security policy via a URL advertised on the DNS. A security policy informs servers about the ability of secure communication of the recipient, thus avoiding possible TLS downgrade attacks. The server's certificate is signed by a trusted CA, and invalid or expired certificates must result in aborted transfers. MTA-STS relies on information stored on DNSs, which in turn become a weakness of the chain of trust and is prone to misconfiguration. Recent studies have highlighted how a low percentage of SMTP sites announce MTA-STS information [13]. We will discuss in section 4 how SAE tries to use a similar approach without relying on DNS for the discovery of security policies and capabilities of involved nodes.

3. The Secure Application Email Architecture

A new model relying on email architecture is proposed with the aim to fix issues highlighted in the previous section. We call this model the Secure Application Email Model. It recalls the OSI layer concepts where the Internet email is the underlying level that offers its service to the overlying Secure Application Email level. The SMTP protocol could authenticate the destination email server by using the TLS extension in order to prevent the email delivery to a malicious email server. But the current use of the opportunistic TLS provides only confidentiality against a passive MITM attack. To enhance the security level of SMTP-TLS to prevent an active MITM attack, the SMTP client should verify that the SMTP server presents a valid certificate at the beginning of the TLS session, that means a certificate released for that server by a trusted certification authority. This permits the SMTP client to verify at the beginning of the SMTP session if the SMTP server is a trusted server and to drop the session in case of a verification failure. If the check passes, the SMTP server becomes a trusted SMTP server inside a community. These concepts are not applicable to the whole Internet MHS where there aren't commonly trusted Certification Authorities and where there are SMTP servers that don't support TLS yet [21]. By TLS authentication we can identify a trusted subset of the Internet MHS to which the trusted email servers (MTA) belong, but we should also have a trusted DNS infrastructure to authenticate the relationship between a domain name and its email server by means of the MX record.

It is important to observe that it is possible to split delegation duties:

- Technical delegation: it requires a technical specialized action on Internet resources under the control of the domain owner (i.e. adding a MTA-STS record to a Name Server, activating a HTTP redirection). By these actions the domain owner delegates to an Internet provider the control of a service.
- Administrative delegation: it shouldn't require any technical specialized actions on the Internet resources controlled by the domain owner. It is based on an agreement between the domain owner and the Internet provider.

SAE introduces the concept of Email Authority (similar to the well known Certification Authority) by reducing the specialized technical action that a domain owner should accomplish to enhance the security of the email domain.

SAE enables the definition of email perimeter for qualified email domains. Any communication external to the SAE perimeter (i.e with the worldwide email community) can be better controlled. For example, any email coming from an external domain could be subject to specific controls and could be tagged before being delivered to the SAE user mailbox. A sending SAE user could receive a warning when an email is addressed to an external domain.

We observed that, without a trusted DNS infrastructure (i.e. DNSSEC), a domain name that points to a trusted email server could be redirected to an untrusted email server (i.e. the attacker's email server). While the email servers belonging to the trusted subset of MHS are managed by skilled organizations with high technical knowledge to manage these technologies, thousands of email domain names that should be managed by a trusted email server are actually managed by organizations without such technical high skills. For example, currently in Italy there are more than 240.000 domain names of certified email (Italian PEC system [22]). All the organizations that manage these domains should update their name servers to DNSSEC as soon as possible to ensure a reliable security level. Let us now introduce some definitions to introduce the following components of a trusted subset of the Internet MHS:

- **trusting policy**: a security policy shared among all the email servers belonging to the same trusted MHS subset.
- **trusted MHS subse**t: a subset of email servers that share a common security policy.
- **trusted email server**: an email server that adopts the trusted policy.
- **trusted email domain name**: an Internet domain name managed by a trusted email server.
- **trusted email address**: an email address of mailboxes local to a trusted email server.
- **trusted sender**: the user that submits the email to its local trusted email server. How the local email server identifies a trusted sender is out of the scope of this document.
- **trusted recipient mailbox**: the mailbox where a trusted email server delivers trusted email.
- **trusted email path**: a trusted path between a trusted sender and a trusted recipient mailbox.

