FakeBehalf: Imperceptible Email Spoofing Attacks against the Delegation Mechanism in Email Systems

Jinrui Ma[†], Lutong Chen[†], Kaiping Xue[†,*], Bo Luo[§], Xuanbo Huang[†], Mingrui Ai[†],
Huanjie Zhang[†], David S.L. Wei[‡], Yan Zhuang[†]
[†] *University of Science and Technology of China,*
{mjr2000,lutong98,hxb777,amr2016}@mail.ustc.edu.cn; {kpxue,james,zy518}@ustc.edu.cn
[§] *The University of Kansas, bluo@ku.edu*
[‡] *Fordham University, wei@cis.fordham.edu*

Abstract

Email has become an essential service for global communication. In email protocols, a Delegation Mechanism allows emails to be sent by other entities *on behalf of* the email author. Specifically, the `Sender` field indicates the agent for email delivery (*i.e.*, the Delegate). Despite well-implemented security extensions (*e.g.*, DKIM, DMARC) that validate the authenticity of email authors, vulnerabilities in the Delegation Mechanism can still be exploited to bypass these security measures with well-crafted spoofing emails.

This paper systematically analyzes the security vulnerabilities within the Delegation Mechanism. Due to the absence of validation for the `Sender` field, adversaries can arbitrarily fabricate this field, thus spoofing the Delegate presented to email recipients. Our observations reveal that emails with a spoofed `Sender` field can pass authentications and reach the inboxes of all target providers. We also conduct a user study with 50 participants to assess the recipients' comprehension of spoofed Delegates, finding that 50% are susceptible to deceiving Delegate information. Furthermore, we propose novel email spoofing attacks where adversaries can impersonate arbitrary entities as email authors to craft highly deceptive emails while passing security extensions. We assess their impact across 16 service providers and 20 clients, observing that half of the providers and all clients are vulnerable to the discovered attacks. To mitigate the threats within the Delegation Mechanism, we propose a validation scheme to verify the authenticity of the `Sender` field, along with design suggestions to enhance the security of email clients.

1 Introduction

Email stands as a significant tool for global communication. In 2023, approximately 347.3 billion emails were generated and transmitted each day, averaging over 4 million emails per second [45]. Unfortunately, due to its convenience and the sensitive information it contains, email has become a preferred tool for network phishing attacks. According to Cofense's annual report [4], malicious email threats bypassing secure email gateways (SEGs) increased by 104.5% in 2023, and email continues to rank first among threat vectors for cybercrime, with 90% of data breaches starting with phish. Consequently, the persistence of email threats poses a formidable challenge to network security.

Among various email threats, email spoofing attacks are particularly concerning. Attackers employ various tactics to impersonate legitimate entities, sending fraudulent emails to deceive recipients and gain their trust. To counter these spoofing emails, researchers have proposed various defensive methodologies. As highlighted in numerous works [3, 11, 36, 47], the email header, which includes vital fields, plays a significant role in assessing the legitimacy of an email. The `From` field, designating the **message's author**, *i.e.,* the individual responsible for composing the email, holds particular significance. Due to its importance, the `From` field is often the primary target of email spoofing attempts, where attackers aim to forge this field in fraudulent emails. Consequently, it is also the central focus of email authentication schemes and protocols [3, 6, 17, 36].

Despite the prevalence of email security mechanisms such as DKIM [6] and DMARC [17] for verifying the authenticity of the `From` field, we identify novel vulnerabilities that can still be exploited in email spoofing attacks. These vulnerabilities arise within what we term *the Delegation Mechanism*, a widely employed strategy that allows individuals to send emails on behalf of the actual author. RFC 5322 [33] specifies the `Sender` field for recording the agent responsible for email delivery (*i.e.*, the Delegate). This field is crucial in the Delegation Mechanism as it denotes the Delegate if different from the email author. However, our observation reveals two significant vulnerabilities: **Vul-1: Sender field Fabrication** - the `Sender` field lacks authentication and can be arbitrarily fabricated by attackers, and **Vul-2: Inconsistent Delegation** - there is no standard protocol for implementing the Delegation Mechanism, resulting in inconsistent implementations across web interfaces and email clients. Attackers exploit these weak-

*Corresponding author: Kaiping Xue, kpxue@ustc.edu.cn

nesses to pass authentications and dispatch spoofing emails to potential victims. Our research introduces a novel email spoofing attack that allows attackers to impersonate legitimate entities, send emails, and evade security checks. By exploiting issues within the Delegation Mechanism, these spoofing emails can reach inboxes of popular service providers without triggering security warnings, making recipients unaware of the spoofing attacks.

This study delves into email authentication issues related to the Delegation Mechanism, with a specific focus on the `Sender` field. While the `From` field undergoes strict verification, the `Sender` field lacks authentication and can be arbitrarily generated by email servers, which raises significant security concerns. Based on the observation above, the primary objective of this paper is to investigate and answer three key research questions:

- **RQ1**: How do mainstream email providers and clients implement the Delegation Mechanism and handle emails with spoofed `Sender` fields?
- **RQ2**: How do mainstream service providers and clients convey the inconsistency between the email author and Delegate to the user? Can users correctly comprehend this inconsistency, especially during spoofing attacks?
- **RQ3**: How can an adversary exploit vulnerabilities within the Delegation Mechanism to craft practical attacks? What measures can mitigate these risks and defend against such attacks?

We conduct comprehensive studies to answer these questions. Firstly, we systematically evaluate the implementations of the Delegation Mechanism across 16 email services and 20 clients, focusing on the effect of the `Sender` field within this mechanism. Our findings reveal that 10 providers have various implementations for the mechanism, while 13 clients solely use the `Sender` field as the Delegate. To understand recipients' sensitivity to such spoofed email Delegates, we conduct a user study involving 50 participants to observe their reactions when encountering emails with various Delegates. Our analysis indicates that deceptive Delegate information misled 50% of the participants into misjudging spoofed emails as legitimate. To explore more effective methods of email spoofing, we propose six distinct email spoofing methods involving the `Sender` field, enabling attackers to impersonate anyone and send spoofed emails to unsuspecting victims without any warnings exposing the attackers' addresses. We also evaluate the efficacy of these attacks across 16 prominent email providers and 20 clients and find half of the providers and all clients vulnerable to these attacks. Finally, we propose a validation scheme for the `Sender` field to enhance the security of email systems.

This paper presents three main contributions:

- We identify the vulnerabilities within the Delegation Mechanism for the first time and highlight the insufficient validation of the `Sender` field by most email service providers.

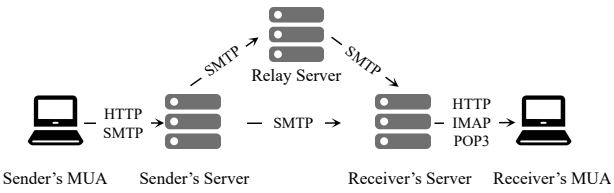


Figure 1: Email Delivery Process.

- We conduct a comprehensive user study involving 50 participants to evaluate the effectiveness of forged Delegates on email users, with half of the participants susceptible to misleading email delegate information.
- We propose six distinct email spoofing attacks that exploit the vulnerability within the Delegation Mechanism, allowing the forged `Sender` field to appear as the genuine email author without any security warnings. We also evaluate the efficacy of these attacks across 16 prominent email providers and 20 clients, finding that half of the providers and all clients are vulnerable to these attacks.

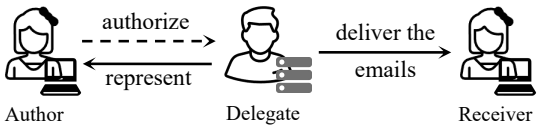
This paper is structured as follows: Section 3 provides an overview of our work. In Section 4 and Section 5, we illustrate the implementations of the Delegation Mechanism and the effectiveness of forged Delegates on recipients, respectively. Section 6 introduces the methodologies of email spoofing attacks. Section 7 discusses our main observations, and Section 8 illustrates the defensive measures. Section 9 introduces related works and their differences from ours. Finally, we draw our conclusion in Section 10.

2 Background

2.1 Mail Delivery and Delegation Mechanism

The Simple Mail Transfer Protocol (SMTP) [16] serves as the fundamental protocol for email transmission. An SMTP session begins with the sender initiating the session through the `HELO` command, followed by the `MAILFROM` command, indicating the initiator of the session, primarily verified by SPF authentication [15]. Upon receiving a valid response from the receiver, the sending server transmits the email using the `DATA` command. The entire email delivery process is presented in Figure 1. Initially, the author generates an email using a Mail User Agent (MUA), which may either be a web interface or an email client. Subsequently, the email is transmitted to the transport server via SMTP or HTTP. The email then traverses several forwarding servers before reaching the recipient's server, which delivers the email to the recipient's mailbox through IMAP [24], POP3 [27], or HTTP.

With the development of the modern Internet, major enterprises worldwide have adopted specialized mail servers to centrally process their emails. In such cases, emails from various origins are assembled in these dedicated agents and processed and transmitted collectively. Similarly, email au-



- The **From** field denotes the **Email Author;**
- The **Sender** field denotes the **Delegate;**
- The **Delegation Mechanism:**
  - The **Delegate on behalf of** the **Author** to send emails.

Figure 2: The Delegation Mechanism and Related Fields.

thors may authorize other individuals to represent them in dispatching emails, a scenario referred to as the Delegation Mechanism in email systems (see Figure 2). This mechanism presents a challenge when the email author is inconsistent with the Delegate. Meanwhile, it is necessary to record the address of the delivery agent in emails for possible bouncebacks, along with the original email author. To address this problem, RFC 5322 specifies two fields in the email header: `From` and `Sender`, as shown in Figure 2. The `From` field denotes the email author and is essential in an email. The `Sender` field is an optional field that indicates the agent for delivering the email. When the specialized agent delivers emails from other entities, it must use the `Sender` field to record its own address as the Delegate [33]. The `Sender` field is primarily designed for the Delegation Mechanism and will be ignored if it aligns with the `From` field.

2.2 Email Spoofing and Countermeasures

2.2.1 Email Spoofing Attacks

Email spoofing attacks can be utilized in phishing attempts, aiming to exploit inherent vulnerabilities in email systems and impersonate others to send spoofed emails. Attackers craft emails that resemble those from legitimate entities, deceiving and gaining the trust of potential victims. This method is more elaborate and effective than traditional phishing emails, as spoofed emails closely mimic legitimate ones. Previous research has developed various spoofing methods to bypass authentication and successfully attack popular email service providers and clients [3, 36], highlighting the severe threat of these attacks.

2.2.2 Email Authentication protocols

The original SMTP protocol lacks authentication for the email author, allowing any user on the Internet to impersonate others and send emails. To address these security issues, several extensions have been developed to enhance email transmission security, with the three most widely used being SPF [15], DKIM [6], and DMARC [17].

SPF. The Sender Policy Framework (SPF) [15] allows domain owners to publish a DNS TXT record that specifies which servers are authorized to send emails on behalf of their domain. Upon receiving an email, the receiving server extracts the domain from the `MAILFROM` or `HELO` command, queries the domain's SPF record via DNS, and compares the sending server's IP address with the SPF record. This process enables the receiver to authenticate the email's validity and handle it based on local policies.

