Weak Links in Authentication Chains: A Large-scale Analysis of Email Sender Spoofing Attacks

Kaiwen Shen [1,*], Chuhan Wang [1,*], Minglei Guo [1], Xiaofeng Zheng [1,2,†], Chaoyi Lu [1], Baojun Liu [1,†], Yuxuan Zhao [4], Shuang Hao [3], Haixin Duan [1,2], Qingfeng Pan [5] and Min Yang [6]

[1]Tsinghua University [2]Qi An Xin Technology Research Institute [3]University of Texas at Dallas
[4]North China Institute of Computing Technology [5]Coremail Technology Co. Ltd [6]Fudan University

Abstract

As a fundamental communicative service, email is playing an important role in both individual and corporate communications, which also makes it one of the most frequently attack vectors. An email's authenticity is based on an authentication chain involving multiple protocols, roles and services, the inconsistency among which creates security threats. Thus, it depends on the weakest link of the chain, as any failed part can break the whole chain-based defense.

This paper systematically analyzes the transmission of an email and identifies a series of new attacks capable of bypassing SPF, DKIM, DMARC and user-interface protections. In particular, by conducting a "cocktail" joint attack, more realistic emails can be forged to penetrate the celebrated email services, such as Gmail and Outlook. We conduct a large-scale experiment on 30 popular email services and 23 email clients, and find that all of them are vulnerable to certain types of new attacks. We have duly reported the identified vulnerabilities to the related email service providers, and received positive responses from 11 of them, including Gmail, Yahoo, iCloud and Alibaba. Furthermore, we propose key mitigating measures to defend against the new attacks. Therefore, this work is of great value for identifying email spoofing attacks and improving the email ecosystem's overall security.

1 Introduction

Email service has been a popular and essential communicative service with abundant individual and corporate information, which makes it a key target of cyber attacks [22]. Yet, the email transmission protocols are far from capable of countering potential attacks. An email system's security relies on a multi-party trust chain maintained by various email services, which increases its systemic vulnerability to cyber attacks.

As the Wooden Bucket Theory reveals, a bucket's capacity is determined by its shortest stave. The authenticity of an email depends on the weakest link in the authentication chain. Even a harmless issue may cause unprecedented damages when it is integrated into a more extensive system. Generally, the email authentication chain involves multiple protocols, roles and services, any failure among which can break the whole chain-based defense.

First, despite the existence of various security extension protocols (e.g., SPF [24], DKIM [2] and DMARC [31]) to identify spoofing emails, spoofing attacks might still succeed due to the inconsistency of entities protected by different protocols.

Second, authentication of an email involves four different roles: senders, receivers, forwarders and UI renderers. Each role should take different security responsibilities. If any role fails to provide a proper security defensive solution, an email's authenticity can not be guaranteed.

Finally, security mechanisms are implemented by different email services with inconsistent processing strategies. Besides, those security mechanisms are implemented by different developers, some of which deviate from RFC specifications while dealing with emails with ambiguous headers. Therefore, there are a number of inconsistencies among different services. Attackers can utilize these inconsistencies to bypass the security mechanisms and present deceptive results to the webmails and email clients.

This work systematically analyzes four critical stages of authentication in the email delivery process: sending authentication, receiving verification, forwarding verification and UI rendering. We found 14 email spoofing attacks capable of bypassing SPF, DKIM, DMARC and user-interface protections. By combining different attacks, a spoofing email can completely pass all prevalent email security protocols, and no security warning is shown on the receiver's MUA. We show that it is still challenging to identify whether such an email is spoofing, even for people with a senior technical background.

To understand the real impacts of spoofing email attacks in the email ecosystem, we conducted a large-scale experiment on 30 popular email services with billions of users in total. Besides, we also tested 23 popular email clients on different

*Both authors contributed equally to this work.
†Corresponding authors:{zxf19, lbj15}@mails.tsinghua.edu.cn.

operating systems to measure the impact of attacks on the UI level. All of them are vulnerable to certain types of attacks, including reputable email services, such as Gmail and Outlook. We have already duly reported all identified issues to the involved email service providers and received positive responses from 11 of them (e.g., Gmail, Yahoo, iCloud, Alibaba Cloud).

Our work shows the vulnerability of the chain-based authentication structure in the email ecosystem. The attacks reveal that more security issues are led by the inconsistency among multiple parties' understanding and implementation of security mechanisms. To counter email spoofing attacks, we proposed a UI notification scheme. Coremail, a well-known email service provider in China, has adopted our scheme and implemented it on the webmails and email clients for users. Besides, we have also released our testing tool on Github for email administrators to evaluate and increase their security.

Contributions. To sum up, we make the following contributions:

- By analyzing the email authentication chain systematically, we identified a total of 14 email spoofing attacks, 9 of which (i.e., A_3, A_6, A_7, A_8, A_9, A_{10}, A_{11}, A_{13}, A_{14}) are new attacks, to the best of our knowledge so far. By combining different attacks, we can forge more realistic spoofing email to penetrate celebrated email services like Gmail and Outlook.
- We conducted a large-scale measurement on 30 popular email services and 23 email clients. We found all of them are vulnerable to some of attacks. We have responsibly disclosed vulnerabilities and received positive responses from 11 email vendors (e.g., Gmail, Yahoo, iCloud and Alibaba Cloud).
- To enhance the protection of email system against spoofing attacks, we proposed a UI notification scheme and provided an email security evaluation tool for email administrators to evaluate and increase their security.

2 Background

2.1 Email Delivery Process

Simple Mail Transfer Protocol (SMTP) [38] is a basic protocol for email services. Figure 1 shows the basic email delivery process. An email written by a sender is transmitted from the Mail User Agent (MUA) to the Mail Transport Agent (MTA) via SMTP or HTTP protocol. Then, the sender's MTA transmits the email to the receiver's MTA via the SMTP protocol, which later delivers the email content to the receiver's MUA via HTTP, IMAP or POP3 [27] protocols.

Extra transmission needs could complicate the actual delivery process. When the original email's target recipient is a mailing list or configured with an automatic email forwarding service, the email will be relayed through an email server, such as the email forwarding server in Figure 1. The email forwarding server will modify the receiver's address and redeliver it.

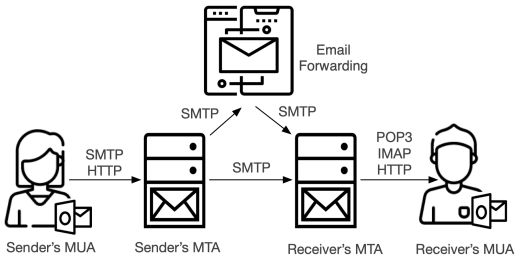


Figure 1: The email delivery process.

In the SMTP communication process, a sender's identity information is contained in multiple fields in a complex manner. (1) `Auth username`, the username used in the `AUTH` command to authenticate the client to the server. (2) `MAIL From`, the sender on the envelope, is mainly used for identity verification during the email delivery process. (3) `From`, the sender in the email body, is the displayed address that the email client shows to the user. (4) `Sender`, the `Sender` field is used to identify the real sender when there are multiple addresses in the `From`. The inconsistency of these fields provides the basis for email spoofing attacks.

As shown in Figure 1, the authentication in the email transmission process involves four important stages.

Email Sending Authentication. When sending an email from the MUA via the SMTP protocol, the sender needs to enter his username and password for authentication. In this part, the sender's MTA not only needs to verify the user's identity but also to ensure the `Mail From` is consistent with the `Auth username`.

Email Receiving Verification. When the receiver's MTA receives the email, MTA validates the sender's authenticity through SPF, DKIM and DMARC protocols. See Section 2.2.1 for details of these protocols.

Email Forwarding Verification. Email automatic forwarding is another commonly used way to send emails. When a forwarder automatically forwards an email, it should verify the sender's address. If the DKIM signature is enabled, the original DKIM verification status should be "pass" at first, then a new DKIM signature will be added. If the ARC [4] protocol is deployed, the ARC verification chain will also be verified.

Email UI Rendering. This stage is to provide users with a friendly email rendering display. Unfortunately, most popular email clients' UI will not present the authenticity check result to users. Some encoding formats or special characters can mislead receiver with a spoofing address. We argue that Email UI rendering is the last but crucial step in the authentication process, which is often overlooked in previous research.

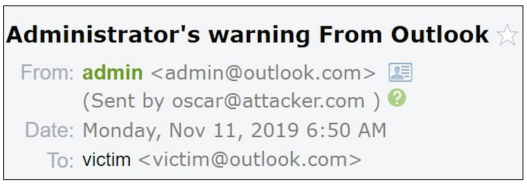


Figure 2: A spoofing email that fails the Sender Inconsistency Checks.

2.2 Email Spoofing Protections

2.2.1 Email Security Extension Protocols

To defend against email spoofing attacks, various security extensions have been proposed and standardized. At present, SPF, DKIM and DMARC protocols are the most widely used ones.

SPF. Sender Policy Framework (SPF) [24] is an IP-based authentication protocol. It marks and records the sender's domain and IP address together. The receiver can determine whether the email is from the claimed domain by querying the SPF record under the DNS server corresponding to the sender's domain name.