As discussed in section 2.3, in the current Internet email architecture we cannot rely on DNS to identify a trusted email domain name, due to its insecurity. On the other hand, the DNSSEC adoption on a large scale is still difficult.

In a trusted MHS subset we could identify a trusted email path where the trusted sender sends an email to what he thinks to be a trusted recipient mailbox. The email is submitted to the local trusted email server. The local trusted email server extracts the domain from the trusted recipient address and queries the DNS to get the IP address of the recipient email server to which it relays the trusted email. As we have already highlighted, the DNS could be a point of failure, so we cannot speak of trusted MHS subset until this problem is solved.

Specifically, in the trusting path of an email inside the trusted MHS subset there are two points of failure where the MHS interacts with the insecure DNS:

- **identification of a trusted domain**: DNS query to get the MX hostname of the domain that should be trusted/authenticated.
- **identification of a trusted email server**: DNS query to get the IP address of the MX hostname that should be a trusted email server (authenticate)

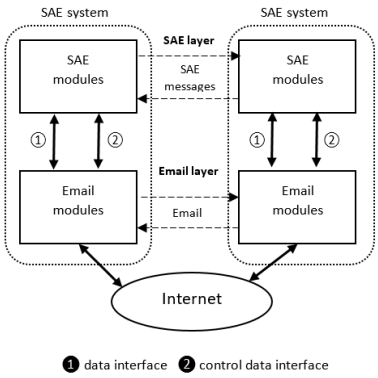

Figure 3: The SAE model.

Figure 3 shows the logical scheme of the SAE architecture. The current Internet email architecture becomes the SAE transport layer. The new architecture does not impact on the email submission and delivery phases, as shown in Figure 4.

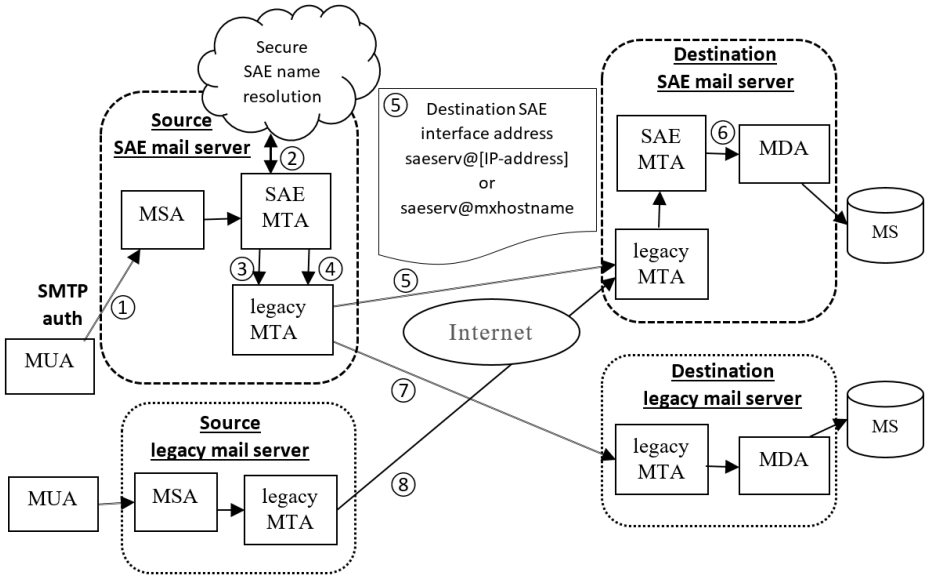

Figure 4: Flow of an email message inside the SAE email architecture. A client sends outgoing email by connecting to its organization's local SMTP server (❶). The SAE MTA queries the Secure SAE name resolution system to check if the destination domain belongs to the SAE community: it gets the destination SAE interface address and the cryptographic public key of the destination SAE mail server (❷). The SAE MTA composes the encrypted S/MIME SAE transport message and enqueues it to the legacy MTA (❹). The legacy MTA relies the SAE transport message to the destination SAE mail server (❺). The destination SAE mail server extracts the original message (from the MIME SAE data part) and the envelope recipient(s) address(es) (from the MIME SAE protocol control data part) of the SAE transport message and delivers the original message to the MDA (❻). If the destination domain does not belong to the SAE community (❷), the SAE MTA enqueues the original message to the legacy MTA (❸) that will relies it to the destination legacy email server (❼). Any Internet email server can rely standard email to a SAE mail server as it belongs to the Internet MHS (❽).