DKIM. DomainKeys Identified Mail (DKIM) [6] employs cryptography to ensure the integrity protection of emails. It enables the email author to choose specific header fields and use a private key to sign them, along with the hash of the email body. Receivers can extract the "selector" entry and the signer's domain from the signature in the email header to query the public key via DNS and then validate the authenticity of the DKIM signatures.

DMARC. Domain-based Message Authentication, Reporting and Conformance (DMARC) [17] is an email security extension built upon SPF and DKIM. DMARC introduces an alignment scheme to ensure that the address in the `From` field aligns with the domain verified by SPF or DKIM. If both SPF and DKIM fail to pass, DMARC authentication fails as well. DMARC also includes a specialized feedback mechanism that allows receivers to report errors or provide suggestions to help develop the domain's DMARC policies.

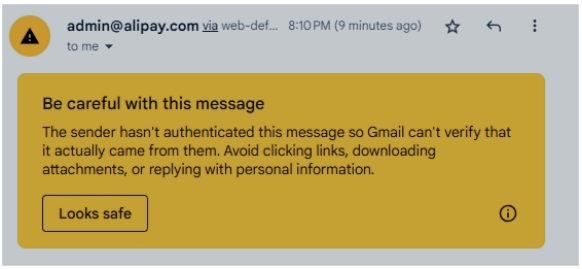
Combining the above security protocols, an email system ensures that spoofing emails based on the `From` field are easy to verify. However, there is currently no validation method for the `Sender` field, leaving open the possibility of successful spoofing attacks on the `Sender` field.

2.2.3 Client-level Protections

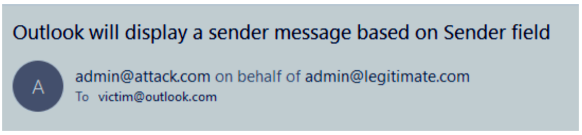
Despite the implementation of security protocols, engineers and researchers have proposed various measures to enhance user-side security. Two commonly used mechanisms are user warnings, which alert users to potentially malicious emails or activities, and exposing the Delegate, which provides information about the actual origin of the email. These mechanisms are extensively employed in email clients and web interfaces.

GUI Warning. Researchers have studied user actions [11, 30, 47] and found that the user's judgment is crucial in email authentication. Due to differences in education and technical backgrounds, recipients may make different decisions when faced with the same email. To assist in verifying suspicious emails, user warnings are vital and widely implemented in email clients and web interfaces. When an email is flagged as phishing or potentially harmful but not directly rejected, a warning prompts the recipient to examine the email carefully, as shown in Figure 3(a).

Exposing the Delegate. As illustrated above, an email contains two distinct "senders": one specified in the `MAILFROM` command within the envelope, and the other in the `From` field of the email header. Since the original transmission protocol does not allow for inconsistency between these two addresses, attackers can exploit this vulnerability to arbitrarily forge the



(a) An Example of User Warning that Gmail.com Displays to the Recipient.



(b) The Outlook Client Displays a Delegate Message Based on the Sender Field.

Figure 3: Two Client-level Protections Employed Worldwide.

`From` field with a controlled domain. In such cases, most email providers will expose the actual transmitter (also referred to as the Delegate) to users, which is defined as a Sender Inconsistency Check (SIC) by Shen *et al.* [36]. Exposing the Delegate is also a defensive measure to prevent email spoofing attacks, as it presents the email's origins to the recipients. As the envelope varies with every SMTP session, RFC 5322 [33] specifies the `Sender` field to record the Delegate's address, as shown in Figure 3(b). However, the absence of authentication for the `Sender` field enables attackers to arbitrarily fabricate this field. This design ambiguity gives rise to additional security issues. Moreover, the lack of a standard protocol for implementing the Delegation Mechanism leads to inconsistent implementations among various providers and clients, raising potential risks.

3 Research Overview

In this work, we primarily investigate the vulnerabilities associated with the email Author and the Delegate within the Delegation Mechanism.

3.1 The Attack Model

The threat model, as depicted in Figure 4, includes three key entities: a trusted email author (Alice) with the mailbox *Alice@legitimate.com*, a victim receiver (Bob) with the account *Bob@victim.com*, and an adversary (Eve) attempting to impersonate Alice and send forged emails to Bob from the domain *@attack.com*. In this model, Alice deploys robust security extensions for the sending domain (SPF, DKIM, and DMARC), as does Eve. It is worth noting that Alice is not influenced by Eve when configuring the security policies and is unaware of these attacks. We also clarify that these addresses presented in

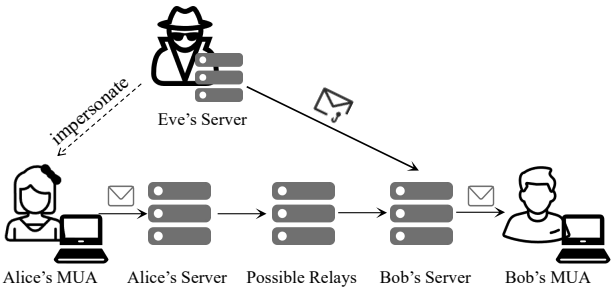


Figure 4: The Model of Email Spoofing Attack.

the model are not actual email addresses but serve as illustrative examples, as well as the attacks mentioned in Section 6. We assume that Eve owns a personal email server and can establish SMTP sessions directly with Bob's receiving server, a task that is easily achieved by attackers [3, 36]. When establishing SMTP sessions, Eve will try to manipulate the email content within the `DATA` command rather than the commands in SMTP interactions, such as `MAILFROM`. This implies that the address specified in the `MAILFROM` command is under the control of the attacker. With this attack model, we can naturally pass the authentication of SPF and DKIM, as the sending domain is fully under our control. The primary factor enabling this is the absence of explicit exposure of the authentication results to users, who are unaware that the verified domain is not aligned with original expectations.

Moreover, the success of an attack in Section 6 is defined by three factors: 1) The forged email successfully enters Bob's inbox; 2) The MUA presents Alice (as indicated in the `Sender` field) as the email author; 3) The attack email closely resembles a legitimate message from Alice, e.g., the MUA displays no warnings or Delegate information exposing the attacker's address. Each attack is repeated, meaning the same attack email is sent twice, and it is considered successful only if both attempts succeed. A demonstration of a successful attack on the Gmail client is presented in Figure 9(a) in the Appendix, while a failed attempt on the Outlook client is shown in Figure 9(b) as a contrast.

3.2 Solution Overview

To address the preceding research questions, we conduct a series of studies comprising three main steps. An overview of our methodology is presented in Figure 5.

We commence with a *Measurement Study* to investigate the implementations of the Delegate Mechanism and the handling of spoofed `Sender` fields in target providers and clients (**Step-1** of Figure 5). We append a spoofed `Sender` field to legitimate emails originating from our domain and dispatch them to target email providers. By checking the email performances on 16 web interfaces and 20 email clients, we discover that most providers lack authentication for the `Sender` field and accept test emails in their inboxes. Moreover, 10 out of 16

providers have various implementations of the Delegation Mechanism, while 13 clients solely display the `Sender` field as the Delegate, as introduced in Section 4.

To evaluate recipients' comprehension of the spoofed Delegate, we conduct a *User Study* involving 50 participants (**Step-2** in Figure 5), as described in Section 5. We carefully craft four types of email Delegates for identical emails and invite each participant to independently assess the validity of one example from the four emails. The observations reveal that misleading Delegates successfully deceive 50% of the participants into recognizing spoofing emails as legitimate, while the remaining half can verify these suspicious emails.

While spoofing email Delegates presents substantial security risks, this simple attack is still identified by 50% of the participants. To achieve more effective spoofing, we employ methodologies that forge the `From` field along with exploiting vulnerabilities within the Delegation Mechanism. Consequently, we propose six email spoofing techniques in **Step-3** that enable attackers to impersonate legitimate entities and send spoofing emails to potential victims without displaying security warnings to expose the attacker's address. Detailed methods are described in Section 6. Our investigations reveal that these attacks successfully deceive half of the providers and all the clients.

Finally, in Section 8, we propose a validation scheme for the `Sender` field along with several security suggestions for email clients and users.

Ethical Considerations and Responsible Disclosure. We have taken several measures to ensure the ethical integrity of our research. One potential risk is the trust credit loss that legitimate domains might suffer if victims receive spoofed emails under those domains. However, our email spoofing targets in the *Spoofing Attack* are fully under our control. We sent only two example emails with addresses beyond our control to demonstrate the concept of our attack (Fig 3(a) and Fig 9). We believe this limited number of emails will not trigger robust defense mechanisms in commercial email systems. Additionally, we reported our findings to relevant email providers, along with our test emails, to help them investigate the vulnerability. In the *Measurement Study* and *Spoofing Attack*, we exclusively used test accounts owned by ourselves, ensuring that no real users were affected by these experiments. We carefully controlled the email-sending rate to be under 10-minute intervals per email to minimize the impact on target servers. We did not request any personal information from participants in our *User Study*. The user study in Section 5 was reviewed by the IRB at the university and received an exempt designation. Our platform implements robust defensive measures to prevent unauthorized access or data leakage. We have responsibly disclosed our findings to all affected email providers in January 2024. As of the submission of this manuscript, we have received feedback from *163.com*, *139.com*, and *coremail.com* that they successfully replicated the attacks and their engineering teams are further investigating the issue. *Evolution* is currently in discussion with us to identify the affected versions. Our further investigation showed that the recently updated version of Evolution is still vulnerable. To prevent malicious attacks by unscrupulous users, we will not disclose the vulnerabilities to the public until 90 days after our final disclosure.

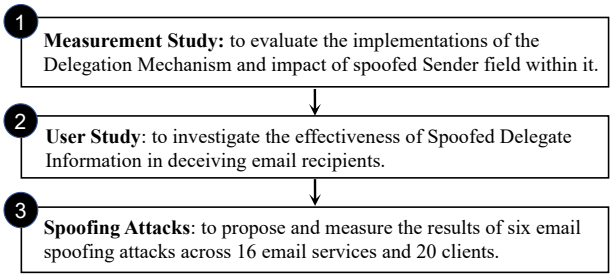


Figure 5: An Overview of Research Methodology.

4 Measurement Study on Implementations of the Delegate Mechanism

Since we observe that the `Sender` field can be modified by attackers, our goal is to explore the implementations of the Delegation Mechanism and how the `Sender` field is presented within this mechanism across widely-used email providers and clients. We conduct a *Measurement Study* involving 16 email providers and 20 clients to examine the performance of emails with a spoofed `Sender` field. The results are presented in Section 4.2.

