Artifact Evaluated NDSS Available Functional

One Email, Many Faces: A Deep Dive into Identity Confusion in Email Aliases

Mengying Wu[†], Geng Hong[†]✉, Jiatao Chen[†], Baojun Liu[‡], Mingxuan Liu[§] and Min Yang[†]✉
[†]Fudan University, China, {wumy21, jtchen24}@m.fudan.edu.cn, {ghong, m_yang}@fudan.edu.cn
[‡]Tsinghua University, China, lbj@tsinghua.edu.cn
[§]Zhongguancun Laboratory, China, liumx@mail.zgclab.edu.cn

***Abstract*—Email addresses serve as a universal identifier for online account management, however, their aliasing mechanisms introduce significant identity confusion between email providers and external platforms. This paper presents the first systematic analysis of the inconsistencies arising from email aliasing, where providers view alias addresses (*e.g., ALICE@example.com*, *alice+work@example.com*) as additional entrances of the base email (*alice@example.com*), while platforms often treat them as distinct identities.**

Through empirical evaluations the alias mechanisms of 28 email providers and 18 online platforms, we reveal critical gaps: (1) Only Gmail fully documents its aliasing rules, while 11 providers silently support undocumented alias behaviors; (2) Due to lack of standardization documentation and de facto implementation, platforms either failed to distinguish alias addresses or over aggressive excluded all emails containing specific symbol. Real-world abuse cases demonstrate attackers exploiting aliases to create up to 139 accounts from a single base email in npm for spam campaigns. Our user study further highlights security risks, showing 31.65% of participants with alias knowledge mistake phishing emails as legitimate emails alias due to inconsistent provider implementations. Users who believe they understand email aliasing, especially those highly educated, male, and technical participants, are more susceptible to being phished. Our findings underscore the urgent need for standardization and transparency in email aliasing. We contribute the OriginMail tool to help platforms resolve alias confusion and disclose vulnerabilities to affected stakeholders.

I. Introduction

Email is one of the most widely adopted methods for identity verification across the world. It is commonly required when registering accounts on online platforms, serving as a unique identifier that links the account to an individual identity. Emails are also used routinely for account management tasks such as activation and recovery. Although the use of phone numbers for authentication has grown in popularity with global connectivity, email remains one of the most universal and stable identifiers for online identity management.

✉ Corresponding authors.

Network and Distributed System Security (NDSS) Symposium 2026
23 - 27 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-8-0
https://dx.doi.org/10.14722/ndss.2026.230148
www.ndss-symposium.org

Email Aliases. To support user privacy and flexible identity management, email providers offer alias mechanisms [1]. Alias mechanisms redirect the email sending to the alternative addresses to the same inbox as the primary email. They aim to help users leverage a single email account to separate different activities, such as work or gaming. For example, *alice+work@gmail.com* and *alice+game@gmail.com* are aliases of *alice@gmail.com*.

As email addresses are widely used as platform identifiers for authentication, access control, and resource allocation, this creates a growing mismatch: *while email providers treat alias addresses as one identity, platforms typically treat them as separate users.* As shown in Figure 1, this inconsistency leads to two key risks. On the one hand, platforms may unknowingly allow an email account to register enormous account creation with alias, dubbed as "Alias Multiplicity Abuse"(AMulA), since they cannot distinguish aliases from real, distinct users. An abuser may repeatedly register new alias-based accounts to continuously exploit free trial offers, thereby gaining unlimited access to premium features without payment. On the other hand, users may misunderstand aliasing and mistakenly associate visually similar addresses as belonging to the same alias set, when in fact they correspond to distinct entity identities. This misunderstanding elevates the risk of phishing and spoofing attacks, dubbed as "Alias Misidentification Attack"(AMisA).

Research Gap. Security risks arising from inconsistencies in email system design have been extensively studied, such as mismatches between different SMTP header fields [2], delegation mechanisms [3], and inconsistency between web interfaces and email clients in sender display [4]. However, they overlook a different kind of inconsistency outside: the identity confusion caused by email aliasing mechanisms. Specifically, *there is a disconnect between how email providers interpret alias addresses and how external platforms (such as GitHub, Facebook) and users understand them, who use email addresses as user identifiers*. More critically, due to inconsistent and non-transparent implementations by email providers, it is challenging for platforms to identify alias accounts.

Our work. In this paper, we performed the first systematic analysis of identity confusion caused by email aliasing mechanism inconsistency between email providers and online platforms. This study is guided by the following research

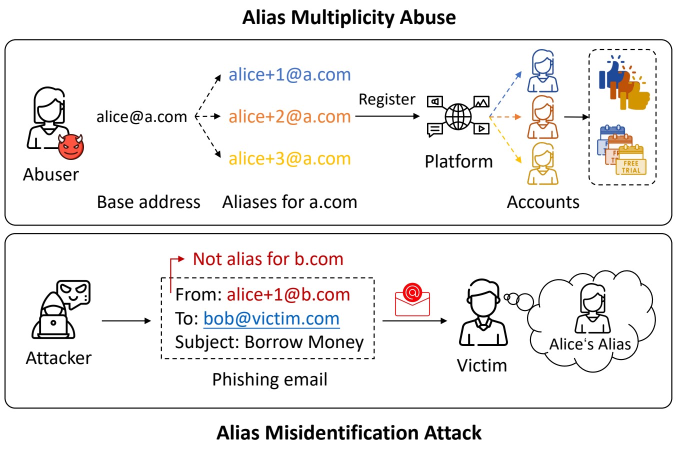


Fig. 1: The attack model. Alias Multiplicity Abuse allows abusers to create unlimited accounts from a single base email address, potentially exploiting resources such as free trials. Alias Misidentification Attack involves mimicking valid email aliases with non-alias addresses to trick victims.

questions:

RQ1 How do email providers document their aliasing mechanisms, and how do these compare to the actual aliasing behaviors?

RQ2 How do online platforms interpret and handle email aliases, and do their practices align with those of email providers?

RQ3 How can adversaries abuse email aliasing mechanisms in real-world attacks, and what countermeasures can mitigate these risks?

Identity in email providers. We examine the alias documentation and implementation of 28 providers. Our findings reveal an inconsistency between the email protocol and how providers implement it. All tested providers treat case variations in usernames (e.g., *ALICE@*, *alice@*) as the same address, though SMTP technically allows case-sensitive usernames. Besides, 12 providers support email alias except for case variation, while only Gmail fully documented its aliasing mechanism. We discovered that eight providers support aliasing without documentation. For example, Eclipso [5] supports a special prefix-based aliasing scheme using 12 special characters without documentation. There are also three providers incompletely documented their aliasing mechanisms, for example, Yandex [6] mentioned its infix alias but omitted its suffix-based alias.

Email as identity in platforms. To evaluate how platforms recognize identity based on email address, we conducted controlled experiments on 18 popular platforms that use email addresses as identifiers during account registration. We developed a semi-automated testing framework that simulates account registration using alias addresses and verified whether these websites accept, reject, or identify alias addresses as equivalent to their base address identities.

Overall, none of the tested platforms were able to fully defend against alias-based account creation. We only found five platforms that performed alias detection. Cloudflare [7] employs an overly strict sanitizing mechanism that invalidates all email addresses containing the plus symbol (+), even those with legitimate syntax. The other four platforms (Facebook, Instagram, TikTok, and Zoom) attempt to detect aliases, but their defenses rely on ad hoc, provider-specific rules. For instance, TikTok only recognizes Gmail's plus-suffix aliases. While some platforms enforce strict character-level constraints on email formats, 9 platforms failed to defend against aliasing from any provider. More concerning, we observed a contradiction between platforms email identity and protocol. npm [8] and PyPI [9] treat email addresses as case-sensitive both in the username and the domain, whereas SMTP requires the domain part to be case-insensitive.

Alias Multiplicity Abuse in the Wild. The lack of effective identification alias-based registrations opens the door to Alias Multiplicity Abuse. Thus, we build *OriginMail*, a tool that detects and normalizes aliases to base address based on aliasing rules from the 28 email providers. We examined the use of alias emails from npm and GitHub, where user email addresses are publicly accessible, and identified 1,062 base addresses that have multiple npm or GitHub accounts. We found that an attacker used a single address with aliases to create up to 139 accounts and publish 3,904 packages, which were then leveraged for black-hat SEO campaigns on npm, as described in [10]. We have reported our findings to the affected platforms, including GitHub, Cloudflare, and Adobe, and received their acknowledgement.

Alias Misidentification Attack Risks. Users may be tricked by *AMisA* when mistakenly trusting a phishing email address as the alias. For example, mistake *alice+1@b.com* as the alias of *alice@b.com*, while *b.com* does not support that. To assess users' understanding of email aliasing, we conducted a user study (N=304) in which participants were asked to determine whether a variant email could be trusted as a known address in the contact. Our result shows that 29.89% users have little familiarity with alias mechanisms; they rejected any address that visually differed from the original as untrustworthy. However, while 45.40% of users self-reported awareness of email aliasing mechanisms, 22.78% of them failed to correctly identify the Gmail alias syntax (*e.g., username+alias@gmail.com*), demonstrating a gap between subjective awareness and objective comprehension.