The SAE architecture defines two MIME structures:

- SAE transport message: it is a MIME formatted message composed of two parts, the SAE data part that contains the original email message and the SAE protocol control data part.

The transport MIME structure is encrypted by the public key of the recipient SAE mail server (S/MIME).

- SAE control message: it is a MIME formatted message composed only by a protocol control data part, used for sharing SAE protocol information between SAE systems.

In the SAE architecture:

- the SAE payload is the original email, in its classic format: header and content.
- the SAE transport message is the payload of the underlying Internet email layer

The main features of the new architecture include:

- the ability to carry protocol data along with the original email (the one presented to the MSA). This feature enables the definition of new protocols that build on the current Internet email architecture, used as the transport layer.
- an enhanced security in the transport of the original message. The SAE transport email can be encrypted by a cryptographic public key of the destination MTA. These produces two advantages:
  - In the current Internet email architecture, end-to-end email encryption, as provided by PGP [23] and S/MIME [24], could leave metadata visible everywhere along the message's path [25]. This information is potentially exposed to attackers, for example in the case of a successful MITM attack. Although greater adoption of end-to-end encryption would undoubtedly be beneficial, for now, the overwhelming majority of messages depend solely on SMTP and its extensions for protection [21]. As SAE adds the encryption to a higher layer, the full original email, metadata + content, is encrypted during its transport also in the absence of the end-to-end encryption (metadata such as subject, sender and recipients are encrypted).
  - In the current Internet email architecture the TLS can be used, other than encrypting the SMTP session, to authenticate the remote mail server (MTA). In SAE it is possible to use the IP address literal [26]. The SAE transport email can be addressed to the remote SAE system by the address literals format. This permits the use of SMTP protocol without TLS and reduces the risk of a DNS MITM attack since it avoids the query to the DNS (to discover the destination email server).
    It is important to note that a MITM attack in SAE architecture can produce only a denial of service with no disclosure of the email content.
- A simplified handling of the relay phase. The current Internet email system manages email addressed to one or more recipients. It optionally manages the transport phases, in case of multiple recipients of the same architecture components (MSA, MTA or MDA), by avoiding the sending of multiple copies. At any stage in the processing the email can be split into multiple copies, each of which having its own subset of destination addresses. For example, at the relay stage of an email between two MTAs, with multiple destination addresses, some may be subject to temporary errors resulting in, only limited to these addresses, one or more retry actions, depending on the sender MTA's retry policy. It turns out that the email will be split into multiple copies, transferred with subsequent events or rejected (if the destination addresses received a permanent error or reached the retry limit period for a temporary error). This indeterminacy, although controllable and manageable, could induce criticism in the definition of a protocol that requires greater homogeneity in the various stages of an email's progress. This is, for example, the case of certified email systems, where in the relay phase between the sender and the recipient MTA, the concept of the receiving MTA taking charge of the email assumes particular importance, even from a legal point of view.

Further detailed analysis of the proposed solution is provided in *Secure Application Email Model - SAE* [27].

An important remark is that the SAE architecture is fully compatible with the current email architecture. To avoid the risks of uncontrolled level of security management (security features are handled by qualified Providers instead of small low-skilled organizations), the trusted email servers should rely on mechanisms alternative to the DNS. In the next section, we explore these possibilities.

4. Security Layer

In the previous sections we described how our application relies on the underlying mail architecture as a transport layer. From a security point of view this is problematic because mails are not strictly secure [13][28]. Some attempts have been made in order to build a more reliable mail architecture [29], but then DNS easily becomes the weak link of the chain since communication is not secure and their management can lead to misconfigurations (both malicious and accidental) [11] [14]. In 2.3. we noted how DNSSEC partly addresses these issues, but its costs and complexity slow its adoption.