4.1 Experiment Methodology

Target selection. This study primarily focuses on assessing the impact of the spoofed `Sender` field within the Delegation Mechanism across various email providers and clients. The selection of target providers is based on three specific criteria. Firstly, we choose widely used providers such as *gmail.com* and *outlook.com* since their security concerns and policies affect a larger global user base. Secondly, our study requires an end-to-end experiment involving gathering information from recipients and necessitating direct SMTP connections with target servers, excluding providers like *protonmail.com*, which does not offer SMTP. Thirdly, we need access to the raw email content or email files from testing accounts for further analysis. Consequently, we select a total of 16 email service providers, including platforms like *163.com* and *gmail.com*. Additionally, as claimed by Liu *et al.* [22], the market share of self-hosted email servers has decreased consistently since 2017, with many organizations opting to deploy their email servers with commercial support such as Outlook. We also evaluate the self-hosted email server of a large university supported by *coremail.com*, a popular email server solution deployed on more than 15,000 private servers [5].

Table 1: Target Providers' Implementations of the Delegation Mechanism and the Handling of the Spoofed Sender field.

Email Handling	Gmail	Outlook	Zoho	Naver	139	163	Yandex	QQ	Mailo	Yeah.net	126	Sohu	Sina	Rambler	Coremail.com	Tutanota
Accept in inbox	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Modify the spoofed Sender field						✓				✓	✓				✓	
Append a Sender field if necessary						✓		✓		✓	✓				✓	
Policy to expose the Delegate	✓		✓		✓	✓		✓		✓	✓	✓	✓		✓	
Header fields used as the Delegate	Return-Path		Return-Path		X-RM-Sender	Sender		Sender		Sender	Sender	Return-Path	X-Sender		Sender	

Furthermore, we meticulously select popular email clients across five desktop and mobile operating systems, including Foxmail and Apple Mail. In total, we include 20 email clients that collectively serve billions of users worldwide [28]. We believe that the security issues they exhibit are representative of the threats faced by most email users. Refer to Table 1 and Table 2 for more details on our chosen targets.

Measurement procedure. We initially set up a personal email server to establish direct interactions with the servers of target providers. Additionally, we employ robust security extensions (SPF, DKIM, and DMARC) to ensure that legitimate emails sent from our domain successfully pass authentication. Next, we generate test emails originating from our domain, with the `From` field aligning with the `MAILFROM` command, and append a spoofed `Sender` field in the email header (*e.g.*, *admin@google.com*). Subsequently, we configure all selected clients as agents for our email account to assess email performance. Finally, we deliver these crafted emails to target providers and observe their manifestations on the web interfaces and clients. Through careful analysis of the results, we summarize our findings in Section 4.2.

4.2 Experiment Results

More details of the experimental results from target providers and clients are presented in Table 1 and Table 2, respectively. Based on these results, we make three main observations as follows:

Deficient validation of the Sender field by email providers. Our experimental findings reveal that all 16 target providers allow test emails to pass their authentications and reach the recipients' inboxes without displaying any user warnings, as shown in Table 1. Notably, the `Sender` field is spoofed and does not match the address in the `From` field or `MAILFROM` command, demonstrating that attackers can forge this field with accounts of any legitimate entities. Although a few providers perform various processing actions on this field, we conclude that the majority of providers do not authenticate the validity of the `Sender` field.

Modification of the spoofed Sender field by certain providers. Upon comparing the received emails with the original ones, we observe that five of the 16 providers modify the `Sender` field in various situations based on their local strategies. When processing "legitimate" emails with a spoofed `Sender` field only, *126.com*, *163.com*, and *yeah.net* delete the existing `Sender` field, while *coremail.com* modifies it to the address in `MAILFROM`. Given our experimental setup, both modification methods seem reasonable. Moreover, when processing various spoofing attack emails mentioned in Section 6.1, these four providers modify the spoofed `Sender` field to the address in the `MAILFROM` command, which is an effective measure to defend against these attacks. Additionally, these four services, along with *qq.com*, append a `Sender` field in original emails if necessary (and none exists), as shown in Table 1. The modification of the spoofed `Sender` field reveals that these email providers employ basic validation measures to verify this field. However, these countermeasures are not extensively employed, as the remaining 11 providers take no action toward the spoofed `Sender` field. On the other hand, the issue within the Delegation Mechanism is a system-level problem relevant to both email providers and clients. Even for providers that modify the spoofed `Sender` field, it is still possible to conduct spoofing attacks with popular email clients.

Inconsistent implementations of the Delegation Mechanism within web interfaces and clients. Our findings indicate that 13 out of 20 clients and 10 out of 16 providers have employed the strategy to expose the Delegate to the recipients, as shown in Table 1 and Table 2, respectively. Specifically, 10 providers deploy various implementations of the Delegation Mechanism, with half of them parsing the `Sender` field for this purpose. While *outlook.com* does not present the Delegate on the web interface, it stores and labels the address in `Sender` as the email transmitter. Conversely, all 13 clients present the `Sender` field as the Delegate, including popular ones like Outlook and eM Client.

According to the results, the `Sender` field serves as a primary source for email Delegates in 50% of web interfaces and is the sole source in email clients. Considering that four providers modify the spoofed `Sender` field, spoofing the Delegate is practical only on the web interface of *qq.com*. However, since the remaining 11 providers keep the spoofed `Sender` field unchanged, we conclude that attackers can forge Delegate information on all 13 clients.

5 User Study

Motivated by the observations above, we aim to investigate whether users comprehend the inconsistency between the

Table 2: Attack Results on Email Clients.

OS	Clients	Version	Exposing Delegate	Success Attack Types
Windows	Outlook	16.0.14332.20637	✓	A_1[1], A_3, A_4
	eM Client	9.2.2157	✓	A_1, A_3
	Win-Email	16005.14326.21904.0	✓	A_1, A_2, A_3, A_5, A_6
	Foxmail	7.2.25.245	✓	A_1, A_3, A_5, A_6
Linux	Thunderbird	115.7.0-1		A_2, A_3, A_6
	Evolution	3.50.0-1		A_3, A_6
	Mailspring	1.13.3		A_1, A_2, A_3, A_4
MacOS	Outlook	16.78.*	✓	A_1, A_2, A_5, A_6
	Apple Mail	Mac 14 (23B74)		A_6
	Foxmail	1.5.5	✓	A_1, A_3
	eM Client	9.2.2144.0	✓	A_1, A_3
iOS	Gmail	6.0.231127		A_1, A_2, A_3
	Apple Mail	iOS 17.1		A_1, A_3, A_5, A_6
	Outlook	4.2347.1	✓	A_1, A_2, A_3, A_5, A_6
	Netease	7.18.1	✓	A_1, A_2, A_3, A_4, A_5, A_6
	QQ	6.5.0	✓	A_1, A_3, A_6
Android	Gmail	2024.02.04.604829058		A_1, A_3, A_4
	Outlook	4.2347.4	✓	A_1, A_2, A_3, A_5, A_6
	Netease	7.18.4	✓	A_1, A_2, A_4, A_6
	QQ	6.5.1	✓	A_1, A_2, A_3, A_6

[1] The subscript A_i identifies the spoofing attacks capable of attacking these clients, which will be discussed in Section 6.1.

email author and the spoofed Delegate. To this end, we conduct a user study involving 50 participants. Analysis of the results reveals that **deceptive Delegate information notably influences 50% of the participants to verify spoofing emails as legitimate, while the other half can identify these emails as fraudulent**.

5.1 Methodology

We initially set up an email platform for our study and invite 50 participants to evaluate their perceptions when presented with emails containing forged Delegates. The participants, who are well-educated college students from different grades, have generally encountered various kinds of phishing emails and possess a certain level of recognition. They are asked to validate and decide how to process (accept or reject) five test emails and provide brief explanations for their judgments. To collect the participants' actions when judging emails, we design our web interface with additional functions beyond traditional MUAs. This setup also streamlines the process for participants to access the web interface, enabling us to better organize the study. Through understanding the participants' actions regarding the test emails, we summarize our key findings in Section 5.2.

Email platform. For better organization and privacy considerations, we initially develop an email platform for the study, consisting of four primary components: the storage module, the controller, the analyzing module, and the web interface. Each participant is allowed to participate in the test only once. The web interface displays test emails with various additional functions for the participants, such as validating the email or marking suspicious portions. Test emails are securely stored in the storage module along with anonymous IDs and hashed passwords. The controller retrieves measuring emails from the storage module, which are later parsed and presented on the web interface. We design our web interface in the style of *gmail.com*, as it remains the most popular email provider worldwide. As most MUAs do, if a `Sender` field is present in the test email, we show it to the participant as the email Delegate; in contrast, if this field is absent, no such information will be displayed, as depicted in Figure 6. During the test, the controller caches the actions performed by each participant and subsequently transmits them to the analyzing module. The analyzing module classifies these actions based on different orientations and provides statistical results for further analysis. This platform is fully under our control, with comprehensive measures to prevent security risks.

Email generation. We investigate the impact of forged email Delegates on users, and a crucial step is to generate emails with distinct `Sender` fields. As the recipients' domain is *victim.com*, we consider four types of Delegates: 1) a domain controlled by the attacker (*attack.com*), which is easily identifiable to recipients; 2) an account from the victim's domain (*e.g.*, *admin@victim.com*), which is most likely to deceive recipients, as it appears to originate from their organization; 3) a domain belonging to a different organization (*organization.com*), which has less potential to mislead recipients than the previous type; 4) no specific Delegate information, acting as a control group.

Each participant is requested to verify five test emails (Emails 1 to 5, as presented in Table 3), with Email 5 focusing on the `Sender` field. To minimize bias, each participant is randomly assigned one of the four types of emails as their final test. The other four emails (Emails 1 to 4), which include three legitimate and one phishing sample, are based on real samples received by our group to ensure participants' basic ability to recognize phishing. Additionally, all emails are desensitized for privacy preservation.

Protocol of the user study. We first inform the participants that our study is an anti-phishing game, where they play the role of an assistant helping a group leader process emails, a commonly used methodology in phishing studies [30, 44, 47]. After account registration with pseudonyms and logging into the web interface, each participant is requested to process five emails, with the final one featuring the spoofed `Sender` field. The web interface provides buttons to process the emails, and participants can also highlight any suspicious parts. Their actions are recorded and subsequently transmitted to the analyzing module for further examination.

After the study, we disclose the actual purpose of the research to the participants to enhance their ability to identify email spoofing. Our main findings are summarized in Table 3.