Lacking standardized alias syntax elevates phishing vulnerability. Users who believe they understand email aliasing, especially those highly educated, male, and technical participants, are more susceptible to being phished, with the overall susceptibility rate rising to 31.65%. Notably, CS students' AMisA susceptibility rate increased from 0% without alias awareness to 35.29% with it, driven by overgeneralization across providers.

Contributions. This paper makes the following contributions:

• We conduct the first systematic analysis of the email aliasing mechanism, and uncover the identity confusion brought by the inconsistencies between email providers, online platforms, and ordinary users.

• We examined the alias implementation of 28 popular email providers, and found that only Gmail fully documented its alias mechanism, while others lack transparency and consistency.

• We found that due to inadequate alias handling, platforms mistakenly treat alias variants as distinct users, allowing a single base address to generate up to 139 accounts in npm.

• Our user study demonstrates that the lack of standardized alias syntax elevates phishing vulnerability, particularly for users with partial knowledge, especially those with high education and technique background.

• For mitigating identity confusion, we disclosed our findings to relevant platforms and open-sourced *OriginMail* [11] to help users and platforms identify the actual mail from aliases.

II. Motivation

Email Alias. To protect user privacy and provide flexible identity management, email providers offer an alias mechanism. These aliases are alternative addresses that redirect to the same inbox as the base email address. These aliases help users separate different activities, such as work or gaming. For example, one can register a game account with **alias address** *alice+game@gmail.com* while the **base address** *alice@gmail.com* will receive the game emails, without any setting in Gmail. The core idea behind email aliases is to provide users with the ability to manage multiple identities without exposing their base email addresses. However, the flexibility provided by email aliases also introduce potential risks and challenges, particularly when these aliases are used across different platforms and services. We summarize two attack models of identity confusion due to alias in Figure 1.

Alias Multiplicity Abuse. Alias Multiplicity Abuse (AMulA) refers to the abuse email aliases of the same base email address to register multiple accounts using variations. When online platforms fail to recognize that aliases point to the same underlying account, they treat alias variants as distinct identities. While some email providers require specific configurations to use alias addresses, many others allow users to create unlimited aliases simply by following certain symbol rules.

Table I shows a real-world case of npm. npm (Node Package Manager) is a widely used package manager for JavaScript, primarily designed to help developers easily manage and share reusable code packages within the Node.js ecosystem. An abuser created seven different accounts by adding dots at different positions in the same Gmail address (*e.g., julayera@gmail.com*), each treated as a separate alias. For Gmail, no special configuration is needed, any email sent to these variations will be directed to the main inbox. However, npm registers these aliases as seven distinct email addresses, resulting in the creation of seven separate accounts. The abuse accounts published 117 spam packages in npm within only two days, which were taken down by npm after four months.

Alias Misidentification Attack. Alias Misidentification Attack occurs when attackers exploit users' assumptions about

Email Address	Username
j.ulayera@gmail.com	bujalsokao
ju.layera@gmail.com	nuilaopmei
jul.ayera@gmail.com	nualosomuina
jula.yera@gmail.com	ikapikangsua
julay.era@gmail.com	nikakulpaliindi
julaye.ra@gmail.com	limaospoiukas
julayer.a@gmail.com	ukariklaopsiwa

TABLE I: Seven alias accounts registered with one base email address (*julayera@gmail.com*) in npm.

alias formats across domains. Due to the complexity and inconsistency of alias mechanisms, this leads developers or users, who are familiar with alias, to misjudge an address's legitimacy. For example, in Figure 1, an attacker send a phishing email with sender *alice+1@b.com* to Bob, where plus-suffix is not a valid alias for *b.com*. Bob, who knows that *alice+1@a.com* is a valid alias, may misidentify and assume *alice+1@b.com* is also Alice's alias, then fall into phishing attacks. The attack is practical because email providers allow different users to register visually similar email addresses, as long as they align with syntactic rules.

III. Identity in Email Providers

In this section, we learn about the identity recognition mechanism of email providers for email addresses. Firstly, we will learn the public email alias mechanism of the email providers through their official documentation. Subsequently, we use a semi-automated alias testing framework to determine the email alias mechanism actually used by these email providers.

A. Alias Documentations

We first examined how different email providers define and implement aliases. To ensure broad coverage and minimize biases arising from regional legal and policy differences, we searched Google for the most widely used email providers globally, ensuring representation across major markets, resulting in a total of 40 providers.

We conducted an extensive review of email providers' official documentation on aliasing mechanisms, identifying the officially recognized rules for creating alias sender emails. Specifically, for each provider, we searched using keywords such as "alias" and "backup" to locate any available information on aliasing policies. In total, we collected alias-related documentation from 20 email providers. After analyzing the documentation, we found that email providers generally implement aliasing through two methods.

Syntactic Aliases. Syntactic alias addresses are derived by modifying the syntax of base address while still being routed to the same inbox. It is used by Gmail [1, 12, 13], 2925Mail [14], Yandex Mail [15], Yahoo Mail [16], and ProtonMail [17]. The modifications mentioned in documentation include: 1) adding characters within or at the end of the username (*e.g., test@gmail.com* has variants of *te.st@gmail.com* or *test+1@gmail.com*), and 2) using different domain suffixes within the same provider, for example, Yandex

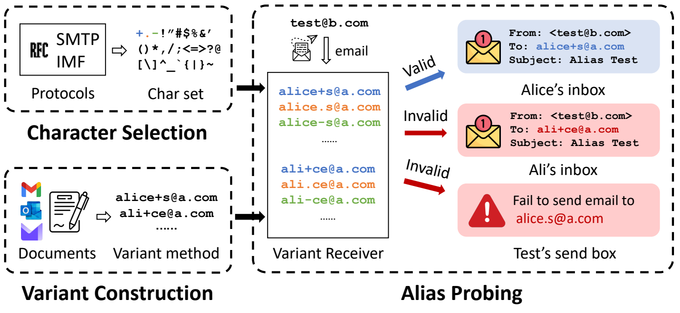


Fig. 2: Actual email aliases probing experiment.

allows *test@yandex.com* to be changed to *test@yandex.ru* or *test@yandex.by*. These aliases typically require no additional setup by users, as email providers handle them automatically by ignoring certain characters or recognizing their own domain variations, which simplifies alias management. As a result, most providers have no restrictions on alias numbers.

Customized Aliases. This allows users to explicitly create alias addresses that may be structurally unrelated to the base address, like *base@aliyun.com* and *variant@aliyun.com*. 16 of 28 email providers offer this approach as alias address, including Zoho Mail [18], 139 Mail [19], and Alibaba Mail [20]. Custom aliases pose less risk of identity confusion than syntactic aliases as they require manual setup and are limited in number (ranging from 1 to 30 aliases per account), making large-scale abuse more difficult.

In this study, we focus on syntactic aliasing mechanisms, as they usually allow users to generate an unlimited number of email variations and highlight the security risks posed by alias-based identity confusion.

B. Alias Implementations

After examining the aliasing mechanisms documented by the service provider, we further try to confirm whether the implementation aligned with their claims and to identify any undocumented behaviors.

Figure 2 shows the roadmap of the experiment. For each provider, we register a base address and generate a set of variant addresses that mimic possible alias forms, covering three perspectives of address transformation. We then attempted to send emails to these variants. By observing whether the emails are successfully delivered to the base inbox, we infer the provider's internal logic for identifying alias addresses as belonging to the same email account.

1) Alias Testing Framework: We used a string composed of lowercase letters and digits as the username for all address registrations. For the 40 providers we get in Section III-A, we successfully registered free email accounts in 28 email providers. The remaining providers failed because of paid-only services, restricted access, or inaccessible email functions.

Character Selection. In order to determine the characters we can test and reduce unnecessary requests, we analyzed the character specifications in email protocols. RFC 5321 (Simple Mail Transfer Protocol) [21] prohibits email addresses that rely on non-ASCII characters or ASCII control characters (ASCII code points 0–31 and 127). Special characters may be used if they properly escape using a backslash. SMTP emphasizes that the local-part of a mailbox *MUST BE* treated as case sensitive, while mailbox domains follow normal DNS rules and are hence not case sensitive. RFC 5322 (Internet Message Format) [22] governs the structure of email headers such as `From:`, thus allows the printable ASCII code except colons (:). According to the protocols, we limited our character set to 32 printable symbols (Figure 2) and alphanumeric (letters and numbers) for generating variants.