For these reasons, we want to rely on the underlying technology as much as possible, designing SAE in order to be fully in charge of security. We also want to reduce the operational overhead for ordinary Node managers, eventually accepting major implications for Providers. In terms of security, our application needs to: 1) identify a Node participating in the messaging community 2) Identify the owner Node of a certain Domain Name. In this section we describe a technological solution that constitutes a security layer on top of the existing technologies in order to fulfill the requirements stated above. To do this, the reader is presented with three different approaches that might be adopted: 1) a simple naming convention 2) GNS (a DNS by GNUnet) 3) a 2-Tier super Peers architecture. We then conclude on considerations about the three different approaches and the motives that drove us into selecting one.

4.1. Naming Convention

A very simple direct approach to our security layer might be a Naming Convention, that is an agreement on a certain format of Domain Names like the presence of a starting suffix (e.g. “SAEDN”). SAE could very well impose this requirement to the participating nodes of the community in order to assess if a certain Domain is expected to be managed by a secure Node, meaning that only secure communication would be allowed. This would solve the TLS degradation attempt of a MITM, since such plain communication would be denied. The advantage of this approach is its simplicity, that would allow for a faster adoption. However, it’s important to note that a Naming Convention would still need a Trusted Registry of servers and related public keys (e.g. a Trusted List) in order to check if the destination server is actually a trusted one and establish a secure connection.

4.2. GNS

Another approach that we find of interest comes from the project GNUnet. GNUnet is a network stack for building decentralized and privacy-preserving distributed applications [30], partly founded by European programs like Horizon2020 and FP7. GNUnet puts particular focus on anonymity, freedom of expression and collaboration, openness and uses only free software. One of the solutions comprising the GNUnet stack is the GNU Name Server (GNS), ”*A Censorship-Resistant, Privacy-Enhancing and Fully Decentralized Name System*” [31]. Being fully decentralized, the network topology in GNS is a mash network in contrast to the tree behind DNS and DNSSEC. In this network there is no root node nor central authorities, and uniqueness of names is only local. This means that each node manages its own root, defining its own *petnames* and eventually delegating subdomains to connected peers. In order to grant privacy both of association and of communication, nodes connect to each other by autonomous linking and the exchange of their public key in order to encrypt every communication. This makes for a really short TCB, fully in control of the node owner. Each node publishes some of their managed records in a DHT, signing them with their key, thus preventing any

tampering of the resolution records by any adversary party, being MITM or a censorship authority. Provider nodes might adopt GNS in order to manage and expose the certified Nodes belonging to the messaging community as well as their managed Domain Names. SAE would then rely on Provider's GNS in order to identify Nodes and related Domain Names. It's important to note that registration on Provider's GNS is not handled by this technology, thus still a certain amount of offline operations need to be performed by Nodes managers.

4.3. 2-Tier Super Peer Architecture

The third solution we'd like to take in consideration is a 2-Tier Super Peer Architecture as described in [32]. In the paper is designed a security framework consisting of an ID authentication server, super peers and normal peers based on self generated private/public key pairs. The potential MITM attack in the public key distribution process is avoided with the pre-authentication of the peers on the ID authentication server. Each peer willing to join the network sends a join request to the ID authentication server publishing its public key and receiving a list of available super peers. The peer then contacts one super peer in order to actually join the community, publishing its relevant information to the super peer. Super peers can verify the peers by asking the ID authentication server for their public key. The super peers are thus in charge of maintaining a registry of the peers that constitute the community. They put this information in a DHT shared by super peers, that can be queried by peers. The proposed file-sharing application described in [32] can be easily translated as a distributed Name Resolution service to use for the secure resolution of destination servers. In this way each peer can globally query for information on the current nodes just by contacting a super peer in a secure way. Certificates are self-signed and programmatically exchanged, reducing the operation overhead.