5.2 Results and Analysis

As shown in Table 3, the experimental results indicate that half of the 18 participants who received emails with deceptive Delegate information (*admin@victim.com*) mistakenly iden-

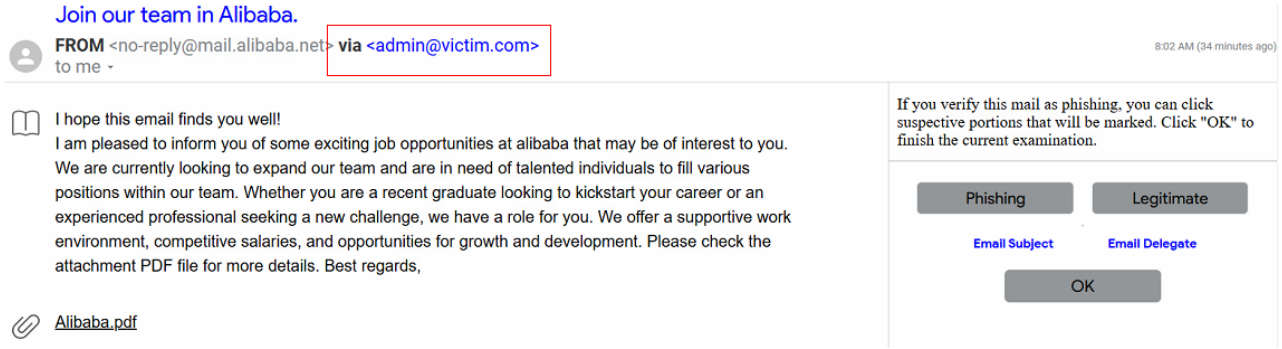


Figure 6: The Web Interface of the User Study, where the Sender Field is Displayed as the Delegate as Marked.

tified them as legitimate. Conversely, the misjudgment rate for identical emails without a spoofed `Sender` field is only 12.5%. For emails with less misleading Delegates (*organization.com*), three out of ten participants incorrectly judged them as legitimate, and only one recipient misjudged the phishing emails with the Delegate *attack.com*. We randomly assign test samples to each participant, resulting in slightly different participant counts across the four groups. However, this inconsistency does not significantly affect our findings, which are more relevant to the experimental setup.

The findings confirm that deceptive email Delegates significantly mislead users into recognizing forged emails as legitimate. Attackers can deceive many users by sending "legitimate" emails with spoofed `Sender` fields, which are accepted by all measured providers. However, this forgery is not always effective, as half of the users recognize the abnormal Delegates and correctly identify the emails as spoofing. Additionally, some participants noted that the deceptive nature of the Delegate itself raised suspicion, leading them to reject such emails.

6 Imperceptible Email Spoofing Attack

The results of the *User Study* reveal that deceptive email Delegates misled half of the participants into trusting spoofed emails, while the other half detected these forgeries. This prompts us to explore more aggressive email spoofing mechanisms. Inspired by previous works [3, 36], we examine various approaches to craft emails with irregular `From` fields and assess their effectiveness in evading security protocols. Consequently, we propose six types of email spoofing attacks and measure their impact across 16 email services and 20 clients. All 20 clients are configured as MUAs for all 16 providers via IMAP, resulting in 336 combinations (including 16 web interfaces of target providers). Detailed demonstrations of these attacks are described in Section 6.1. The experimental results indicate that these spoofing emails can successfully bypass authentication and enter the inboxes of eight target providers. Furthermore, by exploiting issues within the Delegation Mechanism, we demonstrate that all 20 clients display the unauthenticated `Sender` field as the sole email author without any warnings or Delegate information to expose the attacker's address, making these attacks imperceptible to the recipients.

Minimize the bias. This work primarily focuses on the issues within the Delegation Mechanism, which differs from spam filters mainly based on the email body semantics. To minimize the impact of spam filters, we consult an email security expert to create an email body unlikely to be flagged as spam or harmful. Additionally, we use a static IP for our email server and control the sending rate with a 10-minute interval to reduce the impact on target email servers. Considering the billions of daily emails, we believe a low sending rate will not significantly affect target servers. We randomly select pre-sending emails and target providers and send each type of email twice to the same service. An attack is deemed successful only if both attempts succeed. By sending emails randomly, we aim to reduce the impact of prior emails on subsequent ones.

6.1 Email Spoofing with the Sender Field

This section introduces various approaches employed to pass authentications in our email spoofing attacks. Since the `Sender` field is a crucial part of the Delegation Mechanism, all forgery methods are denoted by **"with the Sender field"**. For conciseness, we omit the preceding quotes and use A_i to represent the i-th method. Additionally, to evaluate the `Sender` field's effect on email spoofing, we delete this field from test emails and conduct a comparative trial for each attack. These trials help assess the `Sender` field's impact on the success rate of our attacks.

Attack 1: Multiple addresses in one From field (A_1). RFC 5322 allows multiple addresses within one `From` field [33]. In such cases, the `Sender` field must be used to indicate the one responsible for email delivery. However, no security extensions authenticate multiple email authors, enabling attackers to fabricate addresses in `From` casually while utilizing the `Sender` field to enhance the spoofed account's reliability. Figure 7(a) shows the structure of the forged email. The address

Table 3: Experimental Results of the User Study.

Character	Email 1	Email 2	Email 3	Email 4	Email 5			
					Variant 1	Variant 2	Variant 3	Variant 4
Is Phishing	×	×	✓	×	✓	✓	✓	✓
The Link	✓	×	✓	✓	×	×	×	×
Attachment	✓	×	×	×	✓	✓	✓	✓
Sender Field	legitimate	×	×	×	Victim.com	Organization.com	Attack.com	×
Participant Count	50	50	50	50	18	10	14	8
Misjudgment[1]	10	12	13	5	9	3	1	1

[1] Misjudgment refers to the act of participants mistakenly judging phishing emails as legitimate, or identifying legitimate ones as phishing.

in the `Sender` field is consistent with the first one in the `From` field, which is the spoofed account that clients may regard as the primary author and show to the recipient. Instead, receiving servers employ the second address (controlled by attackers) for authentication and consequently accept this email as legitimate.

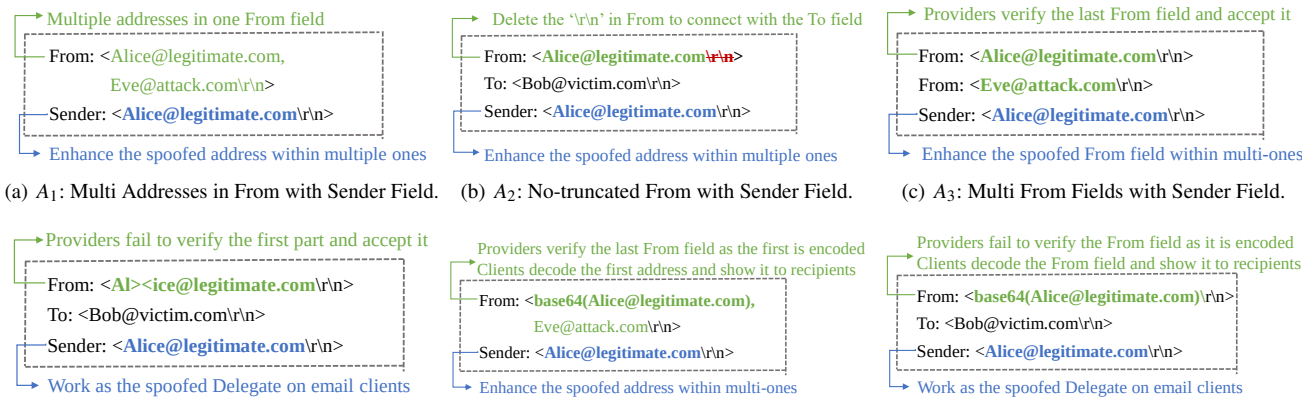
Experimental results show that five providers (*e.g.*, *mailo.com*, *qq.com*) and 17 clients (*e.g.*, Outlook on Windows) are influenced by this attack. These affected providers accept such emails in the inbox, allowing the attacker to successfully conduct this attack against recipients on the above 17 clients. Conversely, *qq.com* will drop the email without the `Sender` field into the spam folder, and this attack fails to impact seven clients that are initially susceptible (*e.g.*, Foxmail on Windows) due to the lack of the `Sender` field. The supporting details are presented in Table 6(a) in the appendix.

Attack 2: From field none-truncation (A_2). The terminator defined in emails is a character consisting of two ASCII letters named "Carriage-Return Line-Feed" (CRLF). It indicates the end of SMTP commands or the email content, including header fields like `From`. As shown in Figure 7(b), an email with none-truncated `From` deletes the CRLF in `From` and connects it with the subsequent field `To`, with the `Sender` field consistent with the `From` field. When parsing the address in `From`, the server recognizes all the content preceding a CRLF (which contains the receiver's address in `To`) as the message author and validates the email as self-sent. However, email clients may recognize it as multi-address and display the first as the email author, with the `Sender` field enhancing the reliability of the spoofed author. Compared to general multi-address ones (A_1), such emails are more likely to enter the inbox, thereby increasing the success rate of spoofing attacks. According to the measurement results, six email providers (*e.g.*, *163.com*, *139.com*) and nine clients (*e.g.*, Windows Mail, Gmail on iOS) are vulnerable to this attack. Compared to the results in A_1, six service providers accept these messages into the inbox, including the ones that directly reject multi-address emails (*e.g.*, *163.com*, *yeah.net*). This significantly improves the inbox rate and raises more potential risks than attack A_1. In the contrast test, no specific differences are observed in email processing. However, the results change on four clients when handling emails without the `Sender` field. More details are shown in Table 6(b) in the appendix.

Attack 3: Multiple From fields (A_3). It is specified that emails with multiple `From` fields should be directly rejected [33]. However, even for providers that employ various preventive measures, such emails are still allowed to enter the inbox (*e.g.*, *139.com*, *mailo.com*). An example is shown in Figure 7(c). The email contains two `From` fields, and the first one is consistent with the `Sender` field, which is the disguised email author that attackers aim to show to recipients. The address in the second `From` field is controlled by the attacker to pass possible authentications. By utilizing the `Sender` field, attackers convince the receiving servers that the address in the first `From` field is the legitimate author, thus delivering the email to the inbox.

We observe that four email providers (*e.g.*, *sohu.com*, *139.com*) and 15 clients (*e.g.*, Outlook, Gmail on iOS) are affected by this attack. Additionally, *mailo.com* accepts the email in the inbox and displays the address in the `Sender` field as the email author on the web interface, marking it as the only successful attack conducted on web interfaces under our strict criteria. The results show no differences in email processing and the performance of target clients, as detailed in Table 6(c) in the appendix.

Attack 4: Parsing with angle brackets (A_4). The `From` field supports rich text with complex forms, which can lead to incorrect email address parsing. By carefully constructing the `From` field with special characters, attackers can misguide email servers to parse incorrectly and display the wrong address to recipients. We discover that angle brackets ("<>") are particularly effective in this attack, as shown in Figure 7(d). The disguised email author in the `From` field consists of two parts enclosed within two pairs of angle brackets, while the `Sender` field contains the correct address. Receiving servers may authenticate the first part as the email author but fail to parse it correctly. Different service providers employ varying strategies in this scenario. Seven email providers, such as *coremail.com*, accept these emails into the inbox, while *zoho.com* rejects them.



(a) A_1: Multi Addresses in From with Sender Field. (b) A_2: No-truncated From with Sender Field. (c) A_3: Multi From Fields with Sender Field.

(d) A_4: Parsing From with Angle Brackets. (e) A_5: Multi From Fields with One Encoded. (f) A_6: Encoded From with Sender Field.

Figure 7: Demonstration of the Methodologies in Spoofing Attacks.

As for the results on email clients, five of them are susceptible to this attack (*e.g.*, Outlook on Windows, Gmail on Android). Influenced by the `Sender` field, these clients tend to tolerate the illegal address and remove the additional angle brackets in the `From` field, presenting the intact address of Alice to the victim Bob. For the comparison trial, no differences are observed during email processing, but the attack fails when targeting NetEase on iOS due to the lack of the `Sender` field. More details are summarized in Table 6(d) in the appendix.