DKIM. DomainKeys Identified Mail (DKIM) [9] is an authentication protocol based on digital signatures. It uses an asymmetric key encryption algorithm to allow a sender to add a digital signature to an email's header to identify spoofing attempts during transmission. The receiver can retrieve the sender's public key from DNS querying to verify the signature, and then determine whether the email was spoofing or modified.

DMARC. Domain-based Message Authentication, Reporting and Conformance (DMARC) [31] is an authentication system based on the results of SPF and DKIM verification. It introduces a mechanism for multiple authenticated identifiers alignment, which associates the identity information in `From` with the authenticated identifier of SPF or DKIM. Meanwhile, the domain owner can publish a policy suggesting solutions to the recipient to handle unverified emails sent by this domain name. The domain owner can get regular feedback from the recipient. Specifically, DMARC employs an "or" status check of the SPF and DKIM verification results. If an email passes the detection of either SPF or DKIM, and `From` can be aligned with the authenticated identifier, it passes the validation of DMARC.

2.2.2 UI-level Spoofing Protections

UI rendering is a crucial part that affects the users' perception of an email's authenticity. However, the necessity of increasing UI level protection has not yet fostered any prevalent security protocol. Each Email vendor employs different UI level protections, and there is no widely accepted comprehensive protection mechanism so far.

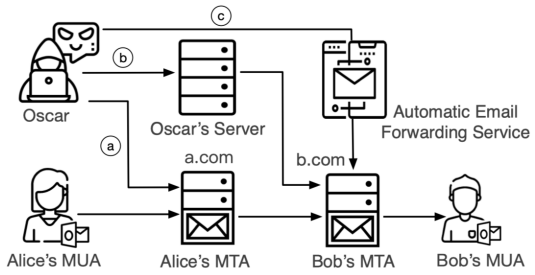


Figure 3: The Attack Model: ⓐ, ⓑ and ⓒ represent shared MTA Attack, Direct MTA Attack and Forward MTA Attack respectively.

Sender Inconsistency Checks (SIC). As shown in Figure 2, some email services add a security indicator to alert the receiver that the actual sender (`MAIL From`) may not be the displayed one (`From`). It is worth noting that this inconsistency exists throughout the email system, including email forwarding, alias, and email subscriptions. Therefore, the receiver's MTA cannot directly reject an email because of the inconsistency, which lowers the success rate to detect spoofing emails. However, the protection measure addressing this issue has not received a clear definition in the industry yet. We define this protection measure as the Sender Inconsistency Checks (SIC).

3 Attack Model and Experiments

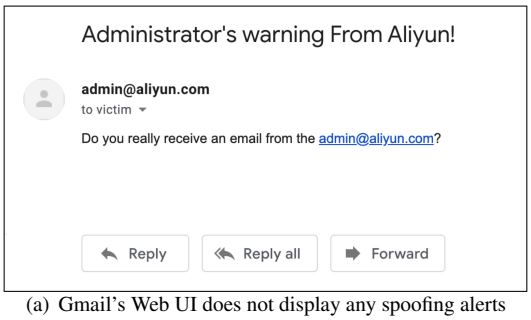
3.1 Attack Model

As shown in Figure 3, the attack model of email spoofing attacks includes a trusted email sender (Alice, which has an email account under a.com), a victim receiver (Bob, which has an email account under b.com), and an adversary (Oscar). Specifically, Oscar's goal is to send an email to Bob, spoofing Alice@a.com and bypassing all security validation.

In general, there are three common types of email spoofing attacks.

ⓐ **Shared MTA Attack.** We assume that Oscar has an email account (Oscar@a.com), which is different from Alice's account (Alice@a.com). Oscar can send spoofing emails through the MTA of a.com by modifying the `Mail From`/ `From`/ `Auth username` headers. Since the credibility of the sender's MTA IP is an essential factor affecting the spam engine's decision algorithm [5], the spoofing email can easily enter the victim's inbox. The IP of the sender's MTA is in a.com's SPF scope. The sender's MTA may also automatically attach DKIM signatures to the spoofing email. Therefore, Oscar has little difficulty in bypassing the SPF/DKIM/DMARC verification and spoofs Alice@a.com.

ⓑ **Direct MTA Attack.** Oscar can also send spoofing emails through his own email server. Note that the communication process between the sender's MTA and the receiver's MTA



(a) Gmail's Web UI does not display any spoofing alerts

Message ID	<5dcf2150.1c69fb81.4f281.9f87SMTPIN_ADDED_MISSING@mx.google.com>
Created at:	Sat, Nov 16, 2019 at 5:42 AM (Delivered after 1432 seconds)
From:	admin@aliyun.com
To:	victim@gmail.com
Subject:	Administrator's warning From Aliyun!
SPF:	PASS with IP 2402:f000:1e:4000:b061:551e:2cec:b6d Learn more
DKIM:	'PASS' with domain aliyun.com Learn more
DMARC:	'PASS' Learn more

(b) The spoofing email passes all email security protocol verification

Figure 4: A spoofing example to impersonate *admin@aliyun.com* via Gmail.

does not have an authentication mechanism. Oscar can spoof an arbitrary sender by specifying the `Mail From` and the `From` headers. This attack model can ensure all spoofing emails reach the receiver's MTA without being influenced by the strict sending check of the sender's MTA.

ⓒ **Forward MTA Attack.** Oscar can abuse the email forwarding service to send spoofing emails. First of all, Oscar can send a spoofing email to Oscar@a.com, an email account belonging to Oscar on the forwarding email service. Next, he can configure the forwarding service to automatically forward this spoofing email to the victim (Bob@b.com). This attack model has three major advantages. First, this attack has the same advantages as the Shared MTA attack mode because the receiver's MTA (b.com) believes that the emails come from the legitimate MTA (a.com). Moreover, this attack can also bypass the strict sending check of the sender's MTA (e.g., a mismatch between `Mail From` and `From` headers). Finally, the forwarding service may give the forwarded email a higher security endorsement (e.g., adding a DKIM signature that shouldn't be added).

As such, the sender authentication issues can occur in four stages, including sending authentication, receiving verification, forwarding verification and UI rendering, which can all pose potential security threats.

Further, we define the goals of a successful attack as follows: (1) the receiver's MUA incorrectly renders the sender address as it comes from a legitimate domain name, rather than the attacker's real one; (2) the receiver's MTA incorrectly verifies the sender of spoofing emails; (3) the receiver's MUA does not display any security alerts for spoofing emails.

Figure 4 shows an example of a successful email sender spoofing attack using the direct MTA attack and forward MTA attack models. The attack details are described in Section 5. All the three email security protocols give "pass" verification results to the spoofing email. Furthermore, the receiver's MUA does not display any security alerts. The victim could hardly recognize any traces of attack from such a seemingly authentic spoofing email. Therefore, it is challenging to identify whether such an email is spoofing, even for people with asenior technical background.

3.2 Experimental Target Selection

We systematically analyze 30 email services, including the most popular free public email services, enterprise-level email services and self-hosted ones. Our testing targets include the public email services that have been measured by Hu et al. [20], except for the ones that can neither be registered in China (e.g., gmx.com and sapo.pt) nor have valid SMTP services (e.g., tutanota.com and protonmail.com).

In total, we select 22 popular emails services that have more than 1 billion users. We believe their security issues can expose a wide range of common users to threats. Besides, we also select 5 popular enterprise email services, including Office 365, Alibaba Cloud and Coremail, to test the threat effect on the institutional users. As for the self-hosted email systems, we build, deploy and maintain 3 famous email systems (i.e., Zimbra, EwoMail, Roundcube).

Further, we test our attacks against 23 widely-used email clients in different desktop and mobile operating systems to evaluate the impact on the UI rendering implementation.

3.3 Experiment Methodology

This work aims to cover all possible verification issues throughout the email delivery process. Hence, we conduct a five-step empirical security analysis:

First, we systematically analyze the email specifications. In terms of syntax, we extract the ABNF rules [10], focusing on headers (e.g., `Mail From/From/Helo/Sender` headers) related to authentication. We also pay attention to semantics, particularly the identity verification of emails at each stage in the RFCs. Second, we collect legitimate email samples and generate the test samples with authentication-related headers based on the ABNF grammar [17]. Since common email services usually refuse to handle emails with highly deformed headers, we specify certain header values for our empirical experiment purposes. For example, we limit the value of `domain` to several famous email domain names (e.g., gmail.com, icloud.com). Third, we introduce the common mutation methods in protocol fuzzing [35], such as header repeating, inserting spaces, inserting Unicode characters, header encoding, and case variation. Fourth, we use the generated samples to test the security verification logic of the target

email system in four stages. Finally, we analyze and summarize the adversarial techniques that make email sender spoofing successful in practice.

3.4 Experiment Setup

In this work, we aim to summarize the potential email spoofing methods against the tested email services. Thus, we try to find out all verification issues from the four stages of the email transmission process mentioned in Section 2. Below, we first introduce the successful attacks from each stage separately. Then, we discuss our efforts to minimize the measurement bias and avoid ethical problems.