Variant Construction. Based on the transformation of aliasing mechanisms explicitly stated by certain email providers in Section III-A, we categorize our email construction into the following five patterns, as shown in Figure 3.

- Prefix Addition: Adding alphanumeric and symbols before the original username (*e.g., p+alice@a.com*).
- Infix Insertion: Inserting characters within the original username, such as between existing letters (*e.g., ali+ce@a.com*).
- Suffix Addition: Appending characters, including symbols after the original username (*e.g., alice+s@a.com*).
- Case Variation: Modifying the capitalization of characters in the username (*e.g., Alice@a.com*).
- Domain Substitution: Replacing the domain with another domain supported by the same provider (*e.g.,* switching from *@yandex.com* to *@yandex.ru*), typically allowed when the provider offers multiple interchangeable domains.

Alias Probing. To minimize bias introduced by third-party email-sending mechanisms (*e.g.,* authentication steps or domain-based spam filtering), we set up our own SMTP server to send test emails. We also constructed the mail in raw MIME format to avoid potential errors caused by character escaping in high-level programming languages. We created a pool of meaningful, non-spammy email content to reduce the risk of

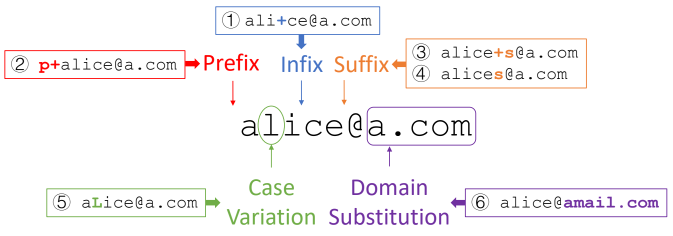


Fig. 3: Variant construction methods, with five operations: infix insertion, prefix addition, suffix addition, case variation, and domain substitution.

TABLE II: Email providers offer more than one domain. Beyond these listed, Runbox has 37 domains in total.

Provider	Email Domains
Gmail	gmail.com, googlemail.com
Yandex	yandex.com, yandex.ru, yandex.by, yandex.kz, ya.ru
GMX	gmx.com, gmx.ur, gmx.co.uk, gmx.ca
Proton	protonmail.com, pm.me, proton.me
Runbox	runbox.com, mailhost.work, rbx.email, runbox.eu, ...

emails being filtered or rejected as spam, all content have been verified by two spam-detection tools [23, 24]. We randomly select one during testing within the same provider.

For each successfully registered base address, we first validated its functionality by sending a test message. Upon successful delivery, we proceeded to generate a set of variant addresses deformed from the base address and conducted remaining tests. If the base address received the email we sent to variant address, we confirm that it is a valid alias address.

Ethical Considerations. We took multiple steps to ensure the ethical conduct of our study. All email addresses used were self-controlled and designed to avoid affecting real users. To reduce server impact, email-sending rates were strictly limited. We also mitigated the risk of misdirected messages by using long randomized usernames and clearly labeling all emails as part of a research study with opt-out instructions. Further details are provided in Section X.

2) Result and Analysis: We summarize alias implementation across all 28 tested providers in Table III. All providers successfully received at least one mail sent to a variant address, primarily because all providers treat the username as case-insensitive, while the SMTP specifies it must be case-sensitive. Meanwhile, most providers ignore the backslash(\) during address interpretation. Detailed unsupported characters are listed in Appendix A-A2. Excluding these two factors, 12 providers still support additional aliasing behaviors.

Suffix Addition Alias. 11 providers, including Alibaba Mail, Zoho Mail, Runbox Mail, supported suffix-based aliasing by appending strings to the username. They used a plus (+) as a separator and allowed arbitrary alphanumeric strings after it, similar to Gmail's aliasing convention, *i.e., alice+1@aliyun.com* is an alias of *alice@aliyun.com*.

2925Mail was unique in that it accepted with all tested characters (except for known invalid ones). Enabling by restricting base addresses to 9–12 characters, without a separator, 2925Mail allowed even raw alphanumeric suffixes to be interpreted as aliases, like *abcdefghi123@2925.com* to alias *abcdefghi@2925.com*. This makes it particularly permissive and potentially exploitable for bypassing identity checks or spam filtering mechanisms that rely on strict email matching. Additionally, the lack of a clear alias delimiter makes it harder for recipients or systems to recognize the address as an alias of an existing one.

Special Prefix Alias. Eclipso [5] represented another special case, as it allowed the addition of prefix characters to the username. When a separator, such as `!#$%*/?^{|}~`, was added before the username, Eclipso would parse only the part after the separator as the valid mailbox identifier. In effect, addresses like *prefix!alice@eclipso.eu* would be delivered to *alice@eclipso.eu*, with the prefix being ignored during recipient resolution. This feature introduces a significant authentication risk: A recipient receiving mail from *alice#bob@eclipso.eu* may misinterpret the sender's identity as *alice@eclipso.eu*, while the message actually originates from *bob@eclipso.eu*. This ambiguity could facilitate social engineering attacks, exploiting users unfamiliar with aliasing mechanisms. Notably, as Eclipso does not accept the plus(+) as a separator, it remains orthogonal to suffix-based aliasing schemes.

Infix Insertation. Only three providers support adding characters within the username: 2925Mail supports %, Gmail supports dot(`.`), and ProtonMail [30] supports four characters (`._-/`) inside its username. Unlike almost all suffix aliases use the same plus (+) to separate suffixes, these three providers adopted different characters, increasing ambiguity. That is, *user-name@proton.me* is an alias of *username@proton.me*, while *user-name@gmail.com* has a different identity with *username@gmail.com*. This inconsistency poses challenges for systems that rely on email addresses for identity recognition, as they may struggle to distinguish legitimate aliases from unrelated accounts.

Finding I: *Unlike the consistent use of + in suffix aliases, infix alias mechanisms vary across each provider (i.e., Gmail (.), Proton (._-/), and 2925Mail (%), making users hard to consistently normalize infix alias.*

Domain Substitution. Despite documentation suggesting support for domain substitution (Table II), our tests only successfully deliver emails to Runbox, Gmail, and Yandex when alternative domains are used. Among the confirmed cases, Runbox is particularly notable, supporting up to 37 alternative domains as aliases, such as *@rbx.email*, *@runbox.eu*, and *@mailhost.work*. Also, it allows users to select a preferred domain during registration and automatically treats the remaining domains as aliases.

C. Inconsistency Between Document and Implementation

When comparing the actual aliasing behaviors with official documentation, we found that public disclosure of alias mechanisms is often insufficient or entirely missing.

A surprising finding is an inconsistency between protocol-level semantics and provider-level implementation. All tested providers silently treat case variations (*e.g., ALICE@a.com* vs *AliCe@a.com*) as the same base address, delivering messages to the same inbox. However, none of the providers documented this in their official documentation. This complete lack of disclosure is particularly notable because case sensitivity in email usernames is technically allowed under the SMTP, yet in practice, every provider silently overrides this specification by enforcing case-insensitive behavior. Developers and security systems that rely on strict interpretations of email identity may misjudge the uniqueness of case aliases, potentially leading to authentication bypasses.

Finding II: *Despite SMTP allowing case-sensitive usernames, all tested providers silently enforce case-insensitive handling, i.e., ALICE@a.com, alice@a.com are the same identity.*

Besides the case alias, among the 28 providers, only Gmail fully documented its aliasing mechanism. Gmail clearly states support for three aliasing schemes: dot-infix (insertion of dot)

TABLE III: Summary of alias implementation of email providers, with the base email address to be test@domain.com. Beside these 12 providers, another 15 providers also treat the username as case-insensitive, including: 163Mail, 126Mail, Yeah, QQMail, 139Mail, Sina, SohuMail, 2980Mail, Exmail.qq, Foxmail, Gmx, Yahoo, Myyahoo, Tuta, Mail.com, and Onet.

Provider	**Prefix Addition**	**Infix Insertion**	**Suffix Addition**	**Case Variation**	**Domain Substitution**	**Example**
Alibaba Mail [20]	-	-	Plus(+)	Insensitive	-	test+t@aliyun.com Test@aliyun.com
Mail.ru [25]	-	-	Plus(+)	Insensitive	-	test+t@mail.ru Test@mail.ru
Zoho [18]	-	-	Plus(+)	Insensitive	-	Test@zohomail.com test+t@zohomail.com
Outlook [26]	-	-	Plus(+)	Insensitive	-	test+t@outlook.com Test@outlook.com
Hotmail [26]	-	-	Plus(+)	Insensitive	-	test+t@hotmail.com Test@hotmail.com
iCloud [27]	-	-	Plus(+)	Insensitive	-	test+t@icloud.com Test@icloud.com
Eclipso [5]	!#$%*/?^{`\}~	-	-	Insensitive	-	t!test@eclipso.eu Test@eclipso.eu
2925 [28]	-	Percent(%)	Add any suffix	Insensitive	-	te%st@2925.com test-t@2925.com Test@2925.com
Gmail [29]	-	Dot(.)	Plus(+)	Insensitive	googlemail.com	te.st@gmail.com test+t@gmail.com Test@gmail.com test@googlemail.com

Protonmail [30]	-	Dot(.) Hyphen(-)
		Underscore(_) Slash(/)
			Plus(+)	Insensitive	-	te.st@protonmail.com
						test+t@protonmail.com
						Test@protonmail.com

Runbox [31]	-	-	Plus(+)	Insensitive	mailhost.work rbx.email runbox.eu runbox.me	test+t@runbox.com Test@runbox.com test@runbox.me
Yandex [6]	-	-	Plus(+)	Insensitive	yandex.ru yandex.by yandex.kz ya.ru	test+t@yandex.com Test@yandex.com test@ya.ru

alias, plus-suffix aliases, and domain substitution between *@gmail.com* and *@googlemail.com*. In addition, it allows users to configure up to 99 custom alias addresses via settings.