Attack 5: Multiple addresses with one encoded (A_5). Per Figure 7(e), address encoding is also an effective method for email spoofing. The original SMTP protocol only supports ASCII characters, and to extend support for non-ASCII characters, RFC 2047 [25] defines two encoding algorithms: Base64 and quoted-printable. Unlike typical multi-address emails, attackers can encode the first address with Base64 and place it in the `Sender` field, while the second address belongs to the attackers themselves. Receiving servers may not identify the encoded address and use the second one for authentication. Clients like Foxmail are inclined to decode the first address and recognize it as a multi-address email, thereby displaying the address in the `Sender` field as the primary author. We find that four email service providers (*e.g.*, *qq.com*, *sohu.com*) and six email clients (*e.g.*, Outlook, Apple Mail on iOS) are vulnerable to this attack. As for the comparative test, no providers show a difference in processing these emails, but the attack fails on four out of the six clients (*e.g.*, Outlook and Netease on iOS) due to the lack of the `Sender` field, as shown in Table 6(e) in the appendix.

Attack 6: Address encoding (A_6). Unlike attack A_5, this type of forged email contains only a single address encoded in the `From` field, as shown in Figure 7(f). SMTP servers receiving such emails fail to authenticate the encoded address and deliver them to the inbox. Conversely, email clients typically decode the `From` field and present the decoded address as the email author to the recipients.

The results indicate that seven service providers (*e.g.*, *163.com*, *coremail.com*) and 11 clients (*e.g.*, Foxmail on Windows, Thunderbird on Linux) are vulnerable to this attack. Compared to the original approach in prior works [3, 36], the `Sender` field helps enhance the reliability of the spoofed address, thereby increasing the success rate of the attack. When the `Sender` field is removed from attack emails, the implementation of A_6 fails on five clients that were previously vulnerable (*e.g.*, QQ Mailbox and NetEase on iOS). Refer to Table 6(f) in the Appendix for more details.

We need to clarify that some methods for manipulating the `From` field (A_1, A_3, A_5, A_6) have been mentioned and evaluated in previous works [3, 36]. While these measures share some similarities with prior research, our work primarily focuses on the issues within the Delegation Mechanism and extends beyond the manipulation of the `From` field. We explore various approaches for crafting this field and chose the above six methods for their broader applications and higher success rates.

6.2 Experiment Results and Analysis

The results of the *Spoofing Attack* are illustrated in Table 2 and Table 4. We summarize our experimental findings as follows.

Finding-1: Attack emails can reach the inboxes of half of the email providers but are not significantly effective on their web interfaces. As shown in Table 4, 8 out of 16 email providers are vulnerable to various attacks. These affected providers tend to accept forged emails into their inboxes and present them to the recipients, raising potential risks for corresponding users. However, only one provider is successfully attacked on its web interface. After careful analysis, we identify three reasons for these unsuccessful attempts.

Firstly, 8 resilient providers implement strict security extensions, such as DMARC, which renders most attack emails invalid in authentication, leading to their rejection or identification as spam. Secondly, the operations for presenting the `From` field do not align with the attacker's expectations on the web interfaces. For instance, *qq.com* and *coremail.com* dis-

Table 4: Results of Email Spoofing Attacks on Email Providers.

Service	A_1[1]		A_2		A_3		A_4		A_5		A_6	
	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender
Gmail.com	×[2]	×	×	×	-	-	-	-	×	×	-	-
Outlook.com	-	-	-	-	-	-	-	-	-	-	-	-
163.com	×	×	✓	✓	×	×	✓	✓	×	×	✓	✓
Zoho.com	×	×	×	×	×	×	×	×	×	×	×	×
Yandex.com	-	-	-	-	×	×	-	-	-	-	-	-
Naver.com	×	×	-	-	×	×	-	-	×	×	×	×
QQ.com	✓	-	×	×	-	-	-	-	✓	✓	✓	✓
126.com	×	×	✓	✓	×	×	✓	✓	×	×	✓	✓
Rambler.com	-	-	-	-	-	-	-	-	-	-	-	-
Sohu.com	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Sina.com	-	-	-	-	-	-	-	-	-	-	-	-
139.com	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Mailo.com	✓	✓	×	×	✓	✓	✓	✓	✓	✓	-	-
Tutanota.com	-	-	-	-	-	-	-	-	-	-	-	-
Coremail.com	✓	✓	✓	✓	✓	✓	✓	✓	-	-	✓	✓
Yeah.net	×	×	✓	✓	×	×	✓	✓	×	×	✓	✓

[1] A_1 - A_6: Attacks 1 to 6 discussed in Section 6.1.
[2] "✓": attack emails reach the inbox; "×": the attack emails are rejected by the service provider; "-": the attack emails are recognized as spam.

Table 5: Results of Spoofing Attacks on Email Providers and Clients.

Attack Results	A_1		A_2		A_3		A_4		A_5		A_6		Total[2]
	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender	Sender	w/o Sender	
Reach the inboxes	5/16[1]	4/16	6/16	6/16	4/16	4/16	7/16	7/16	4/16	4/16	7/16	7/16	8/16
Successfully attack the web interfaces	0/16	0/16	0/16	0/16	1/16	1/16	0/16	0/16	0/16	0/16	0/16	0/16	1/16
Successfully attack the email clients	17/20	17/20	11/20	6/20	17/20	17/20	5/20	4/20	7/20	4/20	13/20	9/20	20/20

[1] The number "x/y" means "x out of y".
[2] The "Total" denotes the proportion of all providers that accept any attack emails in the inbox, and the proportion of clients and web interfaces that are susceptible to any attacks.

play the encoded address to the recipients in attack A_6 rather than presenting the decoded results. Lastly, as explained earlier, exposing the Delegate is an effective way to prevent spoofing emails, and all 10 services adopting the Delegation Mechanism correctly expose the attacker's address to the recipients, as shown in Table 1.

We can classify the 10 providers into four types based on their approaches to implementing the Delegation Mechanism when processing attack emails: 1) *qq.com* utilizes the original `Sender` field as the Delegate and adds the actual sending address (*attack.com*) if necessary; 2) Four providers (*163.com*, *126.com*, *yeah.net*, and *coremail.com*) modify (or add) the address in the `Sender` field to match the `MAILFROM` command, and then use the modified field as the Delegate; 3) *gmail.com*, *zoho.com*, and *sohu.com* add a "Return-Path" entry based on the actual sending address and present it as the Delegate while retaining the original `Sender` field; 4) *139.com* and *sina.com* append a specific field (e.g., "X-Sender") in the email header, with the original `Sender` field unchanged. Although these entries resemble the `Sender` field, they can only be parsed on corresponding web interfaces and are ineffective in email clients. As various providers deploy different strategies to implement the Delegation Mechanism, their web interfaces can correctly expose the email Delegate to the recipients.

Finding-2: All 20 clients are vulnerable to various attacks. With the proliferation of mobile devices and personal computers, email clients on various operating systems have garnered the attention of billions of email users due to their portability and convenience. Based on the results shown in Table 2, all 20 clients exhibit weaknesses to various attacks, including popular ones like Gmail and Outlook. Using forgery approaches, attackers can successfully conduct spoofing attacks on users across all clients via appropriate email providers. Compared to web interfaces, email clients are more susceptible to these spoofing attacks for the following reasons:

1) Email clients are more likely to present the `From` field in a way that aligns with the attacker's intentions compared to web interfaces. Due to a smaller display area, email clients are more inclusive of irregular forms of the `From` field, increasing the risks of email spoofing.

2) As shown in Table 1, 10 providers adopt various strategies to expose the correct Delegate to the recipients, including the `Sender` (*e.g.*, *163.com*), `Return-Path` (*e.g.*, *gmail.com*), and other header fields. However, email clients only employ the `Sender` field for this purpose. Some mainstream providers append specific entries with their local strategies, which email clients cannot parse while keeping the original `Sender` invari-

ant. Consequently, email clients can only display the forged Delegate in the `Sender` field rather than the correct one shown on web interfaces. These differences in implementations explain the different Delegates of the same email, thus resulting in various attack outcomes.

3) Moreover, as emphasized above, exposing the Delegate is a significant way to prevent spoofing emails. However, seven out of the 20 clients (*e.g.*, Gmail) do not employ the policy to present the Delegate, making failed attempts on web interfaces practical for these clients. While some providers employ strategies to verify the `Sender` field such as *163.com*, spoofing emails they receive can still successfully attack the recipients using the Gmail client. The absence of the Delegation Mechanism raises potential risks for spoofing attacks on these clients.

Finding-3: The Sender field significantly increases the success rate of attacks, especially when targeting email clients. Due to the standardization of the feature [33], the `Sender` field significantly influences the design of some email clients, thereby increasing the success rate of various spoofing attacks, as shown in Table 5. Specifically, for clients that adopt the policy to expose the email Delegate (*e.g.*, Outlook, NetEase), the spoofed `Sender` field improves the success rate evidently. However, its impact is limited on web interfaces due to their well-established security strategies. This highlights the importance of implementing robust security measures within client applications.

We also identify two reasons that influence the impact of this field on email clients. Firstly, as described in Section 4.2, seven out of 20 clients do not adopt the policy of presenting the Delegate to the recipients. Consequently, the `Sender` field proves inconsequential to attack results on these seven clients.

Another reason is the modification of the `Sender` field by certain providers. As mentioned in Section 4.2, five email providers (*e.g.*, *163.com*) modify or add the `Sender` field in original emails, resulting in consistent email representations regardless of the existence of this field. These providers adopt various strategies to verify the `Sender` field, thereby preventing such spoofing attacks to some extent. However, by exploiting the inconsistencies in the implementation of the Delegation Mechanism between providers and clients, we demonstrate that they are still vulnerable to these attacks.

7 Root Causes and Main Conclusions

7.1 Root Causes

We summarize the root causes of these attacks in three aspects.

First, the `Sender` field, which is defined as the delegation feature in RFC 5322, is mostly neglected by email security mechanisms. This allows the attackers to arbitrarily modify this field, thereby raising potential security issues.

Second, exposing the email Delegate is an effective measure to defend against spoofing attacks. However, various email providers implement the Delegation Mechanism differently on their web interfaces, while major clients only parse the `Sender` field for this purpose. Moreover, some clients do not expose the Delegate to recipients, increasing the potential risks of email spoofing. These inconsistencies within email providers and clients raise the security vulnerabilities of spoofing attacks, especially for client users at scale.

Third, since it is defined in RFC 5322 [33], the `Sender` field enhances the validity of the spoofed address and significantly influences the implementation of most email clients, thereby increasing the success rate of email spoofing on these clients.

7.2 Answers to Research Questions.

We can now address the research questions presented in Section 1.

Answer-1: Our investigations reveal that 10 out of 16 service providers have adopted different strategies to implement the Delegation Mechanism, with half of them using the `Sender` field as the Delegate. When processing attack emails with the spoofed `Sender` field, these five providers will take various measures to verify this field, while the majority do not validate its authenticity. Conversely, 13 out of 20 email clients display the Delegate to users, and all of them parse the `Sender` field for this purpose. Exposing the email Delegate is widely utilized in both web interfaces and clients as a defensive measure against spoofed emails.