The Successful Attacks. We consider an email spoofing attack successful if either of the following four conditions is satisfied. (1) In the email sending authentication stage, an attacker can modify the identifiers (e.g., `Auth username`/ `MAIL From`/ `From`) arbitrarily. (2) In the email receiving verification stage, the receiver's MTA gives a "none/pass" verification result even if the spoofed domain name has already deployed strict SPF/DKIM/DMARC policies. Since the verification results are not always shown in the email headers, we can infer the result by checking whether the email has entered the inbox as an alternative. Besides, we consider an attack failed if our spoofing email is dropped into the spam box, which means the receiver's MTA has detected the spoofing and taken defensive measures. To avoid accidental cases, we repeat each attack three times, ensuring that the spoofing email has actually penetrated the security protocols. Only the attacks that work all three times are regarded as successful attacks. (3) In the email forwarding stage, the forwarder gives a higher security endorsement to the forwarded email. Additionally, an attack is also considered successful if the attacker can freely configure forwarded emails to any accounts without any authentication verification. (4) In the email UI rendering stage, the displayed email address is inconsistent with the real one. In this stage, we use `APPEND` function of the IMAP [11] protocol to deliver the spoofing emails into the inbox, since we only need to check the UI rendering results rather than bypass the spam engine. Finally, we collect information and analyze the results depend on the webmail and email clients on the UI level.

Minimize the Measurement Bias. First, to exclude the influence of the spam detection, we select the legitimate, benign and desensitized email samples provided by our industrial partner, a famous email provider, as the contents of our spoofing emails. These emails' content is legal and harmless and can not be judged as spam. Second, all spoofing emails are sent from 15 IP addresses located in different regions with an interval of 10 minutes. Furthermore, we deploy MX/TXT/PTR records for the attacker's domain names and IP addresses. Third, to test how the receiver's MTA handles email with "fail" SPF/DMARC verification results, we reproduce the spoofing experiments in Hu's paper [20] on our target 30 email services. We find that 23 of them reject the emails with "fail" SPF/DMARC verification results. The remaining ones mark them as spams. Besides, the results show that most of the vulnerabilities pointed in Hu's paper [20] have been fixed in the past two years.

Ethics. We have taken active steps to ensure research ethics. Our measurement work only uses dedicated email accounts owned by ourselves. No real users are affected by our experiments. We have also carefully controlled the message sending rate with intervals over 10 minutes to minimize the impact on the target email services.

3.5 Experiment Results

This work organizes all testing results in Table 1 and Table 2 to provide a general picture of the experiment results for sender spoofing attacks. The details of each attack and spoofing results are discussed in Section 4. We summarize our experiment findings as follows.

First, we measured the deployment and verification of email security protocols by these email services. All email services deploy the SPF protocol on the sender's side, while only 23 services deploy all of the three protocols. Surprisingly, all email services run the SPF, DKIM and DMARC detection on the receiver's side. However, only 12 services perform the sender inconsistency checks. Second, all target email services and email clients are vulnerable to certain types of attacks. Finally, combined attacks allow attackers to forge spoofing email which looks more authentic.

4 Email Sender Spoofing Attacks

This section describes the various techniques employed in email spoofing attacks. We divide the attacks into four categories, corresponding to the four authentication stages in the email delivery process.

4.1 Attacks in Email Sending Authentication

Email sending verification is a necessary step to ensure email authenticity. Attacks in email sending authentication can abuse the IP reputation of a well-known email service. They can even bypass all the verification of SPF/DKIM/DMARC protocols, which poses a significant threat to the email security ecosystem. These attacks are mainly used in the shared attack model (Model ⓐ).

As mentioned in Section 2.1, there are three sender identifiers in email sending process: (1) `Auth username`; (2) `Mail From`; (3) `From`. An attack is considered successful while it can arbitrarily control these identifiers during email sending authentication process.

The Inconsistency between Auth username and Mail From headers (A_1). As shown in Figure 5(a), an attacker can pretend to be any user under the current domain name to send

Table 1: Sender spoofing experiment results on 30 target email services.

Email Services	Protocols Deployment			UI Protections	Weaknesses in Four Stages of Email Flows			
	SPF	DKIM	DMARC	SIC	Sending	Receiving	Forwarding	UI Rendering
Gmail.com	✓	✓	✓	✓		A_6		A_{12}
Zoho.com	✓	✓	✓	✓	A_2	A_4	A_{11}	A_{13}
iCloud.com	✓	✓	✓		A_2	A_4, A_7	A_9	A_{12}
Outlook.com	✓	✓	✓		A_2	A_7	A_9	A_{14}
Mail.ru	✓	✓	✓			A_4		A_{12}
Yahoo.com	✓	✓	✓		A_2	A_3, A_7	A_{10}	A_{14}
QQ.com	✓	✓	✓	✓	A_2	A_5		A_{13}, A_{14}
139.com	✓		✓	✓		A_4		A_{13}
Sohu.com	✓				A_2	A_4, A_5	A_9	A_{13}
Sina.com	✓				A_2	A_3, A_4, A_5, A_8		A_{13}, A_{14}
Tom.com	✓	✓	✓		A_2		A_9	
Yeah.com	✓	✓	✓	✓	A_2	A_3, A_4, A_5, A_7, A_8	A_9	A_{12}, A_{13}, A_{14}
126.com	✓	✓	✓	✓	A_2	A_3, A_4, A_5, A_8	A_9	A_{12}, A_{13}, A_{14}
163.com	✓	✓	✓	✓	A_2	A_3, A_4, A_5, A_7, A_8	A_9	A_{12}, A_{13}, A_{14}
Aol.com	✓	✓	✓		A_2	A_5, A_7		A_{14}
Yandex.com	✓	✓	✓			A_3, A_4,A_6, A_7, A_8	A_9	A_{14}
Rambler.ru	✓	✓	✓		A_2	A_3		
Naver.com	✓	✓	✓		A_2	A_4, A_5, A_8		
21cn.com	✓				A_2	A_4, A_5	A_9	
Onet.pl	✓				A_2	A_4, A_5		
Cock.li	✓	✓			A_2	A_3, A_4		A_{13}, A_{12}
Daum.net	✓		✓			A_5		
Hushmail.com	✓	✓	✓			A_3, A_4, A_8		A_{12}
Exmail.qq.com	✓	✓	✓	✓	A_2	A_5		A_{14}
Coremail.com	✓	✓	✓	✓	A_2	A_8	A_9	
Office 365	✓	✓	✓	✓	A_2	A_4	A_9, A_{10},A_{11}	A_{14}
Alibaba Cloud	✓	✓	✓	✓	A_2	A_3, A_4, A_5, A_8	A_{10}	A_{13}
Zimbra	✓	✓	✓	✓	A_1, A_2	A_3, A_5, A_8	A_9	A_{12}, A_{13}
EwoMail	✓	✓	✓		A_2	A_3, A_4, A_8		A_{13}
Roundcube	✓	✓	✓		A_1, A_2	A_3, A_4, A_8		A_{12}

[1] The subscript identifies the specific attack (e.g., A_8 identifies the encoding based attack discussed in 4.2).
[2] The abbreviation SIC stands for the receiver's sender inconsistency checks, an email notification custom deployed by providers, described in the background 2.2.2.
[3] The cases with ✓ mean that the domain name deploys with the relevant email security protocol or perform the sender inconsistency checks.

a spoofing email whose `Auth username` (Oscar@a.com) and `Mail From` (Alice@a.com) are inconsistent during email sending authentication. SMTP protocol does not provide any built-in security features to guarantee the consistency of `auth username` and `Mail From` header. Therefore, this type of protection depends only on the software implementation of the email developer.

In our spoofing experiments, most email services have noticed such problems and prohibited users from sending emails inconsistent with their original identity. However, this type of problem still appears in some well-known corporate email software (i.e., Zimbra, EwoMail). These two email services are vulnerable under default security configuration. Email administrators need to upgrade their security configurations to prevent such problems manually.

The Inconsistency between `Mail From` and `From` headers (A_2). An attacker can send a spoofing email with different `Mail From` and `From` headers. Figure 5(b) shows this type of attack. Although some users are allowed to use email aliases to send emails with a different `From` header, no user should be allowed to freely modify the `From` header to any value (e.g., admin@a.com) to prevent attacks. The `From` header should only be allowed to be set within limited legal values. Many prevalent email services (e.g., Outlook, Sina, QQ Mail) and most third-party email clients (e.g., Foxmail, Apple Mail) only display the `From` header, not the `Mail From` header. For these emails which have different `Mail From` and `From` headers, the victim cannot even see any security alerts on the MUA.

Similar inconsistency also exists between the `RCPT To` and `To` headers. In the real world, there are some scenes that

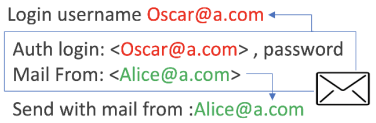
Table 2: Sender spoofing experiment results on 23 target email clients.