We discovered that eight providers silently support aliasing, despite offering no documentation mentioning such functionality. Providers like Alibaba Mail [20], Zoho [18], Outlook [26], Runbox [31], iCloud [27], Mail.ru [25], and Hotmail [26] all accept plus-suffix aliases, yet make no official mention of this behavior. Eclipso goes even further, supporting a rare prefix-based aliasing scheme using 13 special characters.

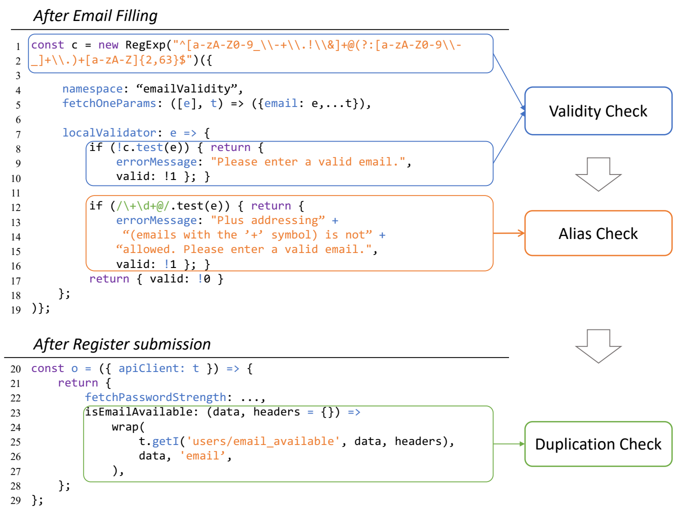
In contrast to providers offering no documentation at all, three providers partially documented their aliasing mechanisms, but omitted key behaviors we observed in practice. 2925Mail states that users can create aliases by appending letters, numbers, or underscores to their username. However, our tests show that all 32 printable symbols are accepted, and it also supports inserting % inside the username, which is a behavior not mentioned anywhere on its help pages. Yandex's documentation explains that users can receive mail sent to alternate four domains and infix of three symbols in usernames are treated as aliases. However, the widely-used plus-suffix aliasing feature is not mentioned. Conversely, ProtonMail documents support for plus-suffix aliases but makes no reference to the fact that users can include `._/-` within usernames to form valid alias addresses.

Finding III: *Most providers fail to clearly document their aliasing behaviors, with 8 silently supporting aliases without any disclosure, and 3 providing incomplete documentation. This lack of transparency leave users and systems from reliably identifying which email addresses are treated as aliases.*

Our findings reveal a severe inconsistency between the actual aliasing behaviors of email providers and their official documentation. This lack of transparency has important security and usability implications. For users, undocumented aliasing behaviors may lead to confusion or misconfiguration when setting up filters, managing identities, or registering on third-party platforms. For developers and service providers, it increases the risk of treating distinct aliases as independent accounts, potentially enabling bypass of duplicate account checks, phishing, or spam evasion. Furthermore, inconsistent documentation hinders efforts to build robust identity validation mechanisms, especially when every provider has inconsistent alias mechanisms.

IV. Email as Identity in Platforms

After uncovering the aliasing mechanisms of email providers, we try to examine how email-based identity platforms handle email aliasing in practice.



```
const c = new RegExp("^[a-zA-Z0-9_\\-+\\.!\\&]+@(?:[a-zA-Z0-9\\-
_]+\\.)+[a-zA-Z]{2,63}$")({

    namespace: “emailValidity”,
    fetchOneParams: ([e], t) => ({email: e,...t}),

    localValidator: e => {
        if (!c.test(e)) { return {
            errorMessage: "Please enter a valid email.",
            valid: !1 }; }

        if (/\+\d+@/.test(e)) { return {
            errorMessage: "Plus addressing” +
             “(emails with the ’+’ symbol) is not” +
            “allowed. Please enter a valid email.",
            valid: !1 }; }
        return { valid: !0 }
    };
)};
```

```
const o = ({ apiClient: t }) => {
    return {
        fetchPasswordStrength: ...,
        isEmailAvailable: (data, headers = {}) =>
            wrap(
                t.getI('users/email_available', data, headers),
                data, 'email’,
            ),
    };
};
```

Fig. 4: Email verifiers of X.com in registration.

We targeted online platforms that require email-based registration, as these systems commonly treat email addresses as primary user identifiers, making them highly sensitive to aliasing-related confusion. Several platforms, including GitHub [32] and Cloudflare [33], explicitly acknowledged in our communications that the use of aliases to create multiple accounts constitutes abuse. Motivated by this, we developed a semi-automated framework to simulate registrations with various variant addresses, checking whether they are accepted, rejected, or recognized as equivalent to the base address.

A. Email Verifier in Registration

To understand how platforms handle email aliases, we first aim to model their email verification process during user registration. Specifically, our analysis monitors all interactions between the user and the registration interface. On the frontend, we inspect the source code and reverse-engineer JavaScript to identify validation logic and timing. On the backend, we examine API responses and error messages to infer server-side checks.

Email verification typically occurs in two phases: email filling and registration submission. As shown in Figure 4, we found that platforms typically apply a three-step verification process in sequence before accepting an email address for account creation, distributed in the two phases. Some platforms perform early checks (*e.g.,* validity) during email filling, while others defer all checks until submission.

Validity check. Validity ensures that the email address follows standard email formatting rules, especially that the username and domain do not contain disallowed characters. For example, Microsoft's validation allows letters, digits, dots (.), underscores (_), and hyphens (-), but disallows continuous dots, *i.e., al..ice@example.com.*

Alias check. Platforms may use specific heuristics to determine whether the submitted address is a known alias. Note that not all platforms have alias checks, we infer their presence based on specific error prompts. For example, in the frontend code of X [34] in Figure 4, we observed that it allowed characters such as English letters, digits, and seven symbols, including “+”, during validity checks. However, in a subsequent check, X specifically checked for the presence of a plus(+) to identify alias email addresses.

Duplication check. Duplication check verifies whether the email has already been used to register an account. Some platforms also check the validity of their email while confirming whether it is occupied, while may not be consistent with the validity check. For example, X's duplication check permits a broader set of 21 symbols compared to its validity check of 7 symbols. This may help bypass the validity and alias checks to register an account.

B. Registration Test

To understand how platforms that rely on email addresses as user identifiers respond to aliasing mechanisms, we designed a semi-automated testing framework focused on the account registration process, confirming whether platforms accept or recognize email aliases.

A key challenge is that platforms often implement multi-step email verification, which is not encapsulated within a single API call. Our insight is that platform designs are typically user-centric: platforms are designed to provide immediate feedback when a user completes the email input or submits the registration form. Such feedback usually appears as front-end changes, *e.g.,* error prompts or page transitions. Thus our framework monitors any DOM changes after two critical user actions: when the user finishes entering an email address, and when the user submits the registration form. If a visible change in the HTML element, especially an error message, is detected after these actions, it suggests that the platform rejected the email. Conversely, if no change occurs or the page proceeds to the next page (URL), the email is likely accepted.

Our implementation builds on DrissionPage [35], a browser automation tool that helps bypass bot detection. Given a target platform and a variant address, the framework opens the registration page and locates the email input field by identifying nearby labels or placeholder text containing “email”, and performs both checks. This design enables scalable testing without account creation, minimizing disruption to platforms.

We tested the registration process on the Tranco [36] Top 100 domains, identifying 18 platforms that allow users to sign up using only an email address. For each platform, we first registered base accounts using the base address from the 28 email providers, and recorded their registration URLs. We then tested whether the variant addresses could be used to register new accounts, following the alias generation methods outlined in Section III-B. We took careful measures to avoid creating real accounts and controlled the testing frequency, detailed measures are discussed in Section X.

C. Alias Defense Strategies

In all scenarios where variant addresses are rejected, we observe two defense strategies: explicit alias detection, which blocks specific alias patterns, and implicit character restrictions, which filter out all potential aliases by enforcing strict constraints on email formats.