Answer-2: To assess recipients' comprehension of the inconsistency between the email author and spoofed Delegate, we conduct a user study involving 50 participants. Our main observation is that deceptive Delegates misled 50% of participants into identifying forged emails as legitimate. However, the remaining half correctly identified such suspicious emails. Moreover, the misleading Delegate information can prompt recipients to scrutinize the email's authenticity more carefully, potentially increasing their vigilance.

Answer-3: Since all clients present the `Sender` field as the Delegate, attackers can deceive many recipients by simply appending a deceptive `Sender` field in spoofed emails. However, this trick is not significantly effective for users. To implement more effective attacks, we propose six email spoofing methods using the `Sender` field, which can bypass the defenses of half the mainstream providers we have measured. By exploiting issues within the Delegation Mechanism, attackers can impersonate any entity to send emails without triggering security warnings, making these attacks imperceptible to recipients. The experimental results indicate that all 20 clients are vulnerable to such attacks, putting billions of users at risk of email forgery.

Our findings demonstrate that vulnerabilities within the Delegation Mechanism pose a significant threat to email security, affecting both email providers and clients. In Section 8, we provide a comprehensive discussion of the countermeasures to address these issues.

8 Defense Suggestions

To prevent vulnerabilities within the Delegation Mechanism, we propose a robust validation scheme for email providers. We also offer several suggestions for clients and users to enhance their security measures.

8.1 Validation Scheme on the Sender Field

As defined in RFC 5322 [33], the `Sender` field indicates the agent responsible for email transmission. Considering possible relays during email delivery, the `Sender` field should be consistent with the `MAILFROM` command in *the first SMTP session* (where the receiving server must be public-trusted). Therefore, we propose implementing an alignment scheme between the `Sender` field and the `MAILFROM` command at the beginning of email transmission. The primary focus of the validation process is to ensure consistency among `Sender`, `MAILFROM`, and `From` under various scenarios utilizing the `Sender` field.

Building upon the concept outlined above, we recommend implementing a rigorous strategy to validate the `Sender` field within the first SMTP session. If the email author does not align with the `MAILFROM` command, or if multiple authors are specified in the `From` field, the receiving server should append a `Sender` entry directly with the address specified in the `MAILFROM` command. Additionally, any original `Sender` field in received emails should be eliminated if it exists. By generating a legitimate `Sender` based on received emails, rather than solely inspecting the original field, we can effectively authenticate the validity of the `Sender` field while ensuring proper parsing of such an added field on email clients.

However, the aforementioned strategy may not be suitable for all real-world scenarios, as organizations may choose to use specialized servers to collectively transmit their emails. In such cases, it is reasonable for these emails to contain a `Sender` field with a reputable domain (the delivery server), and the *first SMTP session* occurs between the organization's server and the dedicated server. It is inappropriate for specialized delivery servers to strictly follow the authentication process. Instead, the common practice is to append a `Sender` field with their domain. Consequently, we emphasize that the authentication strategy should be implemented in two modes, as illustrated in Figure 8. For emails from such organizations, the authenticating server can simplify the validation process by appending a `Sender` field with their domain without checking for inconsistencies within related fields.

8.2 Suggestions for Clients

It is insufficient to take actions only on the providers' side since such vulnerabilities are relevant to the entire email system. Based on our findings, we derive four suggestions for email clients summarized as follows. Firstly, it is essential to

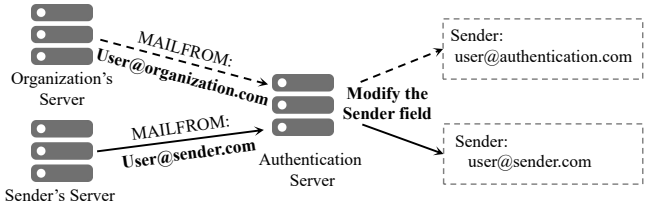


Figure 8: The Validation Scheme on the Sender Field.

display the original content in the `From` field to prevent spoofing email authors in specific formats. Specifically, clients should present all the accounts in multi-address emails (A_1 and A_5) or display the truncated address enclosed in angle brackets for A_4. This approach effectively helps recognize spoofing emails while reducing the payload for processing irregular `From` fields. Secondly, deploying the strategy of exposing the email Delegate in all email clients is crucial. It has proven effective in recognizing phishing and has been adopted by most web interfaces and clients. Thirdly, it is advantageous for clients to parse the entries appended by mainstream service providers within the email header. As mentioned in Section 6.2, providers like *gmail.com* prefer to add a "Return-Path" field to the email header, which could also be parsed similarly to `Sender` and displayed as the Delegate on clients. Lastly, we suggest displaying a warning message to recipients when an email with the `Sender` field is shown, reminding them to carefully verify the authenticity of the email.

8.3 Suggestions for Email Users

For email users, we propose three suggestions to help verify such spoofed emails. Firstly, we recommend checking important emails through web interfaces rather than email clients, as web interfaces of mainstream service providers often deploy well-established security measures and provide more detailed information to help verify emails, such as the raw email content and the authentication results of security extensions. Secondly, for users who prefer email clients, we suggest replying to suspicious emails and observing the return address. This can help verify the legitimacy of important emails and also identify potential phishing attempts. Thirdly, for clients that allow access to fetch the raw email content, such as Thunderbird, we suggest examining the header fields like `From` and `Sender` in the raw email content to identify possible spoofing attacks. We believe that the raw email content significantly helps identify potential spoofed emails.

9 Related Works

Recent research has primarily focused on email security problems from three perspectives.

Email security protocols. The original SMTP protocol lacks

security considerations, making it vulnerable to impersonation attacks. Although security extensions (*e.g.*, SPF, DKIM) have been deployed to mitigate forgery attacks, vulnerabilities within the protocols themselves pose new threats. Due to concerns surrounding these flaws, researchers focus on the problems with email security extensions [8,10,12,39–41]. Additionally, various works concentrate on enhancing the original SMTP protocol [20,32]. Our work primarily addresses issues related to the Delegation Mechanism, an advanced feature in the SMTP protocol that is mostly neglected in the email security literature.

User study within email security. User studies play a crucial role in email security research as they provide realistic data from real-world users, enabling researchers to conduct more accurate analyses. Various user studies have been conducted to address different aspects of email security. Some researchers have recognized the impact of factors such as email presentation [9,23,30,46,47] and text language [11] on phishing recognition. Additionally, various works emphasize the importance of phishing training to improve users' ability to identify phishing emails [1,18,19,21,29,37,42,43]. Our work is partially inspired by their approaches to include a real-world user study to evaluate the effectiveness of forged delegate messages on deceiving email users.

Email spoofing attacks. Email attacks are a severe challenge to cybersecurity, as they allow malicious actors to breach security measures and send harmful emails to unprotected victims. Some research focuses on breaking email encryption or spoofing digital signatures in OpenPGP [2,7,14,26,31,35,38]. Schneider *et al.* explored the issues of email bomb attacks against email system availability [34]. Our work is closely related to the studies that provide new insights into email spoofing [3,13,36]. Such attacks aim to exploit the vulnerabilities inherent in email systems and impersonate others to send spoofing emails. These attacks are more elaborate and effective than original phishing emails, as they significantly resemble those from legitimate entities. Prior research has proposed various email forgery methods. Our work differs in that it concentrates on the authentication issues within the Delegation Mechanism, which is a highly practical threat but is inconsistently handled by popular email service providers and clients, and largely neglected by email security solutions.

10 Conclusion

The Delegation Mechanism is an email service feature that enables users to delegate their permissions to a third party. However, as first presented in this paper, it not only offers convenience but also introduces a new vulnerability by lacking a validation method for the `Sender` field, which is the primary theme in the Delegation Mechanism. This flaw allows malicious attackers to easily manipulate the email Delegates. In this paper, we undertook a comprehensive investigation of the security issues associated with the Delegation Mechanism in three main steps. Firstly, we evaluated the implementations of the Delegation Mechanism within various email providers and clients. The results revealed that email providers and clients have inconsistent implementations of this mechanism. Secondly, we conducted a user study and observed that even exported and trained users can fall victim to spoofed Delegate information. Thirdly, we disclosed six email spoofing attacks that can successfully bypass the security measures of half of the major email providers and affect all the clients. Finally, we presented countermeasures and disclosed these vulnerabilities to the respective service providers.

Limitations and Future Work. Our work has several limitations. Firstly, the diversity of email products presents a significant challenge in measuring the Delegation Mechanism across all platforms. While we selected various mainstream providers and clients as targets for our study, the results may not necessarily apply to other products. Secondly, there are fewer approaches to manipulating the `From` field compared to previous works [3,36], as mainstream email providers have implemented defensive measures to prevent such emails. Although we have explored various methods for crafting spoofed emails, the majority are only effective on certain providers and become useless when targeting others. Future work will expand our research by developing more effective methods for crafting spoofed emails. Additionally, we plan to investigate the security issues within email clients to enhance protections against spoofing attacks.

11 Acknowledgement

We thank all anonymous reviewers and our shepherd for their valuable comments and suggestions. This work of Jinrui Ma, Lutong Chen, Kaiping Xue, Xuanbo Huang, Mingrui Ai, Huanjie Zhang, and Yan Zhuang is supported in part by the National Natural Science Foundation of China under Grant No. 62372425 and No. 62302472, Anhui Provincial Key Research and Development Plan under Grant No. 2022a05020050, and Youth Innovation Promotion Association of the Chinese Academy of Sciences (CAS) under Grant No. Y202093.

References

[1] Shahryar Baki and Rakesh M. Verma. Sixteen years of phishing user studies: What have we learned? *IEEE Transactions on Dependable and Secure Computing*, 20(2):1200–1212, 2023. DOI=10.1109/TDSC.2022.3151103.

[2] Jon Callas, Lutz Donnerhacke, Hal Finney, David Shaw, and Rodney Thayer. Openpgp message format. https://www.rfc-editor.org/info/rfc4880, 2007. RFC 4880, IETF, Accessed on June, 2024.

[3] Jianjun Chen, Vern Paxson, and Jian Jiang. Composition kills: A case study of email sender authentication. In *Proceedings of the 2020 USENIX Security Symposium (USENIX Security)*, pages 2183–2199. USENIX Association, 2020.

[4] Cofense. 2024 annual state of email security report. https://cofense.com/annualreport/, 2024. Accessed on June, 2024.

[5] Coremail. The introduction of coremail. https://www.coremail.cn/, 2024. Accessed on June, 2024.

[6] Dave Crocker, Tony Hansen, and Murray Kucherawy. Domainkeys identified mail (DKIM) signatures. https://www.rfc-editor.org/rfc/rfc6376.txt, 2011. RFC 6376, IETF, Accessed on June, 2024.

[7] Luca De Feo, Bertram Poettering, and Alessandro Sorniotti. On the (in) security of ElGamal in OpenPGP. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 2066–2080. ACM, 2021.