OS	Clients	SIC	Weaknesses
Windows	Foxmail	✓	A_6, A_7, A_{13}, A_{14}
	Outlook	✓	A_6, A_{13}
	eM Client	✓	A_6, A_{12}
	Thunderbird		A_6, A_{13}, A_{14}
	Windows Mail		A_6, A_7, A_{13}, A_{14}
MacOS	Foxmail		A_6, A_{13}
	Outlook	✓	A_6, A_{13}
	eM Client	✓	A_6, A_7, A_{12}, A_{13}, A_{14}
	Thunderbird		A_6, A_{13}, A_{14}
	Apple Mail		A_6, A_{13}, A_{14}
Linux	Thunderbird		A_6, A_{13}
	Mailspring		A_6, A_{13}, A_{14}
	Claws Mail		A_6, A_{14}
	Evolution		A_6, A_{13}, A_{14}
	Sylpheed		A_6, A_{13}, A_{14}
Android	Gmail		A_6, A_{13}
	QQ Mail	✓	A_6, A_{13}, A_{14}
	NetEase Mail		A_6, A_{12}, A_{13}
	Outlook	✓	A_6, A_{13}
iOS	Mail.app		A_6, A_7, A_{13}, A_{14}
	QQ Mail	✓	A_6, A_{13}
	NetEase Mail		A_6, A_{12}, A_{13}
	Outlook	✓	A_6, A_{13}

[1] The subscript identifies the specific attack.
[2] The SIC stands for the sender inconsistency checks.
[3] The cases with ✓ mean that the email client performs the sender inconsistency checks.
[4] Since email clients do not involve verification of the mail protocol, we only tested attacks (i.e., A_6, A_7, A_{12}, A_{13}, A_{14}) related to email UI rendering.

cause the inconsistency, such as email forwarding and Bcc. However, this kind of flexibility increases attack surfaces and introduces new security risks. For example, an attacker can send an email to a victim, even if the email's `To` header is not the address of the victim. In this case, an attacker can further use this method to obtain a spoofing email with a DKIM signature that normally could not be obtained, which is helpful for further attacks. This technique might not be effective when used alone, but it can often achieve excellent spoofing results when combined with other attack techniques.

14 email services are vulnerable to this type of attack in our experiments. In addition, we also found that some email services (e.g., Outlook, Zoho, AOL, Yahoo) have realized these risks and have implemented corresponding security restrictions. They refused to send emails with inconsistent `Mail From` and `From` headers during SMTP sending process. However, these defenses can still be bypassed by two types of attacks (i.e., A_4, A_5). For example, we can send a spoofing



(a) Attack with different auth username and Mail From header

(b) Attack with different Mail From and From headers

Figure 5: Two attacks of bypassing sending service's verification.

email with the `Mail From` header as `<Oscar@a.com>` and the `From` header as `<Alice@a.com, Oscar@a.com>` in Yahoo which introduces another source of ambiguity and eventually bypasses email protocol verification. Therefore, it is still possible to send such spoofing emails, even if the sender has deployed relevant security measures.

4.2 Attacks in Email Receiving Verification

SPF, DKIM and DMARC are the prevalent mechanisms used to counter email spoofing attacks. If an attacker can bypass these protocols, it can also pose a serious security threat to email security ecosystem. There are three attack models to launch this type of attack: shared MTA attack, direct MTA attack, and forward MTA attack. An attack is successful while the receiver's MTA incorrectly gets a 'none/pass' verification result.

Empty Mail From Attack (A_3). RFC 5321 [25] explicitly describes that an empty `Mail From` is allowed, which is mainly used to prevent bounce loop-back and allow some special message. However, this feature can also be abused to launch email spoofing attacks. As shown in Figure 6, an attacker can send an email with an empty `Mail From` header, and the `From` header fabricates Alice's identity (Alice@a.com).

The SPF protocol [23] stipulates that the receiver's MTA must complete the SPF verification based on the `Helo` field if the `Mail From` header is empty. However, the abuse of the `Helo` field in real life make some email services disobey the standard and take a more loose approach of verification. Thus, when the recipient deals with those emails, they can not complete SPF verification based on the `Helo` field, but directly return "none". This type of error allows an attacker to bypass the SPF protection. As a result, an attacker can change the SPF result of this attack from "fail" to "none".

13 email services (e.g., Yahoo, Yeah, 126, Aol) are vulnerable to this type of attacks. Fortunately, there are already 17 email services that have fixed such security issues, 5 of

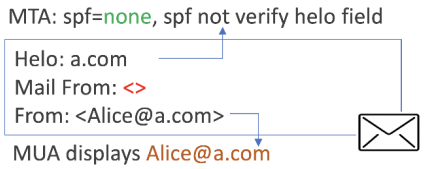
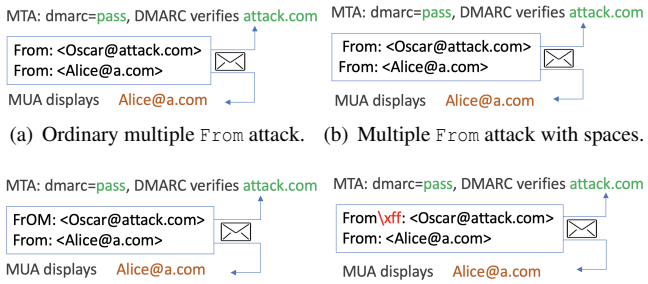


Figure 6: Empty `Mail From` attack bypassing the SPF verification.



(a) Ordinary multiple `From` attack. (b) Multiple `From` attack with spaces.

(c) Multiple `From` attack with case variation. (d) Multiple `From` attack with invisible characters.

Figure 7: Multiple `From` attacks to make DMARC verify Oscar@attack.com while the MUA displays Alice@a.com.

which (e.g., Zoho.com, iCloud.com, exmail.qq.com) drops such emails into spam.

Multiple From Headers (A_4). Inspired by the work of Chen et al. [6], we also utilize multiple headers techniques in email spoofing attacks. Compared with Chen's work, we have more distortions from the `From` header, such as adding spaces before and after the `From`, case conversion, and inserting non-printable characters. As shown in Figure 7, an attacker can construct multiple `From` headers to bypass security policies. RFC 5322 [40] indicates that emails with multiple `From` fields are typically rejected. However, there are still some email services that fail to follow the protocol and accept emails with multiple `From` headers. It can introduce inconsistencies in the email receiving verification stage, which could lead to additional security risks. Figure 7(c) shows an example that the displayed sender address is Alice@a.com, while the receiver's MTA may use Oscar@attack.com for the DMARC verification .

Only 4 mail services (i.e., Gmail, Yahoo, Tom, Aol) reject emails with multiple `From` headers, and 19 mail services are affected by this type of attacks. Most tested email services tend to display the first `From` header on the webmail, while 6 services (e.g., iCloud, Yandex, Alibaba Cloud) choose to display the last `From` header. Besides, 7 vendors have made specific security regulations against such attacks, such as showing two `From` addresses on the webmail simultaneously (e.g., QQ Mail, Coremail) or dropping such emails into the spam folder (e.g., Outlook, rambler.ru).

Multiple Email Addresses (A_5). Using multiple email addresses is also an effective technique to bypass protocol verification. Usage of multiple addresses was first proposed in RFC2822 [39] and is still explicitly allowed in RFC 5322 [40]. It is suitable for such scenarios: an email with multiple authors is supposed to list all of them in the `From` header. Then, the `Sender` field is added to mark the actual sender. As shown in Figure 8(a), an attacker can bypass DMARC verification with multiple email addresses (`<Alice@a.com>, <Oscar@attack.com>`). In addition, we can also make some rule-based mutations to these addresses, such as `[Alice@a.com], <Oscar@attack.com>`.

(a) Ordinary multiple address attack. (b) Multiple address attack with null address.

(c) Multiple address attack with semantic characters. (d) Multiple address attack with comments.

Figure 8: Multiple email addresses attacks to make DMARC verify Oscar@attack.com while MUA displays Alice@a.com.

15 mail services (e.g., QQ mail, 21cn.com and onet.pl) would still accept such emails. Only 4 services (e.g., Gmail and Mail.ru) directly reject those emails, and 5 other services (e.g., zoho.com, tom.com, outlook.com) put them into spam. The rest 6 services (e.g., 139.com, cock.li and Roundcube) display all of these addresses, making spoofing emails more difficult to deceive the victim.

Parsing Inconsistencies Attacks (A_6). `Mail From` and `From` headers are in rich text with a very complicated grammatical format. As a result, it is challenging to parse display names and real addresses correctly. These inconsistencies can allow attackers to bypass authentication and spoof their target email clients.

A mailbox address is one of the essential components of these two headers. First, mailbox addresses were allowed to have a route portion [39] in front of the real sender address when enclosed in "<" and ">". Therefore, the mailbox (`<@a.com, @b.com:admin@c.com>`) is still a legal address. Among them, `@a.com, @b.com` is the route portion, and "admin@c.com" is the real sender's address. Second, it is allowed to use mailbox-list and address-list [39], and they can have "null" members, such as `<a@a.com>, ,<b@b.com>`. Third, comment [40] is a string enclosed in parentheses. They were allowed between the period-separated elements of local-part and domain, such as `<admin(username)@a.com(domain name)>`. Finally, there is an optional display-name [40] in the `From` header. It indicates the sender's name, which is displayed for receivers. Figure 9 shows three types of attacks

(a) Parsing inconsistency with route portion. (b) Parsing inconsistency with "null" mailbox-list. (c) Parsing inconsistency with comment.

(d) NUL character truncates string parsing. (e) Invisible unicode characters truncate string parsing. (f) Semantic characters truncate string parsing.

Figure 9: Six spoofing examples of bypassing receiving service's verification.

(a) Encoding based attack bypassing DMARC verification.

(b) Combined encoding and truncated attack.