1) Explicit Alias Detection: Based on how platforms handle specific alias patterns, we found that two strategies: provider-independent detection, where platforms apply general rules such as case normalization, and provider-specific detection, where platforms explicitly recognize aliasing formats used by particular email providers (*e.g.,* Gmail dot or plus aliases).

Provider-independent Alias Check. Among the 18 tested platforms, 16 platforms consistently detected and blocked registration attempts using case variation variant addresses. In contrast, npm [8] and PyPI [9] were case-sensitive for both the local-part and the domain, treating *Alice@example.com* and *alice@EXAMPLE.com* as distinct identities. This behavior contradicts from the SMTP standard, which defines domain names as case-insensitive, and increases further opportunities for alias-based abuse.

Besides case variation, only Cloudflare plays a provider-independent alias detection. It rejects email addresses with plus-suffixes regardless of whether the original provider supports this aliasing behavior. However, it is an over-aggressive detection, as no standards inform that all addresses with plus(+) are aliases.

Provider-specific Alias Check. We found five platforms that support provider-specific alias detection. Cloudflare, Facebook [38], Tiktok [46], and Zoom [45] correctly detected Gmail's dot-insertion aliasing and rejected attempts to register dot alias of an existing Gmail address. For plus-suffix aliases, Facebook and Instagram [39] recognized and blocked aliases from Outlook, Gmail, Hotmail, and Mail.ru, while TikTok only handled plus-suffix aliases from Gmail. These findings indicate that only a handful of platforms implement alias detection logic, and even those implementations are limited to specific providers and formats.

In general, these platforms adopt conservative strategies, addressing only a small subset of known aliasing rules. As none of the tested platforms allowed an alias to be used to log into an existing base-email account, they avoid potential account takeover risks.

Finding IV: *Only 5 out of 17 platforms have alias detection during email registration, and they can only defend specific email providers, i.e., Gmail, Outlook, Hotmail, and Mail.ru. This narrow scope leaves platforms exposed to alias-based abuse from other providers.*

2) Implicit Character Defense: While platforms may not explicitly address aliasing, many inadvertently limit its impact by restricting the use of certain characters or email domains during registration.

Symbol Sanitizer. Most platforms adopt character-level sanitization when validating email addresses, inadvertently limiting alias usage. However, the accepted symbols vary widely.

We found three platforms impose notably stricter restrictions on acceptable symbols. Microsoft only allows the use of hyphen (-), dot (.), and underscore (_) in email usernames. TikTok additionally allows the plus (+) character. Unity further expands the accepted set by including percent (%).

For the other platforms, among the 32 symbols we tested, 20 symbols were commonly accepted by most platforms, which we called *popular symbols*, and seven symbols were universally rejected. For example, Cloudflare, Pinterest, Zoom, Vimeo, and Spotify only accept the *popular symbols*, while Gandi [47] rejects the slash (/) despite accepting the rest.

Domain-level Restrictions. We also observed email domain-level restrictions in four platforms. Adobe explicitly blocks *runbox.com*, reporting "This email address is not allowed." Similarly, Vimeo rejects both *sina.com* and *qq.com* with the message "Please enter a valid email address." X.com blocks more domains, *sina.com*, *aliyun.com*, *sohu.com*, and *2925.com* are rejected to deliver the confirmation emails.

D. Identity Inconsistency between Provider and Platform

By conducting registration experiments using variant email addresses across different platforms, and aligning these results with the actual aliasing mechanisms supported by email providers, we identified which alias emails were able to successfully register multiple accounts. We summarized the result in Table IV. Overall, no platform was able to fully handle the aliasing mechanisms of all 12 tested email providers.

Among the 18 platforms we tested, Microsoft and Cloudflare exhibited the least impact from the inconsistent identity recognition between email providers and platform registration. Specifically, Microsoft, due to its strict symbol restrictions (allowing only three symbols), was able to block aliases from most email providers, with only three providers bypassed and could use for register multiple accounts with aliases. Cloudflare, having handled all plus-suffix alias variations and Gmail's unique rules, managed to filter most alias email registrations, with only 2925Mail and ProtonMail successfully bypassing its alias checks. However, 9 platforms were completely unable to counter any of the alias mechanisms from the tested email providers.

Finding V: *No platform fully defends against aliasing rules across all 12 providers. As a result, all are susceptible to account creation via alias variants, potentially enabling Alias Multiplicity Abuse.*

Every tested email provider was able to register at least one alias email on at least 13 platforms, with an average of 17.17 successful registrations per provider. Among them, Yandex and ProtonMail proved the most successful, managing to register accounts with alias emails on all tested platforms. 2925Mail, due to its complex alias rules and extensive character support, could bypass almost all platform defenses unless its domain was specifically blocklisted, as seen with *X.com*.

TABLE IV: Alias mechanisms that can have different identities in the platforms.

Platform	Alibaba	2925	Yandex	Zoho	Gmail	Outlook	Proton	Mail.ru	Hotmail	Runbox	iCloud	Eclipso
Microsoft [37]		$\mathcal{S}$	$\mathcal{D}$		$\mathcal{I},\mathcal{D}$		$\mathcal{I}$			$\mathcal{D}$		
Facebook [38]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$			$\mathcal{I},\mathcal{S}$			$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
X [34]			$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Instagram [39]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I}$		$\mathcal{I},\mathcal{S}$			$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Github [32]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Cloudflare [33]		$\mathcal{I},\mathcal{S}$	$\mathcal{D}$				$\mathcal{I}$			$\mathcal{D}$		$\mathcal{P}$
Netflix [40]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Pinterest [41]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Adobe [42]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$		$\mathcal{S}$	$\mathcal{P}$
Vimeo [43]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Spotify [44]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Zoom [45]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Tiktok [46]	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	
Gandi [47]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
Unity [48]	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$
npm [8]	$\mathcal{S},\mathcal{C}$	$\mathcal{I},\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C},\mathcal{D}$	$\mathcal{S},\mathcal{C}$	$\mathcal{I},\mathcal{S},\mathcal{C},\mathcal{D}$	$\mathcal{S},\mathcal{C}$	$\mathcal{I},\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C},\mathcal{D}$	$\mathcal{S},\mathcal{C}$	$\mathcal{P},\mathcal{C}$
Pypi [9]	$\mathcal{S},\mathcal{C}$	$\mathcal{I},\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C},\mathcal{D}$	$\mathcal{S},\mathcal{C}$	$\mathcal{I},\mathcal{S},\mathcal{C},\mathcal{D}$		$\mathcal{I},\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C}$	$\mathcal{S},\mathcal{C},\mathcal{D}$	$\mathcal{S},\mathcal{C}$	$\mathcal{P},\mathcal{C}$
ChatGPT [49]	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{I},\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S},\mathcal{D}$	$\mathcal{S}$	$\mathcal{P}$

$\mathcal{P}$ indicates the platform accept the provider's $\mathcal{P}$*refix-addition* alias as different identity.
$\mathcal{I}$ indicates the platform accept the provider's $\mathcal{I}$*nfix-insertion* alias as different identity.
$\mathcal{S}$ indicates the platform accept the provider's $\mathcal{S}$*uffix-addition* alias as different identity.
$\mathcal{C}$ indicates the platform accept the provider's $\mathcal{C}$*ase-variation* alias as different identity.
$\mathcal{D}$ indicates the platform accept the provider's $\mathcal{D}$*omain-substitution* alias as different identity.

Finding VI: *The absence of alias detection allows aliases from each email provider to bypass registration checks on at least 13 platforms. Notably, ProtonMail and Yandex aliases were accepted by all tested platforms.*

V. Alias Multiplicity Abuse in the Wild

After determining the alias mechanism of the email provider and the identity recognition mechanism of the platform, we explore the usage of alias email in the real world, especially whether there is a possibility of abuse. We first propose three alias abuse threats in practice, then analyze how popular the usage of alias addresses is in the real world by collecting mail addresses from public mail lists and user data from platforms.

A. Threat Scenarios

Due to the limited detection of email aliases by platforms, users can register an unlimited number of accounts using a single primary email address. This offers significant convenience to legitimate users, such as register separate accounts for work, personal projects, or hobbies, thus avoiding the mixing of personal and professional information. However, this convenience is not without its risks. Abusers can exploit these alias mechanisms to engage in malicious activities. Here we categorize some scenarios of Alias Multiplicity Abuse.