[8] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J Alex Halderman. Neither snow nor rain nor MITM... an empirical analysis of email delivery security. In *Proceedings of the 2015 ACM Internet Measurement Conference (IMC)*, pages 27–39. ACM/USENIX, 2015.

[9] Serge Egelman, Lorrie Faith Cranor, and Jason I. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the 2008 ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pages 1065–1074, 2008.

[10] Jim Fenton. Analysis of Threats Motivating DomainKeys Identified Mail (DKIM). https://www.rfc-editor.org/rfc/rfc4686.html, 2006. Accessed on June, 2024.

[11] Ayako A Hasegawa, Naomi Yamashita, Mitsuaki Akiyama, and Tatsuya Mori. Why they ignore english emails: The challenges of Non-Native speakers in identifying phishing emails. In *Proceedings of the 2021 USENIX Symposium on Usable Privacy and Security (SOUPS)*, pages 319–338. USENIX Association, 2021.

[12] Hang Hu, Peng Peng, and Gang Wang. Towards understanding the adoption of anti-spoofing protocols in email systems. In *Proceedings of the 2018 IEEE Cybersecurity Development (SecDev)*, pages 94–101. IEEE, 2018.

[13] Hang Hu and Gang Wang. End-to-End measurements of email spoofing attacks. In *Proceedings of the 2018 USENIX Security Symposium (USENIX Security)*, pages 1095–1112. USENIX Association, 2018.

[14] Fabian Ising, Damian Poddebniak, Tobias Kappert, Christoph Saatjohann, and Sebastian Schinzel. Content-Type: multipart/oracle - tapping into format oracles in email End-to-End encryption. In *Proceedings of the 2023 USENIX Security Symposium (USENIX Security)*, pages 4175–4192. USENIX Association, 2023.

[15] Scott Kitterman. Sender policy framework (SPF) for authorizing use of domains in email, version 1. https://www.rfc-editor.org/rfc/rfc7208.txt, 2014. RFC 7208, IETF, Accessed on June, 2024.

[16] John Klensin. Simple mail transfer protocol. https://www.rfc-editor.org/rfc/rfc5321.txt, 2008. RFC 5321, IETF, Accessed on June, 2024.

[17] Murray Kucherawy and Elizabeth Zwicky. Domain-based message authentication, reporting, and conformance (DMARC). https://www.rfc-editor.org/rfc/rfc7489.txt, 2015. RFC 7489, IETF, Accessed on June, 2024.

[18] Ponnurangam Kumaraguru, Justin Cranshaw, Alessandro Acquisti, Lorrie Cranor, Jason Hong, Mary Ann Blair, and Theodore Pham. School of phish: a real-world evaluation of anti-phishing training. In *Proceedings of the 2009 Symposium on Usable Privacy and Security (SOUPS)*, pages 1–12. USENIX Association, 2009.

[19] Daniele Lain, Kari Kostiainen, and Srdjan Čapkun. Phishing in organizations: Findings from a large-scale and long-term study. In *Proceedings of the 2022 IEEE Symposium on Security and Privacy (S&P)*, pages 842–859. IEEE, 2022.

[20] Hyeonmin Lee, Md. Ishtiaq Ashiq, Moritz Müller, Roland van Rijswijk-Deij, Taekyoung "Ted" Kwon, and Taejoong Chung. Under the hood of DANE mismanagement in SMTP. In *Proceedings of the 2023 USENIX Security Symposium (USENIX Security)*, pages 1–16. USENIX Association, 2023.

[21] Tian Lin, Daniel E Capecci, Donovan M Ellis, Harold A Rocha, Sandeep Dommaraju, Daniela S Oliveira, and Natalie C Ebner. Susceptibility to spear-phishing emails: Effects of internet user demographics and email content. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 26(5):1–28, 2019.

[22] Enze Liu, Gautam Akiwate, Mattijs Jonker, Ariana Mirian, Stefan Savage, and Geoffrey M. Voelker. Who's

got your mail?: characterizing mail service provider usage. In *Proceedings of the 2021 ACM Internet Measurement Conference (IMC)*, pages 122–136. ACM, 2021.

[23] Enze Liu, Lu Sun, Alex Bellon, Grant Ho, Geoff Voelker, Stefan Savage, and Imani N. S. Munyaka. Understanding the viability of gmail's origin indicator for identifying the sender. In *Proceedings of the 2023 USENIX Symposium On Usable Privacy and Security (SOUPS)*, pages 77–95. USENIX Association, 2023.

[24] A Melnikov and B Leiba. Internet message access protocol (IMAP)-version 4rev2. https://www.rfc-editor.org/rfc/rfc9051.txt, 2021. RFC 9051, IETF, Accessed on June, 2024.

[25] Keith Moore. Mime (multipurpose internet mail extensions) part three: Message header extensions for non-ascii text. https://www.rfc-editor.org/info/rfc2047, 1996. RFC 2047, IETF, Accessed on June, 2024.

[26] Jens Müller, Marcus Brinkmann, Damian Poddebniak, Hanno Böck, Sebastian Schinzel, Juraj Somorovsky, and Jörg Schwenk. "Johnny, you are fired!"–spoofing OpenPGP and S/MIME signatures in emails. In *Proceedings of the 2019 USENIX Security Symposium (USENIX Security)*, pages 1011–1028. USENIX Association, 2019.

[27] John Myers and Marshal Rose. Post office protocol-version 3 (POP3). https://www.rfc-editor.org/rfc/rfc1939.txt, 1996. RFC 1939, IETF, Accessed on June, 2024.

[28] Oberlo. Email client market share (as of march 2024). https://www.oberlo.com/. Accessed on June, 2024.

[29] Kathryn Parsons, Agata McCormac, Malcolm Pattinson, Marcus Butavicius, and Cate Jerram. The design of phishing studies: Challenges for researchers. *Computers & Security*, 52:194–206, 2015.

[30] Justin Petelka, Yixin Zou, and Florian Schaub. Put your warning where your link is: Improving and evaluating email phishing warnings. In *Proceedings of the 2019 ACM Conference on Human Factors in Computing Systems (CHI)*, pages 1–15. ACM, 2019.

[31] Damian Poddebniak, Christian Dresen, Jens Müller, Fabian Ising, Sebastian Schinzel, Simon Friedberger, Juraj Somorovsky, and Jörg Schwenk. Efail: Breaking S/MIME and OpenPGP email encryption using exfiltration channels. In *Proceedings of the 2018 USENIX Security Symposium (USENIX Security)*, pages 549–566. USENIX Association, 2018.

[32] 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 *Proceedings of the 2021 USENIX Security Symposium (USENIX Security)*, pages 4365–4382. USENIX Association, 2021.

[33] Paul Resnick. Internet message format. https://www.rfc-editor.org/rfc/rfc5322.txt, 2008. RFC 5322, IETF, Accessed on June, 2024.

[34] Markus Schneider, Haya Shulman, Adi Sidis, Ravid Sidis, and Michael Waidner. Diving into email bomb attack. In *Proceedings of the 2020 IEEE International Conference on Dependable Systems and Networks (DSN)*, pages 286–293. IEEE, 2020.

[35] Jörg Schwenk, Marcus Brinkmann, Damian Poddebniak, Jens Müller, Juraj Somorovsky, and Sebastian Schinzel. Mitigation of attacks on email end-to-end encryption. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1647–1664. ACM, 2020.

[36] Kaiwen Shen, Chuhan Wang, Minglei Guo, Xiaofeng Zheng, Chaoyi Lu, Baojun Liu, Yuxuan Zhao, Shuang Hao, Haixin Duan, Qingfeng Pan, et al. Weak links in authentication chains: A large-scale analysis of email sender spoofing attacks. In *Proceedings of the 2021 USENIX Security Symposium (USENIX Security)*, pages 3201–3217. USENIX Association, 2021.

[37] Kuldeep Singh, Palvi Aggarwal, Prashanth Rajivan, and Cleotilde Gonzalez. Cognitive elements of learning and discriminability in anti-phishing training. *Computers & Security*, 127:103105, 2023. DOI: https://doi.org/10.1016/j.cose.2023.103105.

[38] Michael A. Specter, Sunoo Park, and Matthew Green. KeyForge: Non-Attributable email from Forward-Forgeable signatures. In *Proceedings of the 2021 USENIX Security Symposium (USENIX Security)*, pages 1755–1773. USENIX Association, 2021.

[39] Dennis Tatang, Florian Zettl, and Thorsten Holz. The evolution of dns-based email authentication: measuring adoption and finding flaws. In *Proceedings of the 2021 Springer International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, pages 354–369. Springer, 2021.

[40] Chenkai Wang and Gang Wang. Revisiting email forwarding security under the authenticated received chain protocol. In *Proceedings of the ACM Web Conference (WWW)*, pages 681–689. ACM, 2022.

[41] 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 *Proceedings of the 2022 USENIX Security Symposium (USENIX Security)*, pages 1185–1201. USENIX Association, 2022.

[42] Rick Wash and Molly M Cooper. Who provides phishing training? facts, stories, and people like me. In *Proceedings of the 2018 ACM Conference on Human Factors in Computing Systems (CHI)*, pages 1–12, 2018.

[43] Rick Wash, Norbert Nthala, and Emilee Rader. Knowledge and capabilities that Non-Expert users bring to phishing detection. In *Proceedings of the 2021 USENIX Symposium on Usable Privacy and Security (SOUPS)*, pages 377–396. USENIX Association, 2021.

[44] Zikai Alex Wen, Zhiqiu Lin, Rowena Chen, and Erik Andersen. What.hack: Engaging anti-phishing training through a role-playing phishing simulation game. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI)*, page 1–12, 2019.

[45] Jason Wise. How many emails are sent per day in 2023? (update data). https://earthweb.com/how-many-emails-are-sent-per-day/, 2023. Accessed on June, 2024.

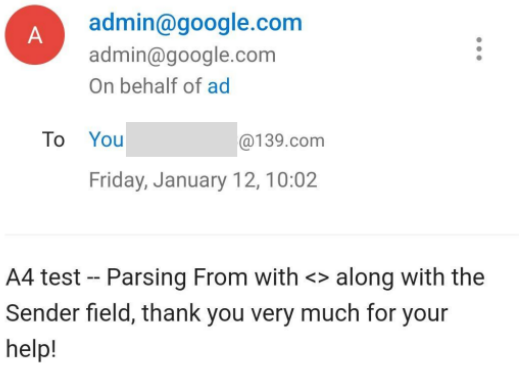
[46] Yaman Yu, Saidivya Ashok, Smirity Kaushik, Yang Wang, and Gang Wang. Design and evaluation of inclusive email security indicators for people with visual impairments. In *Proceedings of the 2023 IEEE Symposium on Security and Privacy (S&P)*, pages 2885–2902. IEEE, 2023.

[47] Sarah Zheng and Ingolf Becker. Presenting suspicious details in user-facing e-mail headers does not improve phishing detection. In *Proceedings of the 2022 USENIX Symposium on Usable Privacy and Security (SOUPS)*, pages 253–271. USENIX Association, 2022.