Figure 10: Two spoofing examples with encoding based attacks.

based on parsing inconsistencies.

Truncated characters are a series of characters that terminate string parsing. When parsing and extracting the target domain name from the email headers, truncated characters will end the parsing process. Figure 9(d) shows that the program gets an incomplete domain name (a.com) when parsing the target domain name from the string "admin@a.com\x00@attack.com". Attackers can use these techniques to bypass the verification of email security protocols. Overall, this work finds three types of truncated characters in the email string parsing process. First, NUL (\x00) character can terminate string in the C programming language. It has the same effect in the email field. Second, some invisible Unicode characters (e.g., `\uff00-\uffff`,`\x81-\xff`) can also terminate the string parsing process. Third, certain semantic characters, such as "`[`,`]`,`{`,`}`,`\t`,`\r`,`\n`,`;`", can be used to indicate a tokenization point in lexical analysis. Meanwhile, these characters also influence the string parsing process.

We found that 13 email services have problems in the UI rendering stage under such attacks. For Gmail and Yandex, we can use these attack techniques to bypass DMARC.

Encoding Based Attack (A_7). RFC 2045(MIME) [15] describes a mechanism denoting textual body parts, which are coded in various character sets. The ABNF grammar of these parts is as follows:`=?charset?encoding?encoded-text?=`. The "charset" field specifies the character set associated with the not encoded text; "encoding" field specifies the encoding algorithm, where "b" represents base64 encoding, and "q" represents quoted-printable encoding; "encode-text" field specifies the encoded text. Attackers can use these encoded addresses to evade email security protocol verification. Figure 10(a) shows the details such attacks. For an encoded address, such as `From: =?utf-8?b?QWxpY2VAYS5jb20=?=`, most email services do not decode the address before verifying the DMARC protocol, thus fail to extract the accurate domain and get a "None" in the following DMARC verification. However, some email services display the decoded sender address (`Alice@a.com`) on the MUA. Furthermore, this technique can be combined with truncated strings. As shown in the Figure 10(b), an attacker can construct the From header as "b64(Alice@a.com>b64(\uffff)@attack.com". Email client programs could get incomplete username(i.e., Alice@a.com), but it would still use the attacker's domain (attack.com) for DMARC verification.

7 email services are affected by the vulnerability, including some popular services (e.g., Outlook, Office 365, Yahoo) with more than one billion users.

The Subdomain Attack (A_8). An attacker can send spoofing emails from a non-existent subdomain (no MX record) of well-known email services (e.g., admin@mail.google.com). Thus, there are no corresponding SPF records. The spoofing email only gets a "None" verification result, and the receiver's MTA does not directly reject it. Although the parent domain (e.g., google.com) deploys strict email policies, attackers can still attack in this way. Unfortunately, many companies use sub-domains to send business subscription emails, such as Paypal, Gmail, and Apple. As a result, ordinary users tend to trust such emails.

Unfortunately, RFC 7208 [24] states that the use of wildcard records for publishing SPF records is discouraged. And few email administrators configure wildcard SPF records in the real world. Besides, the receiver's MTA can usually reject emails from domains without an MX record. But RFC

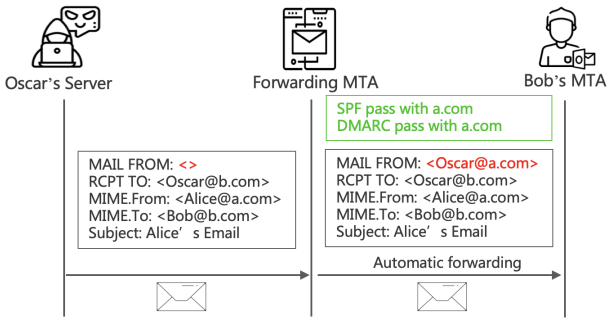


Figure 11: Exploiting forwarding services to bypass SPF and DMARC.

2821 [26] mentions that, when a domain has no MX records, SMTP assumes an A record will suffice, which means any domain name with an A record can be considered a valid email domain. In addition, many well-known websites deploy a wildcard DNS A record that makes this type of attack more applicable. As a result, it is difficult for the receiver's MTA to determine whether to reject such emails.

Experimental results show that 13 email services are vulnerable to such attacks. Only one email service (Mail.ru) deploys a wildcard DNS entry for the SPF record in our experiments. By default, the DMARC policy set for an organizational domain should apply to any sub-domains, unless a DMARC record has been published for a specific sub-domain. However, the experimental results show that our attack is still effective, even if the receiver's MTA conducted a DMARC check.

4.3 Attacks in Email Forwarding Verification

This work shows that attackers can abuse the email forwarding service to send spoofing emails that would fail in the shared MTA attack model. Besides, forwarding service may give the forwarded email a higher security endorsement. Both situations are exploitable for attackers to send spoofing emails.

Unauthorized Forwarding Attack (A_9). If the attacker can freely configure forwarded emails to any accounts without any authentication verification, the email service has unauthorized forwarding issues. First, the attacker should have a legitimate email account on the email forwarding service. Because these emails are sent from a well-known email forwarding MTA, the receiver's MTA generally accepts such emails. We can also exploit forwarding services to bypass SPF and DMARC protocols when the target domain name is the same as the forwarding domain name. This attack is depicted in Figure 11. Based on this attack, attackers can abuse the credibility of well-known MTAs to craft an realistic spoofing email.

Among our experimental targets, 12 email services have such vulnerabilities. 7 email services do not provide the email forwarding feature. The other email services have realized the risks and performed corresponding forwarding verification to fix it.

The DKIM Signature Fraud Attack (A_{10}). The forwarding service may give the forwarded email a higher security endorsement. But this feature can be abused by the attacker to send spoofing emails. The forwarder should not add a DKIM signature of its domain name if the forwarded email does not have a DKIM signature or fails the DKIM validation before. Otherwise, the attacker can defraud the forwarding services of legitimate DKIM signature. However, both RFC 6376 [34] and RFC 6377 [30] suggest that forwarders should add their signatures to the forwarded emails. It has further led to more email services have such problems.

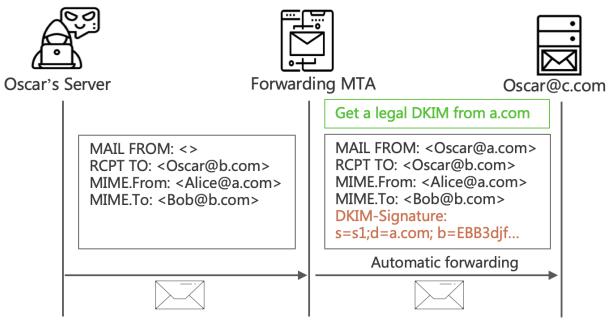
Figure 12 illustrates the complete process of the attack. The email forwarding service (a.com) signs and adds DKIM signatures to all forwarded emails without strict verification. First, the attacker can register an account (Oscar@a.com) under the email forwarding service. Second, he can configure all receiving emails forward to another attacker's email address (Oscar@c.com). The attacker can then send a spoofing email with `From`: Alice@a.com, `To`: Bob@b.com to Oscar@a.com through the direct MTA attack model. The forwarding service (a.com) adds a legal DKIM signature to this spoofing email. As a result, the attacker gets a spoofing email with a legal DKIM signature signed by a.com. In our experiments, Alibaba Cloud, Office 365, and Yahoo Email are all vulnerable to such attacks.

ARC Problems (A_{11}). ARC [4] is a newly proposed protocol that provides a chain of trust to link the verification results of SPF, DKIM, and DMARC in the email forwarding process. Only three email services (i.e., Gmail, Office 365, and Zoho) deploy the ARC protocol in our experiments. However, our research found that both Office 365 and Zoho have security issues with the ARC protocol implementation. Besides, except for the A_{10} attack, ARC cannot defend against most of the attacks discussed above.

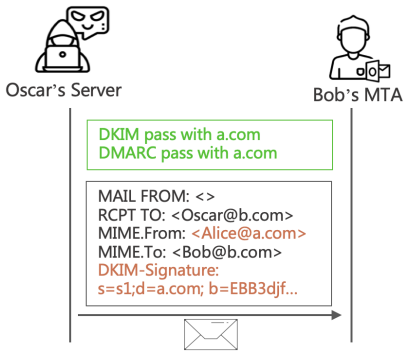
For Zoho email services, it shows alerts for users if the email fails the sender inconsistency checks. However, there is an error in Zoho's ARC implementation. When a spoofing email is automatically forwarded to the Zoho mailbox via Gmail, the ARC-Authentication-Results (AAR) header added by Zoho shows a wrong "pass" DMARC verification result. Even worse, this incorrect ARC implementation can also bypass the sender inconsistency checks. Zoho does not display alerts to users for this spoofing email. Office 365 also has errors in the implementation of ARC. It passes the wrong verification results of SPF, DKIM, and DMARC in the AAR header. This would break the ARC trust chain, which introduces more security risks.

4.4 Attacks in Email UI Rendering

The last and most crucial part of the email system is to ensure that emails are rendered correctly. Once the attacker can break the defensive measures in this stage, ordinary users are easily



(a) The spoofing email defraud a DKIM signature signed by a.com.



(b) Spoofing with the legal DKIM signature.