Free Trial Abuse. Many platforms offer free trials to attract new users, but attackers can exploit alias emails to repeatedly register and gain prolonged access to premium features at no cost. For example, it is possible for users to exploit unlimited alias registrations to repeatedly claim Microsoft's 1-month Office 365 trials, Adobe's 7-day Creative Cloud access, and Spotify Premium's ad-free 320kbps streaming (vs 160kbps Free tier), bypassing revenue safeguards. Abusers can also drain bandwidth quotas (Cloudflare) and hosting resources (Gandi) from multiple free plan by distribute traffic across alias-bound accounts to bypass per-account limits, systematically undermining monetization while raising costs. We successfully registered more than one account by aliases on these platforms, and all accounts received free trial invitations.

Fake Accounts for Social Manipulation. On platforms like Facebook, Instagram, and TikTok, users can create multiple accounts using alias emails to manipulate engagement metrics, such as likes, comments, or shares. These fake accounts disrupt user interaction data and skew content popularity, undermining the platform's content recommendations and ecosystem integrity. Moreover, alias accounts allow abusers to spread prohibited content or misinformation without facing significant barriers, harming the platform's reputation and user trust. The ease of creating endless variations of accounts enables malicious exploitation, bypassing typical registration checks. We also tested whether alias accounts could be linked to primary email accounts by searching for the primary email, but none of the 18 platforms supported account searches by email. This makes it difficult for users to identify linked accounts, especially those registered with alias emails.

Bypassing Resource Limits. Attackers can exploit alias emails to bypass API rate limits or quota systems by creating multiple accounts, each appearing as a separate identity, allowing them to scrape data, automate interactions, or perform other malicious actions that would otherwise be restricted. For instance, GitHub's REST API [50] limits unauthenticated users to 60 requests per hour and authenticated users to 15,000 per token. Although multiple tokens under one account share this quota, alias-based accounts each receive a full quota, enabling large-scale scraping or abuse. Similarly, alias-based accounts can be used to evade daily usage limits imposed by premium

TABLE V: Overview of alias email address detected by *OriginMail*.

Platform	# of email addresses	# of alias addresses	% of alias address	# BAM [1]
npm	539,105	126,082	23.39%	1,007
GitHub	1,602,342	184,054	11.49%	55
Total	2,141,447	310,136	14.48%	1,062

[1] Number of base addresses that have multiple accounts.

AI models like GPT-4o on ChatGPT, which restricts access per account to prevent overuse.

Finding VII: *Email aliases enable unlimited account creation on platforms, facilitating abuse of free trials, fake account operations, and API rate-limit bypasses.*

B. Measurements

To evaluate real-world alias usage, we collected user email addresses from open-source platforms where such data is publicly accessible. Among the 40 surveyed platforms, only GitHub and npm disclose user emails to support software traceability and security. We gathered 534,400 unique users from 3.3M npm packages and 1,593,131 email addresses from 1.28M GitHub accounts (one account may bind multiple addresses) between 2009 and 2025. Details of the collection process are provided in Appendix A-B.

Based on the alias mechanisms we learn from Section III-B, we developed a tool, *OriginMail* [11], which identifies whether an email address from 28 providers is an alias and extracts the primary email. We applied *OriginMail* to the extracted addresses of npm and GitHub to determine which accounts were registered using alias emails.

Landscape. Table V summarizes the use of alias emails across npm and GitHub. Among the 2,141,447 collected email addresses, we identified 310,136 aliases. Gmail was the most used (94.61%), likely due to its aliasing rules being widely known and clearly documented. The vast majority (97.92%) mapped one-to-one with base emails, likely reflecting legitimate privacy practices. However, single base addresses that used to register multiple accounts may indicate potential misuse. We found that on GitHub, 111 accounts were registered using 55 unique base addresses, while on npm, 2,737 accounts were associated with 1,007 base addresses. Such behaviors date back over a decade, with the earliest plus-suffix alias on GitHub observed in 2009.

npm Abuser Campaign. We found that a significant portion of alias-based accounts on npm were involved in RepSEO campaigns—a form of SEO abuse where attackers publish large volumes of spam packages with promotional README content and no functional code [10]. Cross-referencing with the RepSEO package list [51], among the base addresses used to register multiple accounts, we found that 533 base addresses (52.93%) were involved in publishing SEO packages, collectively releasing 42,699 packages.

The largest campaign was associated with the base email *umekiyanai@gmail.com*, which had 139 alias accounts. These accounts were activated within a 10-day span and published a total of 3,904 packages between April 1–10, 2023. The aliases used a combination of plus-suffixing and case variation (*e.g., UmekiYanai+patrickcabler61@gmail.com* and *umekiyanai+justinwafford25@gmail.com*), and each account published an average of 36.87 packages. All packages shared a name prefix like “pdf_read_down_load”, such as “pdf_read_down_load_imagined_communities_reflections_on”.

VI. Alias Misidentification Attack

We conducted a user study to assess the general familiarity of users with alias email systems. Our findings highlight significant gaps in how various email providers handle aliasing, complicates users’ ability to discern legitimate communications, increasing their vulnerability to *AMisA* attacks.

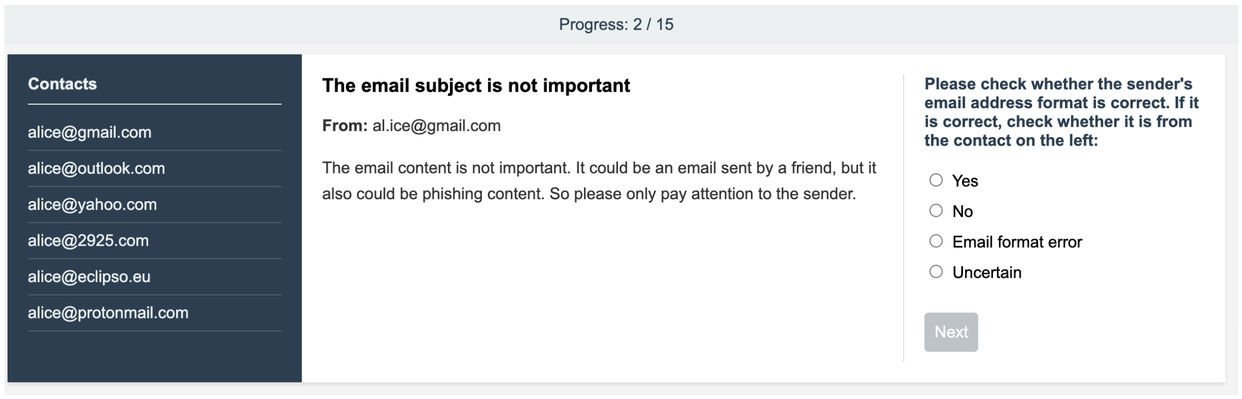
A. User Study Methodology

To evaluate users’ understanding of email aliases, we conducted an experiment where participants acted as users receiving help requests from a friend. The sender’s email address could either be an alias of the friend’s legitimate email or a phishing address resembling the friend’s email, simulating a potential phishing attempt. Participants were asked to determine whether the sender was a known contact. By varying the email address formats, we evaluated how inconsistencies in the alias mechanism affect users’ ability to detect phishing emails.

For this study, we developed a controlled email platform that includes a contact list, emails, and a decision interface, as shown in Figure 5. The platform was fully managed by us, with comprehensive security measures in place to mitigate risks. Participants were tasked with evaluating 15 emails based on the contact list, classifying each sender as a known friend, not a known friend, an invalid address, or uncertain.

Question generation. We investigated how the email alias mechanism influences users’ ability to identify phishing emails, with a key step being the generation of emails from various aliasing schemes to create sender variations. To achieve this, we selected six email providers representing different aliasing mechanisms, including familiar ones and no alias ones: Gmail, Outlook, ProtonMail, 2925Mail, and Yahoo.

We first asked participants how frequently they use email in a week to know their familiarity with email systems. Then, we provided them with six known contacts in the platform’s address book. As shown in Table VI, our 15-email evaluation task divided into four progressive question types to assess participants’ understanding of email aliasing mechanisms: attention validation, basic alias awareness, alias generalization, and confusing aliasing. To capture participants’ genuine reactions and knowledge about aliasing mechanisms, we did not inform them beforehand that the study focused on email aliases. The detailed design purpose of question generation is in Appendix A-C.



Progress: 2 / 15

Contacts

alice@gmail.com
alice@outlook.com
alice@yahoo.com
alice@2925.com
alice@eclipso.eu
alice@protonmail.com

The email subject is not important

From: al.ice@gmail.com

The email content is not important. It could be an email sent by a friend, but it also could be phishing content. So please only pay attention to the sender.

Please check whether the sender's email address format is correct. If it is correct, check whether it is from the contact on the left:

- Yes
- No
- Email format error
- Uncertain

Fig. 5: The web interface of the user study.

TABLE VI: Experimental results of the user study. While receiving 304 results, starting from Q2, the number of users is based on users who answered Q1 correctly, *i.e.,* 174 participants. When calculating correctness, we treated responses marked as "Email format error" as equivalent to a "No" answer.