In SAE, each Node would need to: 1) authenticate to our ID authentication server with our credentials 2) present our its current public key 3) contact one of the Provider nodes presented by authentication server (along with their public keys) in order be registered in the DHT along with its managed Domain Names. A SAE node that needs to send a message would then: 1) query the DHT in order to resolve the Node responsible for the destination Domain Name 2) obtain the destination Node public key.

4.4. Security considerations and selected approach

The network layer underlying SAE is insecure, thus SAE needs to incorporate an additional level of security. The 2 main requirements to SAE security are: 1) Identification of certified Nodes 2) Identification of the Node owner of SAE Domain Names. In order to allow for SAE faster adoption and easier / more secure management, we try to introduce a security layer on top of the underlying existing technologies. We described 3 different approaches to implement SAE security layer: 1) Naming Convention 2) GNS 3) a 2-Tier Super Peer Architecture. While the Naming Convention simplicity would allow for easy adoption, it would restrict naming choices while not completely addressing our security needs. GNS would be a technically viable solution, but in order to exempt Node managers from managing their own GNS, they would delegate all information to Providers offline. We think that the 2-Tier Super Peers architecture seems to fully address our needs: information management is secure and programmatic, interdependence of manual / offline operation is reduced to its minimum thanks to the use of authenticated self-signed certificates exchange.

5. Conclusion

In this paper we investigate the application of security in the Internet email architecture to bring advantages waiting for the DNSSEC large-scale adoption. Our model shows how it is possible to use the Internet email architecture as a transport layer for new valued-added applications. The proposed Secure Application Email model avoids two kinds of attacks: TLS downgrade attack and the MX

record misleading. Thus, the standard email service is enhanced by the introduction of the SAE systems, benefiting from an increased degree of security.

Every solution that aims to identify a trusted community relies on authentication mechanisms and trusting mechanisms. Some of the major challenges in adopting these solutions are technical effort and scalability. Furthermore, any new solution should be fully compliant with current standards to be successful in a short time. We have identified a solution where the technical effort is concentrated only on the service providers and which is scalable. The solution is fully compatible with the current Internet email system, without negative consequences from the insecurity of the DNS.

6. References

[1] ETSI EN 319 532-4 V1.2.1 (2022-05) Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 4: Interoperability profiles

[2] Crocker, D. (2009). RFC 5598: Internet Mail Architecture. *IETF Trust*.

[3] Gellens, R., & Klensin, J. (2011). RFC 6409: Message Submission for Mail.

[4] Hoffman, P. (2002). RFC3207: SMTP Service Extension for Secure SMTP over Transport Layer Security.

[5] Crocker, D., Hansen, T., & Kucherawy, M. (Eds.). (2011). RFC 6376: Domainkeys identified mail (DKIM) signatures.

[6] Kucherawy, M., & Zwicky, E. (Eds.). (2015). RFC 7489: Domain-based message authentication, reporting, and conformance (DMARC).

[7] Kitterman, S. (2014). RFC 7208: Sender policy framework (SPF) for authorizing use of domains in email, version 1.

[8] Dukhovni, V., & Hardaker, W. (2015). RFC 7671: The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance.

[9] Margolis, D., Risher, M., Ramakrishnan, B., Brotman, A., & Jones, J. (2018). RFC 8461: SMTP MTA Strict Transport Security (MTA-STS).

[10] Granström, M. (2009). *What is wrong with DNS?*. TKK Technical Reports in Computer Science and Engineering, Helsinki University of Technology. Retrieved on June 12, 2011, from http://cse. tkk. fi/en/publications/B/5/papers/Granstrom_final. Pdf.

[11] Arindam Bhattacharya. (2023). The role of DNS security in mitigating cyber threats: An analysis of recent attacks and recommended strategies. Advocacy Unified Network. https://doi.org/10.57939/WQK7-P542

[12] Yang, H., Osterweil, E., Massey, D., Lu, S., & Zhang, L. (2010). Deploying cryptography in Internet-scale systems: A case study on DNSSEC. *IEEE Transactions on Dependable and Secure Computing*, *8*(5), 656-669.

[13] B. Holst-Christensen and E. Frøkjær, "Security Issues in SMTP-based Email Systems," 2021 14th CMI International Conference - Critical ICT Infrastructures and Platforms (CMI), Copenhagen, Denmark, 2021, pp. 1-6, doi: 10.1109/CMI53512.2021.9663741.