Figure 12: Exploiting forwarding services to bypass DKIM and DMARC.

deceived by such spoofing emails unconsciously.

The displayed address is the sender address shown on the MUA, but the real address is the sender identity (`From`) used in SMTP communication. If an attacker can make the displayed address inconsistent with the real address, the attack is considered successful. Besides, as shown in Figure 2, some MUAs add a security indicator to those emails which fail the sender inconsistency checks. If an attacker can bypass the sender inconsistency checks, it is also regarded as an effective attack technique.

There are various attacks in the email UI rendering stage. Some are similar to the A_6, A_7 attacks discussed previously. The difference is that a UI level attack's goal is to bypass the sender inconsistency checks and spoof the email address shown for users, rather than bypass the three email security protocols' verification. Thus, we usually construct ambiguous `From` headers rather than `Mail From` headers. In this section, we only discuss the attack techniques not previously mentioned.

IDN Homograph Attack (A_{12}). The homograph attack [16] is a known web security issue, but its security risks to the email system have not been systematically discussed. As popular email providers gradually support the emails from internationalized domain names (IDN), this attack is likely to have a wider security impact.

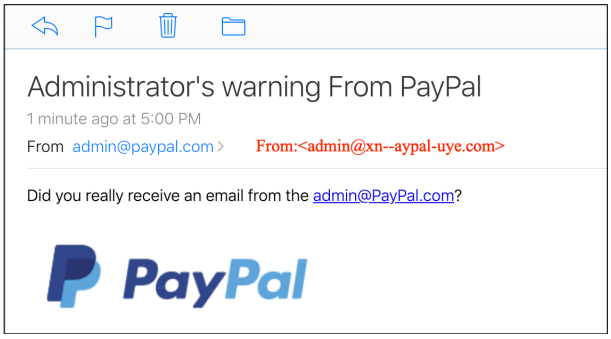


Figure 13: A example of IDN homograph attack to impersonate *admin@paypal.com* on iCloud.com web interface.

Punycode is a way of converting words that cannot be displayed in ASCII into Unicode encoding. Notably, Unicode characters can have a similar appearance on the screen while the original addresses are different. Figure 13 shows a spoofing email that seems to come from the address (admin@paypal.com), but is actually from the address (admin@xn–aypal-uye.com).

Modern browsers have implemented some defensive measures against the IDN homograph attack. For example, the IDN should not be rendered if the domain label contains characters from multiple languages. Unfortunately, we found few similar defensive measures in email systems.

The experimental results show that 10 email services (e.g., Gmail, iCloud, Mail.ru) support IDN email is displayed. Currently, only Coremail fixes this vulnerability. With our assistance, Coremail adds white spaces before and after the Unicode characters in the address bar. In this way, users can easily distinguish between ASCII characters and Unicode characters to prevent such attacks.

Missing UI Rendering Attack (A_{13}). We also find that many characters can affect the rendering of the MUA. Some characters may be discarded during the rendering process. Additionally, some characters may also cause the email address to be truncated (similar to the attack A_6). These characters include invisible characters (`U+0000-U+001F,U+FF00-U+FFFF`) and semantic characters (`@,:,;,"`). For example, the MUA renders the address `admin@gm@ail.com` as `admin@gmail.com`.

There are still 12 email services (e.g., zoho.com, 163.com, sohu.com) vulnerable to such attacks. Other services refuse to receive or just throw such emails into the spam box.

Right-to-left Override Attack (A_{14}). Several characters are designed to control the display order of the string. One of these is the "RIGHT-TO-LEFT OVERRIDE" character, U+202E which tells computers to display the text in a right-to-left order. It is mainly used for writing and reading Arabic or Hebrew text. Although this attack technique [1] has been discussed elsewhere, its security risk to email spoofing has not yet been fully explored. An attacker can construct a string as `\u202emoc.a@\u202dalice`, which is displayed

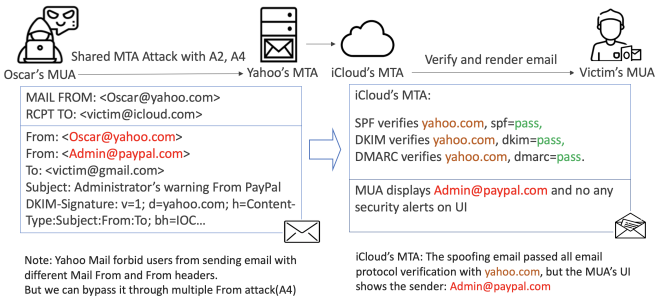


Figure 14: Combining A_2 and A_4 attacks to impersonate *admin@paypal.com* on iCloud.

as `Alice@a.com`. Because spoofing emails with RTL characters may be directly thrown into the spam box, we generally encode the payload (with utf-8 mode) to attack.

11 email services (e.g., Outlook, Yahoo, Yandex) are still vulnerable to this attack. 10 services (e.g., cock.li,daum.net, onet.pl) cannot correctly render this type of email address. Other email services directly reject such mails.

5 Combined Attacks

According to four authentication stages in email delivery process, we divide our attacks into four categories. However, these attacks have certain limitations. First, some attacks (e.g., A_2, A_3) can have a spoofing effect on the recipent. However, they can not bypass all email spoofing protections. For example, a spoofing email via Empty Mail From Attack (A_3) bypasses the SPF verification but fails in the DMARC verification. In addition, most email vendors have fixed the individually conducted attacks which can bypass all the three email security protocols in our experiment. Thus, combining multiple attacks of different stages is more feasible in practice. With a "cocktail" joint attack combining different attack techniques, we can easily construct a spoofing email that can completely pass the verification of three email security protocols and user-interface protections. Finally, there is no difference shown on the receiver's MUA between this spoofing email and a legitimate one.

There are numerous feasible combined attacks by combining 3 types of attack models and 14 attack techniques in the 4 authentication stages. This work selects two of the most representative examples to illustrate the effects of combined spoofing attacks. Table 3 lists key information of the two examples.

Combined Attacks under the Same Attack Model. We identified a total of 14 email spoofing attack techniques, of which 14 attack techniques can be combined under the same attack model to achieve better attack effects. In addition, although some vendors might fix a vulnerability through one security check, the attacker can accurately combine other attack techniques to bypass the corresponding security check.

Figure 14 shows a representative example under the shared MTA attack model. Yahoo email performs a simple sender check policy to defend against the A_2 attack. It prohibits user from sending emails with different `Mail From` and `From` headers. However, the attacker can still bypass this sender check policy through the A_4 attack. To be specific, we can send a spoofing email with a first `From` header (`Oscar@yahoo.com`), which is same as the `Mail From` header. Then, we add a second `From` header (`Admin@paypal.com`). Interestingly, iCloud does not reject such a spoofing email with multiple `From` headers. Even worse, iCloud uses the first `From` header to perform the DMARC verification and gets a "pass" result with yahoo.com, while the second `From` (Admin@paypal.com) header is displayed on the webmail's UI for users. Therefore, this combined attack can eventually bypass all three email security protocols and spoof the MUA.

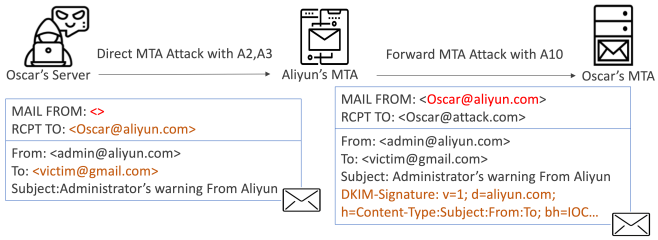
Combined Attacks under Different Attack Models. The attacker can also conduct a more effective attack by combining different attack models. The email system is a complex ecosystem with a multi-party trust chain, which relies on security measures implemented and deployed by multiple parties. Under different attack models, multiple parties may have various vulnerabilities. For example, it is difficult to attack through the shared MTA attack model if a email service's sending MTA performs strict checks in sending authentication. However, once it fails to provide a correct and complete security defensive solution in other stages, the attacker can still bypass and send spoofing emails through the other two attack models. Hence, we have more combination attacks in the real world by combining multiple attack models.

Figure 4 shows a successful spoofing attack by combining the direct and forward MTA attack models. For instance, Oscar employs the attack techniques (A_2,A_3) to send a spoofing email with empty `Mail From` and crafted `From` headers. Besides, Oscar has a legitimate account (Oscar@aliyun.com), which is different from the victim's account. Thus, Oscar can configure this account to automatically forward the received emails to one of his accounts (Oscar@attack.com). Alibaba Cloud service adds a DKIM signature to all forwarded emails without a necessary verification check (A_{10}). It grants Oscar's spoofing email a legitimate DKIM signature. Then, Oscar can send this spoofing email with `Mail From:<admin@attack.com>` header through the direct MTA attack model, which is illustrated in Figure 15(b).