Question Type	No.	Sender	Valid alias	Correct	Incorrect	Uncertain
Attention Validation	1	alice@gmail.com	Same as contact	174 (57.24%)	92 (30.26%)	38 (12.50%)
Basic Alias Awareness	2	al.ice@gmail.com	Yes	96 (55.17%)	73 (41.95%)	5 (2.87%)
	3	alice+friend@gmail.com	Yes	11 (6.32%)	157 (90.23%)	6 (3.45%)
Alias Generalization	4	alice+friend@outlook.com	Yes	17 (9.77%)	143 (82.18%)	14 (8.05%)
	5	alice+friend@2925.com	Yes	20 (11.49%)	141 (81.03%)	13 (7.47%)
	6	alice+friend@yahoo.com	No	146 (83.91%)	12 (6.70%)	16 (9.20%)
	7	al.ice@protonmail.com	Yes	16 (9.20%)	145 (83.33%)	13 (7.47%)
	8	al.ice@outlook.com	No	155 (89.08%)	11 (6.32%)	8 (4.60%)
Confusing Aliasing	9	friend+alice@eclipso.eu	Yes	10 (5.75%)	154 (88.51%)	10 (5.75%)
	10	friend+alice@yahoo.com	No	149 (85.63%)	14 (8.05%)	11 (6.32%)
	11	alice-friend@2925.com	Yes	12 (6.90%)	154 (88.51%)	8 (4.60%)
	12	alice-friend@eclipso.eu	No	159 (91.38%)	8 (4.60%)	7 (4.02%)
	13	al-ice@protonmail.com	Yes	8 (4.60%)	154 (88.51%)	12 (6.70%)
	14	al-ice@gmail.com	No	151 (65.52%)	14 (8.05%)	9 (5.17%)
	15	ALICE@yahoo.com	Yes	26 (14.94%)	139 (79.89%)	9 (5.17%)

Recruiting Participants. To investigate how users with varying levels of familiarity with email and aliasing mechanisms perceive phishing emails under complex aliasing conditions, we recruited a diverse participant pool. We employed two recruitment strategies. First, we used Prolific [52], a platform regularly used for academic surveys, known for providing a participant pool that is slightly more diverse than typical internet samples. Second, we recruited well-educated graduate students in computer science, who may frequently encounter phishing emails and tend to have stronger security awareness, based on the expectation that such students typically receive some information security education during their campus life, potentially making them more familiar with aliasing mechanisms. To ensure data quality and avoid non-serious participants, we applied common screening criteria used in Prolific studies [53]. We restricted recruitment to users above 18 years old with a minimum of 50 prior surveys on the platform, and had a minimum approval rate of 95%, also fluency in English.

Each participant received $0.5 for completing the study. Participants were informed that we would collect demographic data, including their gender, age, and education level. In total, we recruited 304 participants, the demographic data can be found in Appendix A-C.

B. Results and Analysis

Table VI shows the statistics for the user study results. Overall, participants struggled to accurately recognize email aliases. Among the 304 responses we collected, 174 passed our attention checks and were considered valid. The high failure rate may stem from social desirability bias, where participants chose seemingly "safe" answers rather than carefully evaluating the emails. Since they were not told the study focused on aliases, some may have assumed that an email appearing normal at first glance could not be trusted. Our 174 valid sample exhibited diverse demographic characteristics: 55.75% are male and 44.25% are female. 90.80% participants are 18-50 years old. Most of the participants have a bachelor degree (50.00%) or a master degree (35.63%), followed by those with a high school degree (9.19%) and PhD degree (5.17%).

The accuracy among valid responses was 40.07%, indicating a generally low ability to correctly identify alias emails. Surprisingly, we found that whether users frequently check

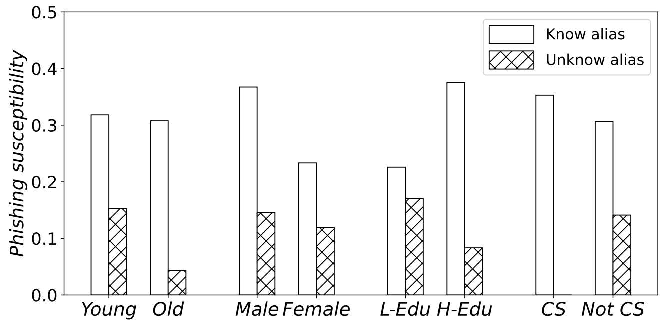


Fig. 6: The joint impact of demographic factors and awareness of alias on phishing susceptibility.

their email has little impact on their ability to correctly recognize alias email addresses. Notably, 10.92% of users failed to identify any of the 14 variant addresses as a known contact ("No") or lacked confidence in judging ("Uncertain"), revealing alias-induced recognition challenges.

Basic Alias Awareness. Since Gmail's aliasing mechanism is the most widely known, we use it as a benchmark to assess participants' basic alias awareness. A total of 101 participants (58.05%) demonstrated some level of awareness: 6 correctly identified all Gmail alias formats, while 95 were partially correct. Notably, awareness of dot-based aliases was higher, with 55.17% recognizing them correctly.

Alias Generalization. Our findings suggest that participants often overgeneralized Gmail's aliasing rules, mistakenly believing that the same formats were supported by all email providers. While we set two non-alias addresses in this stage, 8.91% participants who correctly identified Gmail's aliases erroneously believed other providers universally supported this pattern. That is, while Yahoo does not support any alias, they assumed *alice+friend@yahoo.com* is a valid alias, just like *alice+friend@gmail.com*. Interestingly, 52.48% participants did not choose the same answers for the variants in the same pattern across providers, indicating that they are also questioning whether every email supports this alias mechanism.

Confusing Aliasing. Capitalization changes appeared to be more acceptable to users. 14.94% participants recognized that uppercase variants still represented the same underlying email address. Among them, 19.23% accepted only capitalization-based aliases as valid, while rejecting other forms such as dot or plus variations. Other less common aliasing methods, such as prefixes, non-plus suffixes, or hyphen injection, had very low recognition accuracy (only 5.75%), indicating a general lack of awareness and understanding.

Impact of Alias Awareness. We cross-examine the results with respect to the demographic factors. We divided participants by age (young <40 vs old >= 40), gender, education (below or at least bachelor degree), and technical background (CS students vs non-technical users). Misclassifying non-alias addresses as valid aliases indicates a potential vulnerability to phishing attacks. In this context, we define phishing susceptibility as the proportion of participants who misidentified at least one non-alias address as a valid alias.

Figure 6 shows that alias awareness significantly increased phishing susceptibility across all demographic groups, with the overall phishing susceptibility rate rising from 12.63% to 31.65%. Interestingly, although 45.40% of participants self-reported that they knew about email aliasing, 22.78% of them still failed to correctly identify even the basic Gmail alias formats. This gap between perceived and actual knowledge highlights a risk of overconfidence. Participants with higher education levels (Fisher's exact test $p = 0.0012$) and male participants ($p = 0.0194$) exhibited the highest phishing susceptibility rates (37.5% and 36.73%, respectively) when they know something about alias, suggesting that those who believe they understand the mechanism may be more prone to misjudgment. Notably, CS students, who initially showed no misjudgment, increased to 35.29% phishing susceptibility if they know alias. In contrast, the impact of alias awareness on low-education group was minimal, with only a slight increase of 5.56%. These results highlight that the inconsistent aliasing mechanisms across providers can confuse users and lead to greater identity misrecognition, particularly among those who believe they understand the system.

Finding VIII: *In the absence of clear aliasing standards, users with partial alias knowledge are more prone to misjudgment, making them especially vulnerable to AMisA.*

VII. Discussion

In this section, we discuss the security implications, then propose mitigation and suggestion for email providers and platforms, and discuss our limitations.

A. Security Implication

This work aims to raise community awareness of identity confusion risks introduced by email aliasing mechanisms. Our study reveals a lack of transparency in aliasing policies among email providers, as well as wide inconsistencies across providers, which allow a single email account to generate a large number of aliases that are difficult to identify.

Meanwhile, internet platforms, as the primary users of email identities, have not demonstrated sufficient alarm against aliasing. Most platforms treat different aliases of the same email as separate identities. While a few platforms have adopted partial alias detection, these mechanisms still fall short of comprehensively identifying all aliases. The unlimited creation of alias-based accounts introduces significant abuse risks for platforms, such as free trial abuse, fake accounts for social manipulation, and bypassing API rate limits. We further demonstrate that email aliases have already been exploited in large-scale SEO attacks.

The risks of inconsistent aliasing rules go beyond identity confusion. Although we haven't witnessed existed phishing in the public email lists of the Internet Engineering Task Force (IETF) [54] and the Linux kernel development community [55], there may happen in the future. Our user study shows that users who believe they understand aliasing mechanisms are more likely to misidentify non-alias phishing emails as legitimate aliases, significantly increasing their susceptibility to phishing attacks.