[14] G. Kambourakis, G. D. Gil and I. Sanchez, "What Email Servers Can Tell to Johnny: An Empirical Study of Provider-to-Provider Email Security," in IEEE Access, vol. 8, pp. 130066-130081, 2020, doi: 10.1109/ACCESS.2020.3009122.

[15] Yao, Y., He, L., & Xiong, G. (2013). Security and cost analyses of DNSSEC protocol. In *Trustworthy Computing and Services: International Conference, ISCTCS 2012, Beijing, China, May 28–June 2, 2012, Revised Selected Papers* (pp. 429-435). Springer Berlin Heidelberg.

[16] Osterweil, E., Massey, D., & Zhang, L. (2009, December). Deploying and monitoring dns security (dnssec). In *2009 Annual Computer Security Applications Conference* (pp. 429-438). IEEE.

[17] Lampson, B., Abadi, M., Burrows, M., & Wobber, E. (1992). Authentication in distributed systems: Theory and practice. *ACM Transactions on Computer Systems (TOCS)*, *10*(4), 265-310.

[18] Nebuchadnezzar, H. (2012). The collateral damage of internet censorship by dns injection. *ACM SIGCOMM CCR*, *42*(3), 10-1145.

[19] Fujitsu Research Institute. (2010). Personal data in the cloud: A global survey of consumer attitudes. http://www.fujitsu.com/downloads/SOL/fai/reports/fujitsu_personal-data-in-the-cloud.pdf
[20] Margolis, D., Risher, M., Ramakrishnan, B., Brotman, A., Jones, J. (2018). RFC 8461: SMTP MTA Strict Transport Security (MTA-STS)
[21] Durumeric, Z., Adrian, D., Mirian, A., Kasten, J., Bursztein, E., Lidzborski, N., ... & Halderman, J. A. (2015, October). Neither snow nor rain nor MITM... an empirical analysis of email delivery security. In Proceedings of the 2015 Internet Measurement Conference (pp. 27-39).
[22] Petrucci, C., Gennai, F., Shahin, A., and A. Vinciarelli, "La Posta Elettronica Certificata - Italian Certified Electronic Mail", RFC 6109, DOI 10.17487/RFC6109, April 2011
[23] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., & Thayer, R. (2007). RFC 4880: OpenPGP message format.
[24] Schaad, J., Ramsdell, B., & Turner, S. (2019). RFC 8551: Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification.
[25] D. K. Gillmor, D. K., Hoeneisen B., Melnikov A. (2023). Header Protection for S/MIME. draft-ietf-lamps-header-protection-11
[26] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <https://www.rfc-editor.org/info/rfc5321>.
[27] Gennai F., Sinibaldi F., Buzzi M.; Martusciello L. (2023). A secure application email model. ISTI Technical Report, ISTI-2023-TR/002
[28] Nopanen, J. (2022). Enforcing SMTP encryption: Research on current DANE and MTA-STS implementations in Finland.
[29] Babrahem, A. , Alharbi, E. , Alshiky, A. , Alqurashi, S. and Kar, J. (2015) Study of the Security Enhancements in Various E-Mail Systems. Journal of Information Security, 6, 1-11. doi: 10.4236/jis.2015.61001
[30] Grothoff, C. (2017). *The gnunet system* (Doctoral dissertation, Université de Rennes 1)
[31] Wachs, M., Schanzenbach, M., & Grothoff, C. (2014). A censorship-resistant, privacy-enhancing and fully decentralized name system. In *Cryptology and Network Security: 13th International Conference, CANS 2014, Heraklion, Crete, Greece, October 22-24, 2014. Proceedings 13* (pp. 127-142). Springer International Publishing.
[32] Kwon, H., Kim, S., Nah, J., & Jang, J. (2007, June). Public key management framework for two-tier super peer architecture. In *27th International Conference on Distributed Computing Systems Workshops (ICDCSW'07)* (pp. 72-72). IEEE.