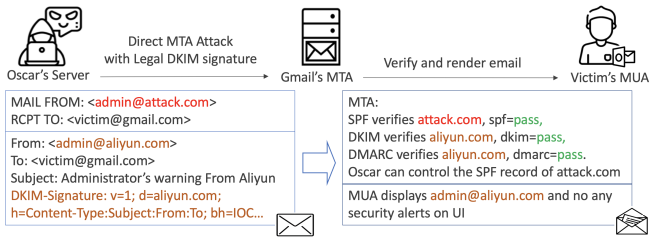
For this spoofing email, the SPF protocol verifies the `attack.com` domain, while the DKIM and DMARC protocols verify the `aliyun.com` domain. Therefore, this email can pass all the three email security protocols, and enter the inbox of Gmail. In addition, no email service shows alerts for users about the email with different verified domains of the three protocols. It further makes this type of attack more deceptive to ordinary users.

Table 3: Details of two combined attack examples.

Attack	From	To	Attack Model	Combination of attacks
Case 1	admin@paypal.com	victim@icloud.com	Shared MTA Attack	A_2 + A_4
Case 2	admin@aliyun.com	victim@gmail.com	Direct & Forward MTA Attack	A_2+A_3+A_{10}



(a) The first stage of the attack obtained an Alibaba Cloud legal DKIM signature.



(b) The second stage of the attack passed Gmail's three mail protocol security verifications.

Figure 15: A combination attack with A_2,A_3 and A_{10} from admin@aliyun.com to victim@gmail.com.

6 Root Causes and Mitigation

6.1 Root Causes

As aforementioned, the security of email systems relies on several protection policies that are separately enforced by multiple parties. Thus, the inconsistencies in these multiple parties could create more vulnerabilities and lead to severe spoofing attacks. We identify the root causes of the attacks as follows.

Weak Links among Multi-protocols. The protocol verification process is one of the weak links in the authentication chain, due to the ambiguity of email specifications, the lack of best practice and the complexity of the MIME standard. In the SMTP communication process, multiple fields of protocols contain sender's identity information (i.e., `Auth username`, `MAIL From`, `From`, `Sender`). The inconsistency of these fields provides the basis for email spoofing attacks.

SPF, DKIM, and DMARC are proposed and standardized to prevent email spoofing attacks from different aspects. However, an email system can prevent email spoofing attacks only when all protocols are well enforced. In this chain-based authentication structure, a failure of any link can render the authentication chain invalid.

Weak Links among Multi-roles. In the email system, authenticating the sender's identity is a complicated process. It involves four important roles: senders, receivers, forwarders, and UI renderers. Standard security models work on the assumption that each role properly develops and implements related security verification mechanisms to provide the overall security. However, many email services do not implement the correct security strategy in all four roles.

Many email services (e.g., iCloud, Outlook, Yeah.com) do not notice the security risks caused by unauthorized forwarding attacks (A_9) in the email forwarding stage. In addition, the specifications do not state any clear responsibilities of four roles (i.e., senders, receivers, forwarders, and UI renderers) in email security verification.

Weak Links among Multi-services. Different email services usually have different configurations and implementations. Some services (e.g., Gmail, Yandex.com) forbid sending emails with ambiguous headers but receive them with tolerance. Conversely, some (e.g., Zoho, Yahoo) tend to allow the sending of emails with an ambiguous header, but conduct very strict checks in the email receiving verification stage. The differences among security policies allow attackers to send spoofing emails from a service with a tolerant sending policy to a service with a loose receiving strategy.

Besides, some email providers deviate from RFC specifications while dealing with emails with ambiguous headers. When MUA handles with multiple `From` headers, some services (e.g., Outlook,Mail.ru) display the first header, while others (e.g., iCloud, yandex.com) display the last header.

Moreover, different vendors support Unicode characters to various degrees. Some vendors (e.g., 21cn.com, Coremail) have been aware of the new security challenges caused by Unicode characters, but some (e.g., 163.com, yeah.net) have no knowledge. Particularly, some (e.g., zoho.com, EwoMail) even have not yet supported Unicode characters' rendering.

Finally, only a few email providers show visual UI notification to alert users of spoofing emails and only 12 vendors implement sender inconsistency checks. In particular, the sender inconsistency checks in practice are significantly diverse because of the absence of a unified implementation standard. The lack of an effective and reasonable email security notification mechanism is also one reason why email spoofing has been repeatedly prohibited, but never eliminated.

6.2 Mitigation

This subsection discusses the key mitigating measures. Since email spoofing is a complex problem involving multiple parties, multi-party collaboration is required to counter the relevant issues.

More Accurate Standard. Note that email providers may fail to offer a secure and reliable email service with ambiguous definitions in email protocols. Thus, providing more accurate email protocol descriptions is necessary to eliminate inconsistencies in the practice of multi-party protocols. For example, the DKIM standard should specify when a DKIM signature should be added to forwarded emails. It is reasonable for forwarders to add DKIM signatures to improve the credibility of emails; however, they should not add DKIM signatures to emails that have never passed DKIM verification.

UI Notification. Email UI rendering is a significant part that affects the users' perception of an email's authenticity. Unfortunately, most of webmails and email clients in our experiments only show the `From` header without any more authentication details. Therefore, it is difficult for ordinary users to judge the authenticity of emails.

Additionally, some visual attacks (e.g., A_{12}, A_{13}) can not be defended at the protocol level. An effective defense method is to provide a user-friendly UI notification and alerts users that their received emails may be spoofing emails. Hu et al. [20] also demonstrate that a good visual security notification has a positive effect on mitigating phishing email threats in the real world. As shown in Figure 4, the spoofing email in Section 5 can be verified by all the three email protocols. Nevertheless, users can not distinguish this spoofing email from normal emails without a UI notification.

As shown in Figure 16, users intuitively can recognize whether the received email contains malicious behaviors, based on the UI notification. Coremail, a well-known email service provider in China, has adopted our suggestions and implemented the UI notification on its webmail and email client. In addition, we have released the UI notification scheme in the form of a chrome extension for Gmail called "NoSpoofing"[1].

Evaluation Tools. We have released our testing tool publicly on GitHub [2] for email administrators to evaluate and increase their security. After configuring the target email system information, the tool can interact with the target system and evaluate whether the target system is vulnerable to the attacks.

7 Disclosure and Response

Vulnerabilities found in this work have already been reported to all 30 relevant email vendors in detail. We have been contacting these entities to help them mitigate the detected threats. Our contact results are summarized as follows.

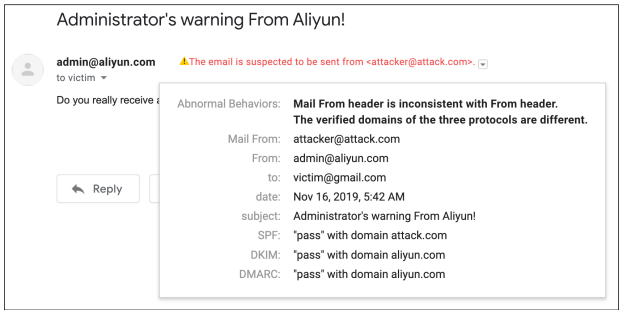


Figure 16: An example of UI notification against the combined attack

Alibaba Cloud: They are interested in the attacks and have an in-depth discussion with us about the specifications. They mention that RFC 6376 suggests adding a DKIM signature in the email forwarding stage to increase emails' credibility. They have now recognized the risk of adding DKIM signatures without verification and promise to evaluate and fix such issues. They also suggest we contact the authors of related RFCs to reach an agreed fix proposal.

Gmail: They acknowledge our report and will fix related issues in subsequent updates. They contact us for discussing the essential reasons behind these security issues.

iCloud: They discuss with us about the details of the attacks and their potential consequences. In particular, Apple iCloud Email has already fixed related security issues with our cooperation.

Sina: They evaluate the issue as a high-risk vulnerability and internally assess the corresponding protective measures. As a bonus, they provide us a reward of $\approx$ \$90.

Yandex: They accept our report and confirm the vulnerability. At the same time, they provide a bonus of \$200 for appreciation.

Yahoo: They confirm the vulnerability. But they claim that it is not an immediate risk.

Coremail: They acknowledge our report and particularly thank us for reporting the issue of UI attacks. To counter those security issues, they adopt our suggestions and and start to implement the UI notification to protect users against email spoofing attacks.

QQ Mail and 163.com: They appreciate our work and inform us that they would fix those security issues by anti-spam strategies.

Outlook and Mail.ru: They claim that they are strictly operating their email service in accordance with RFC standards. They categorize these problems as phishing emails and promise to pay more attention to the impact of such attacks.

Others: We have contacted other relevant email vendors and look forward to receiving their feedback.

[1] NoSpoofing : https://chrome.google.com/webstore/detail/nospoofing/ehidaopjcnapdglbbbjgeoagpophfjnp

[2] Email Spoofing Test Tool: https://github.com/mo-xiaoxi/EmailSpoofingTestTool

8 Related Work

Prior works have revealed certain threats of phishing email attacks [8,12], including the impacts of spear phishing attacks on email user's behavior [32]. Our work focuses on more novel forms of spoofing attacks and their influence on the whole authentication process. Poddebniak et al. [37] discuss how practical spoofing attacks break various protections of OpenPGP and S/MIME email signature verification. They also discuss two new protocols that are proposed to enhance spoofing detection, such as BIMI (Brand Indicators for Message Identification) [41] and ARC (Authenticated Received Chain) [3]. However, BIMI is built on DMARC and has not been fully standardized. Thus, the attacks we found are also effective. ARC protocol is standardized in 2019, yet, only three vendors (i.e., Gmail, Office 365, Zoho) have deployed the protocol in our experimental targets. Our work finds that, however, both Office 365 and Zoho have flaws with the implementation of ARC, which can still lead to some security issues .