Appendix A - Demonstration of spoofing attacks

We list the demonstrations of attack A_4 targeting *139.com* in Figure 9. Notably, it has proven successful in executing a spoofing attack on the Gmail client, as shown in Figure 9(a). However, when attempting the same attack on the Outlook client, it encounters failure due to the presence of the Delegate, as illustrated in Figure 9(b). This inconsistency in implementing the Delegation Mechanism among different email clients may lead to varying results for these attacks, emphasizing the necessity of standardizing implementations.

(a) Successful A_4 Attack Targeting 139.com on the Gmail Client.



(b) Failed Attempt of A_4 Targeting 139.com on the Outlook Client.

Figure 9: Demonstrations of Spoofing Attack on Clients.

Appendix B - Results of six spoofing attacks

We summarize the experimental results of the six attacks targeting eight affected services across all 20 email clients in Table 6. The results of each attack consist of two rows: the highlighted row indicates the attack results involving the `Sender` field, while the second row shows the email performance without this field on various email clients. In the results table, a "✓" denotes a successful attack on the clients, and a "×" represents failed attempts. During the measurement, we noticed that some clients failed to fetch emails from certain services due to indefinite configuration errors, and we used "-" to identify these samples.

Table 6: Results of Six Spoofing Attacks Targeting Affected Service Providers on 20 Email Clients.

(a) Results of the A_1 Attack.

Service	Web interface	Windows				Linux			MacOS				iOS					Android			
		Outlook	eM Client	Win-email	Foxmail	Thunderbird	Mailspring	Evolution	Outlook	eM Client	Apple Mail	Foxmail	Gmail	Outlook	Apple Mail	QQ	Netease	Gmail	Outlook	QQ	Netease
QQ.com	×	✓	✓	✓	✓	×	✓	×	-	-	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓
	×	×	×	×	×	×	×	×	-	-	-	-	×	×	×	×	×	×	×	×	×
Sohu.com	×	✓	✓	-	✓	×	-	×	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	×	✓	✓	-	✓	×	-	×	✓	✓	×	✓	✓	✓	✓	✓	×	✓	✓	✓	✓
Coremail.com	×	×	×	✓	×	-	✓	×	×	×	×	×	✓	×	✓	-	×	✓	×	×	×
	×	✓	✓	✓	✓	-	✓	×	✓	×	×	✓	✓	×	✓	-	×	✓	✓	✓	✓
Mailo.com	×	×	✓	✓	×	×	×	×	×	×	×	×	✓	×	✓	✓	✓	✓	✓	✓	✓
	×	×	✓	✓	×	×	×	×	×	×	×	×	✓	×	✓	✓	×	✓	✓	✓	✓
139.com	×	✓	✓	✓	✓	×	✓	×	×	✓	×	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	×	✓	✓	✓	✓	×	✓	×	×	✓	×	✓	✓	✓	✓	✓	×	✓	✓	✓	✓

(b) Results of the A_2 Attack.

Service	Web interface	Windows				Linux			MacOS				iOS					Android			
		Outlook	eM Client	Win-email	Foxmail	Thunderbird	Mailspring	Evolution	Outlook	eM Client	Apple Mail	Foxmail	Gmail	Outlook	Apple Mail	QQ	Netease	Gmail	Outlook	QQ	Netease
Sohu.com	×	×	×	-	×	✓	-	×	✓	×	×	×	×	×	×	×	×	×	×	✓	✓
	×	×	×	-	×	✓	-	×	✓	×	×	×	×	×	×	×	×	×	×	✓	×
Coremail.com	×	×	×	✓	×	-	✓	×	×	×	×	×	✓	×	×	-	×	×	×	×	×
	×	×	×	✓	×	-	✓	×	×	×	×	×	✓	×	×	-	×	×	×	×	×
Yeah.net	×	×	×	✓	×	✓	✓	×	-	-	-	-	✓	×	×	×	×	×	×	×	×
	×	×	×	✓	×	✓	✓	×	-	-	-	-	✓	×	×	×	×	×	×	×	×
126.com	×	×	×	✓	×	✓	✓	×	-	-	-	-	✓	×	×	×	×	×	×	×	×
	×	×	×	✓	×	✓	✓	×	-	-	-	-	✓	×	×	×	×	×	×	×	×
163.com	×	×	×	✓	×	✓	✓	×	-	-	-	-	✓	×	×	×	×	×	×	×	×
	×	×	×	✓	×	✓	✓	×	-	-	-	-	✓	×	×	×	×	×	×	×	×
139.com	×	×	×	✓	×	✓	×	×	×	×	×	×	✓	✓	×	×	✓	×	✓	✓	✓
	×	×	×	✓	×	✓	×	×	×	×	×	×	✓	×	×	×	×	×	×	✓	×

(c) Results of the A_3 Attack.

Service	Web interface	Windows				Linux			MacOS				iOS					Android			
		Outlook	eM Client	Win-email	Foxmail	Thunderbird	Mailspring	Evolution	Outlook	eM Client	Apple Mail	Foxmail	Gmail	Outlook	Apple Mail	QQ	Netease	Gmail	Outlook	QQ	Netease
Sohu.com	×	✓	✓	-	✓	✓	-	×	×	✓	×	×	✓	✓	✓	×	✓	✓	✓	✓	×
	×	✓	✓	-	✓	✓	-	×	×	✓	×	×	✓	✓	✓	×	✓	✓	✓	✓	×
Coremail.com	×	×	×	✓	×	-	✓	✓	×	×	×	×	✓	×	✓	-	×	✓	×	×	×
	×	×	×	✓	×	-	✓	✓	×	×	×	×	✓	×	✓	-	×	✓	×	×	×
Mailo.com	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	×	×	✓	×	✓	✓	×	✓	✓	✓	×
	✓	✓	✓	✓	✓	✓	✓	✓	×	✓	×	×	✓	×	✓	✓	×	✓	✓	✓	×
139.com	×	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	×	✓	✓	✓	✓	×
	×	✓	✓	✓	✓	✓	✓	✓	×	✓	×	✓	✓	✓	✓	×	✓	✓	✓	✓	×

(d) Results of the A_4 Attack.

Service	Web interface	Windows				Linux			MacOS				iOS					Android			
		Outlook	eM Client	Win-email	Foxmail	Thunderbird	Mailspring	Evolution	Outlook	eM Client	Apple Mail	Foxmail	Gmail	Outlook	Apple Mail	QQ	Netease	Gmail	Outlook	QQ	Netease
Sohu.com	×	✓	×	-	×	×	-	×	×	×	×	×	×	×	×	×	×	×	×	×	✓
	×	✓	×	-	×	×	-	×	×	×	×	×	×	×	×	×	×	×	×	×	✓
Coremail.com	×	×	×	×	×	-	✓	×	×	×	×	×	×	×	×	-	×	✓	×	×	×
	×	×	×	×	×	-	✓	×	×	×	×	×	×	×	×	-	×	✓	×	×	×
126.com	×	×	×	×	×	×	✓	×	-	-	-	-	×	×	×	×	×	✓	×	×	×
	×	×	×	×	×	×	✓	×	-	-	-	-	×	×	×	×	×	✓	×	×	×
Mailo.com	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	✓	×	×	✓
	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	✓
163.com	×	×	×	×	×	×	✓	×	-	-	-	-	×	×	×	×	×	✓	×	×	×
	×	×	×	×	×	×	✓	×	-	-	-	-	×	×	×	×	×	✓	×	×	×
139.com	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	✓	×	×	✓
	×	✓	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	✓	×	×	✓
Yeah.net	×	×	×	×	×	×	✓	×	-	-	-	-	×	×	×	×	×	✓	×	×	×
	×	×	×	×	×	×	✓	×	-	-	-	-	×	×	×	×	×	✓	×	×	×

(e) Results of the A_5 Attack.

Service	Web interface	Windows				Linux			MacOS				iOS					Android			
		Outlook	eM Client	Win-email	Foxmail	Thunderbird	Mailspring	Evolution	Outlook	eM Client	Apple Mail	Foxmail	Gmail	Outlook	Apple Mail	QQ	Netease	Gmail	Outlook	QQ	Netease
Sohu.com	×	×	×	-	✓	×	-	×	✓	×	×	×	×	×	✓	×	×	×	×	×	×
	×	×	×	-	✓	×	-	×	✓	×	×	×	×	×	✓	×	×	×	×	×	×
QQ.com	×	×	×	✓	✓	×	×	×	-	-	-	-	×	×	✓	×	✓	×	×	×	×
	×	×	×	✓	×	×	×	×	-	-	-	-	×	×	✓	×	×	×	×	×	×
139.com	×	×	×	✓	✓	×	×	×	×	×	×	×	×	✓	✓	×	✓	×	✓	×	×
	×	×	×	✓	✓	×	×	×	×	×	×	×	×	×	✓	×	✓	×	×	×	×
Mailo.com	×	×	×	✓	✓	×	×	×	✓	×	×	×	×	×	✓	×	✓	×	×	×	×
	×	×	×	✓	✓	×	×	×	✓	×	×	×	×	×	✓	×	✓	×	×	×	×

(f) Results of the A_6 Attack.

Service	Web interface	Windows				Linux			MacOS				iOS					Android			
		Outlook	eM Client	Win-email	Foxmail	Thunderbird	Mailspring	Evolution	Outlook	eM Client	Apple Mail	Foxmail	Gmail	Outlook	Apple Mail	QQ	Netease	Gmail	Outlook	QQ	Netease
Sohu.com	×	×	×	-	✓	✓	-	✓	✓	×	✓	×	×	×	✓	×	×	×	×	✓	×
	×	×	×	-	✓	✓	-	✓	✓	×	✓	×	×	×	✓	×	×	×	×	✓	×
QQ.com	×	×	×	✓	✓	✓	×	✓	-	-	-	-	×	×	✓	✓	✓	×	×	✓	✓
	×	×	×	✓	✓	✓	×	✓	-	-	-	-	×	×	✓	×	×	×	×	×	×
126.com	×	×	×	✓	×	✓	×	✓	-	-	-	-	×	×	✓	×	×	×	×	×	×
	×	×	×	✓	×	✓	×	✓	-	-	-	-	×	×	✓	×	×	×	×	×	×
163.com	×	×	×	✓	×	✓	×	✓	-	-	-	-	×	×	✓	×	×	×	×	×	×
	×	×	×	✓	×	✓	×	✓	-	-	-	-	×	×	✓	×	×	×	×	×	×
139.com	×	×	×	✓	✓	✓	×	✓	×	×	✓	×	×	✓	✓	×	✓	×	✓	✓	✓
	×	×	×	✓	✓	✓	×	✓	×	×	✓	×	×	×	✓	×	✓	×	×	✓	×
Coremail.com	×	×	×	✓	×	-	×	✓	×	×	×	×	×	×	✓	-	×	×	×	×	×
	×	×	×	✓	×	-	×	✓	×	×	×	×	×	×	✓	-	×	×	×	×	×
Yeah.net	×	×	×	✓	×	✓	×	✓	-	-	-	-	×	×	✓	×	×	×	×	×	×
	×	×	×	✓	×	✓	×	✓	-	-	-	-	×	×	✓	×	×	×	×	×	×