Hu et al. [20] analyzed how email vendors detect and handle spoofing emails through an end-to-end email spoofing experiment. We find that the vulnerabilities they mentioned have been mostly fixed in the past two years. Besides, they did not discuss bypassing security protocols detection. Our work focuses on new attacks that can bypass security protocols or user-interface protections. We can construct a highly realistic spoofing email that can completely bypass all the email security protocols and user-interface protections.

In addition, prior literature has proposed many techniques to defend traditional phishing attacks. SMTP extensions, such as SPF, DKIM, and DMARC, are designed to protect the authenticity of emails. Foster et al. [14] measured the implementation and deployment of these protocols and pointed out that, unfortunately, despite years of development, the acceptance rate of these security protocols are still not very high. This low acceptance rate seriously jeopardizes the security of the email ecosystem [19].

Besides, there are many works discussing phishing detection methods based on features extracted from email content and headers [7,13,28], lots of which rely on machine learning technology. Furthermore, Ho et al. [18] point out the positive effects of a good security metric against phishing attacks. Other works [21,36] indicates that the current email services does not have a UI Notification as HTTPS [33]. The contemporary visual security indicators are not enough to provide full phishing protection [20,29]. For email spoofing attacks, our research provides a UI notification scheme and evaluation tools for email systems' administrators. It could effectively boost the development of protective measures against email spoofing in the future.

9 Conclusion

This paper explored the vulnerabilities of the chain-based authentication structure in the email ecosystem. Specifically, a failure in any part can break the whole chain under this chain-based structure. Namely, the authenticity of an email depends on the weakest link in the email authentication chain.

We presented a series of new attacks that can bypass SPF, DKIM, DMARC and user-interface protections through a systematic analysis of the email delivery process. In addition, we conducted a large-scale analysis of 30 popular email services and 23 email clients. Experiment results show that all of them are vulnerable to the new attacks, including famous email services, such as Gmail and Outlook. We underscore the unfortunate fact that many email services have not implemented adequate protective measures. Besides, recognizing the limitation of past literature, which focused on spoofing attacks' impacts on a single step of the authentication process, we concentrated on spoofing attacks' influence on the chain-based email authentication process as a whole.

Based on our findings, we analyzed the root causes of these attacks and reported the issues to corresponding email service providers. We also proposed key mitigating measures for email protocol designers and email providers to defend against email spoofing attacks. Our work is devoted to helping the email industry more efficiently protect users against email spoofing attacks and improve the email ecosystem's overall security.

Acknowlegments

We sincerely thank our shepherd Zakir Durumeric and all the anonymous reviewers for their valuable reviews and comments to improve this paper. We also thank Mingming Zhang, Kangdi Cheng, Zhuo Li, Ennan Zheng, and Jianjun Chen for peer-reviewing and assisting in editing this paper.

This work is supported in part by the National Natural Science Foundation of China (Grant No. U1836213 and U1636204), the BNRist Network and Software Security Research Program (Grant No. BNR2019TD01004).

References

[1] Bidirectional text. https://en.wikipedia.org/wiki/Bidirectional_text. Accessed: November 11, 2019.

[2] E Allman, Jon Callas, M Delany, Miles Libbey, J Fenton, and M Thomas. Domainkeys identified mail (dkim) signatures. Technical report, RFC 4871, May, 2007.

[3] Kurt Andersen, Brandon Long, S Jones, and Murray Kucherawy. Authenticated received chain (arc) protocol. *ser. Internet-Draft'17*, 2017.

[4] S Blank and M Kucherawy. The authenticated received chain (arc) protocol. 2019.

[5] Enrico Blanzieri and Anton Bryl. A survey of learning-based techniques of email spam filtering. *Artificial Intelligence Review*, 29(1):63–92, 2008.

[6] Jianjun Chen, Jian Jiang, Haixin Duan, Nicholas Weaver, Tao Wan, and Vern Paxson. Host of troubles: Multiple host ambiguities in http implementations. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1516–1527. ACM, 2016.

[7] Asaf Cidon, Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser, and Alexey Tsitkin. High precision detection of business email compromise. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 1291–1307, 2019.

[8] Dan Conway, Ronnie Taib, Mitch Harris, Kun Yu, Shlomo Berkovsky, and Fang Chen. A qualitative investigation of bank employee experiences of information security and phishing. In *Thirteenth Symposium on Usable Privacy and Security ({SOUPS} 2017)*, pages 115–129, 2017.

[9] D Crocker, T Hansen, and M Kucherawy. Domainkeys identified mail (dkim) signatures (rfc6376). *Internet Society Requests for Comments.(Year: 2011)*, 2011.

[10] Dave Crocker and Paul Overell. Augmented bnf for syntax specifications: Abnf. Technical report, RFC 2234, November, 1997.

[11] Robin Dhamankar, Yoonkyong Lee, AnHai Doan, Alon Halevy, and Pedro Domingos. imap: discovering complex semantic matches between database schemas. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 383–394, 2004.

[12] Christine E Drake, Jonathan J Oliver, and Eugene J Koontz. Anatomy of a phishing email. In *CEAS*. Citeseer, 2004.

[13] Ian Fette, Norman Sadeh, and Anthony Tomasic. Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web*, pages 649–656. ACM, 2007.

[14] Ian D Foster, Jon Larson, Max Masich, Alex C Snoeren, Stefan Savage, and Kirill Levchenko. Security by any other name: On the effectiveness of provider based email security. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 450–464. ACM, 2015.

[15] Ned Freed, Nathaniel Borenstein, et al. Multipurpose internet mail extensions (mime) part one: Format of internet message bodies, rfc2045. *See for instance http://ietf. org/rfc/rfc2045. txt*, 1996.

[16] Evgeniy Gabrilovich and Alex Gontmakher. The homograph attack. *Communications of the ACM*, 45(2):128, 2002.

[17] Markus Gruber, Phillip Wieser, Stefan Nachtnebel, Christian Schanes, and Thomas Grechenig. Extraction of abnf rules from rfcs to enable automated test data generation. In *IFIP International Information Security Conference*, pages 111–124. Springer, 2013.

[18] Grant Ho, Aashish Sharma, Mobin Javed, Vern Paxson, and David Wagner. Detecting credential spearphishing in enterprise settings. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pages 469–485, 2017.

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

[20] Hang Hu and Gang Wang. End-to-end measurements of email spoofing attacks. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 1095–1112, 2018.

[21] Hang Hu and Gang Wang. Revisiting email spoofing attacks. *arXiv preprint arXiv:1801.00853*, 2018.

[22] Tom N Jagatic, Nathaniel A Johnson, Markus Jakobsson, and Filippo Menczer. Social phishing. *Communications of the ACM*, 50(10):94–100, 2007.

[23] Scott Kitterman. Rfc 7208–sender policy framework (spf) for authorizing use of domains in email, version 1, 2014.

[24] Scott Kitterman. Sender policy framework (spf) for authorizing use of domains in email, version 1. 2014.

[25] J Klensin. Simple mail transfer protocol (rfc5321). *Network Working Group, Internet Engineering Task Force. http://tools. ietf. org/html/rfc5321*, 2008.

[26] John Klensin. Rfc 2821: Simple mail transfer protocol. *Request For Comment, Network Working Group*, 2001.

[27] John Klensin, Randy Catoe, and Paul Krumviede. Imap/pop authorize extension for simple challenge/response. In *RFC 2195*. Network Working Group, 1997.

[28] Tim Krause, Rafael Uetz, and Tim Kretschmann. Recognizing email spam from meta data only. In *2019 IEEE Conference on Communications and Network Security (CNS)*, pages 178–186. IEEE, 2019.

[29] Kat Krol, Matthew Moroz, and M Angela Sasse. Don't work. can't work? why it's time to rethink security warnings. In *2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS)*, pages 1–8. IEEE, 2012.

[30] M Kucherawy. Domainkeys identified mail (dkim) and mailing lists. Technical report, RFC 6377, September, 2011.

[31] Murray Kucherawy and Elizabeth Zwicky. Domain-based message authentication, reporting, and conformance (dmarc). 2015.

[32] 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):32, 2019.

[33] Meng Luo, Oleksii Starov, Nima Honarmand, and Nick Nikiforakis. Hindsight: Understanding the evolution of ui vulnerabilities in mobile browsers. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 149–162. ACM, 2017.

[34] DomainKeys Identified Mail. Signatures rfc 6376.

[35] Joshua Pereyda. boofuzz: Network protocol fuzzing for humans. *Accessed: Feb*, 17, 2017.

[36] 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 CHI Conference on Human Factors in Computing Systems*, page 518. ACM, 2019.

[37] 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 *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 549–566, 2018.

[38] Jon Postel. Simple mail transfer protocol. *Information Sciences*, 1982.

[39] Paul Resnick. Rfc2822: Internet message format, 2001.

[40] Paul Resnick. Rfc 5322, internet message format. *Online: https://tools. ietf. org/html/rfc5322*, 2008.

[41] T. Loder S. Blank, P. Goldstein and T. Zink. Brand indicators for message identification (bimi). Technical report, 2019.