To support the community in understanding and mitigating these inconsistencies, we summarize alias mechanisms of major email providers and release OriginMail, an open-source tool designed to extract the origin email behind its aliases.

B. Disclosure

We actively engaged with both platforms and users to disclose our findings. For all tested platforms, we reported the alias formats that could be used to create accounts. We also provided GitHub and npm with lists of users who had registered multiple accounts using email aliases, and received their acknowledgements. GitHub confirmed that creating multiple accounts via email aliases constitutes an abuse of their service and has suspended the associated spammy users.

In addition, we educated all participants in our user study about email aliasing mechanisms at the end of the survey, to raise their awareness and understanding of this identity risk.

C. Suggestion

By evaluating how different parties interpret email aliases, our findings reveal the identity confusion risks introduced by complex and inconsistent aliasing mechanisms. Based on our results, we offer the following suggestions to email providers, platforms, and end users.

For email providers, we recommend increased collaboration toward standardizing aliasing rules, for example, restricting suffix aliases to the + sign as a separator. To avoid confusion between aliases and visually similar addresses, infix alias symbols should avoid common username characters such as dot (.), hyphen (-), and underscore (_). In addition, providers should increase the transparency of their alias mechanisms and ensure consistency between implementation and documentation. Yahoo limits users to three syntactic aliases for one base address, this may help to mitigate Alias Multiplicity Abuse.

For email-based identity consumers like internet platforms, those who discourage multiple registrations via aliases should implement alias check during email registration checks. Our open-source tool, *OriginMail* [11], summarizes aliasing rules from 28 providers and can help platforms normalize user emails to detect duplicates. Furthermore, platforms should ensure consistency between client and server-side email validation to prevent bypassing. We recommend that platforms notify the base address when a variant address is used for registration attempts, in order to prevent potential account takeover due to misclassification of alias emails, especially as platforms continue to refine their understanding of complex aliasing mechanisms. For example, in addition to disallowing alias login, Facebook proactively sends a password reset email to base address upon detecting a registration attempt using a known alias.

When encountering unfamiliar email addresses that resemble known contacts, users can use *OriginMail* to verify whether an alias resolves to a known address. Nevertheless, we strongly advise caution toward all unfamiliar email variants, as valid alias email and *Alias Misidentification Attack* may be difficult to distinguish.

D. Limitations

Our study has several limitations. First, we excluded independent secondary aliases from our scope due to their lack of traceable similarity to the base address. While these aliases pose the same threats, they are typically subject to strict quantity limits by providers, making them less likely to be abused at scale for identity obfuscation. When testing alias rules, although using multiple variants per aliasing pattern may help uncover additional edge-case aliasing rules, we intentionally limited our probing to a single variant per pattern due to ethical considerations.

In the selection of email provider, constrained by access restrictions and cost, we tested alias mechanisms of 28 email providers, and we cannot guarantee that we have found all aliasing rules. Consequently, *OriginMail*'s coverage is limited to these providers. However, the diversity of alias types we uncovered is sufficient to raise awareness of the associated risks. As for the enterprise email services, we investigated 20 providers and checked their documents, no provider claim that they have syntactic aliases, and 10 of them have customized aliases that need manual setup. Our current actual alias test set includes one enterprise provider, Tencent Exmail, however, we were unable to test other enterprise services, as most require a registered business account.

Our alias account registration testing framework is semi-automated and involves manual steps prior to executing automated tests, which limits our testing scope, we could not learn all platforms' identity recognition policies toward alias addresses. However, we tested 18 widely used platforms of different categories to ensure broad representativeness, understanding email identity confusion of the top 100 Tranco domains also has real-world security implications. Meanwhile, due to ethical considerations on avoiding account management issue for platforms, we minimized the creation of real accounts and did not activate every alias-based account that passed all checks. We acknowledge that this may overlook checks that occur after account activation, but when we activated one alias account per platform to demonstrate the potential abuse of alias emails, we did not observe any later rejection of alias-based accounts after activation.

VIII. Related Work

A. Email spoofing attacks

Email has long been fraught with security issues such as email spoofing attacks [2, 56, 57]. To address these problems, various security extensions have been proposed and standardized, most notably SPF [58], DKIM [59], and DMARC [60]. However, a number of studies have demonstrated techniques for bypassing these defenses. Bennett et al. [61] identified a buffer overflow vulnerability in libSPF2, which is one of the SPF libraries. Shen et al. [2] exploited automatic email forwarding service to bypass the security validation.

Chen et al. [56] leveraged inconsistencies in how different components of mail systems perform sender authentication to bypass protocol enforcement. Ma et al. [3] identified an overlooked delegation mechanism within mail infrastructures that enables attackers to forge legitimate-looking messages.

While previous research all focused on weaknesses within email systems themselves, our work investigates how different interpretations of alias-addressing mechanisms between providers and platforms can cause identity confusion and lead to successful *Alias Multiplicity Abuse*.

B. Email Phishing Attacks

Most prior work on email phishing has focused on detection rather than interception and proposes a variety of methods [62, 63, 64, 65, 66]. Ho et al. [67] developed a lateral phishing detection approach for enterprise environments, which compares similarities between recipient sets and historical mailing patterns, checks for phishing-related keywords, and inspects URLs against known malicious domain patterns. In another study, Ho et al. [68] employed sender and domain reputation features to flag phishing attempts. The systems proposed by Stringhini and Thonnard [69], Duman et al. [70] and Khonji et al. [71] build behavioral models for senders based on metadata, stylometry, and timing features. They then classify an email as spearphishing or not by using the behavioral model to see whether a new email's features differ from the sender's historical behavioral profile. In our study, when users fail to correctly understand email aliases, they may misidentify an attacker's phishing email as coming from a legitimate alias address and thus fall victim to the phishing attack.

IX. Conclusion

Email aliasing, while enhancing usability and privacy, introduces systemic identity confusion. This study presents the first comprehensive analysis of email aliasing mechanisms and reveals the confusion stemming from inconsistencies across email providers, online platforms, and end users. Our examination of 28 popular email providers shows a widespread lack of transparency and consistency in alias-related documentation. These discrepancies create opportunities for alias multiplicity abuse on platforms and increase the risk of misidentification attacks, particularly among users who are aware of aliasing but misunderstand its differences. Our findings highlight the urgent need for standardization and improved transparency in alias handling. To aid in mitigating these issues, we have open-sourced *OriginMail*, an alias normalization tool. We responsibly disclosed our findings to relevant platforms and received acknowledgments.

X. Ethics Considerations

We place a strong emphasis on ethical integrity throughout all stages of our research, and have taken several measures in each experiment.

In our provider alias mechanism experiment, all email accounts used were created and owned by us, ensuring no real users were affected by our experiments. To minimize the number of sent emails, we restrict character modifications to fixed positions (*e.g.,* only altering the capitalization of the first letter in the username) rather than exhaustively testing all possible positions (*e.g.,* second-letter capitalization). This ensures efficiency while maintaining systematic variation. For each provider, except for providers that can substitute domains, we need to send 98 emails to complete all variant testing, thus we carefully controlled the email-sending rate to be under 10-minute intervals per email to minimize the impact on target servers. Some alias-supported symbols may unintentionally route emails to unintended recipients in certain edge cases. For example, in Figure 2, although our intent is to test whether the '+' symbol can be used in the infix position (*e.g., ali+ce@a.com*), which is not a valid alias of *alice@a.com*, the message may still reach *ali@a.com* if that address supports '+'-suffix aliases. To minimize the risk of misdirected emails, we used long, randomized usernames (12 characters) to avoid collisions with existing accounts. Additionally, each email explicitly stated that it was part of a scientific research study and included an opt-out notice to adhere to ethical communication practices.

When registering alias accounts on platforms, we took careful measures to avoid creating real accounts that could lead to management issues for the platforms. Our process involved a two-step check to verify the availability of variant addresses. Many variants were blocked during the validity check, and while a small number of registrations were successful, most platforms required email activation to finalize the account creation. To minimize the creation of real accounts, we intentionally refrained from activating them. Also, we controlled our testing frequency, ensuring at least a ten-minute interval between registration attempts for each variant address per platform. For each platform, we activated only two accounts to demonstrate the potential abuse of alias emails, such as receiving free trial offers or mutual interactions like likes. After the experiment concluded, we ensured that these accounts were deactivated. To avoid raising operation costs, we did not really use the trial functions. We responsibly disclosed our findings to the respective platform security teams, and received acknowledgments from them.

In our analysis of real-world alias usage, we relied solely on publicly available data provided by ietf, linux community, npm, and github, without engaging in any database attacks against these platforms. Additionally, during our periodic collection for email address, we strictly abide by the limitations of our account, responsibly using their query or download API.

Our user study procedure is approved by the IRB. The study platform was self-hosted and designed with appropriate security measures to protect participants' data. Although IP addresses were technically accessible during the survey, we chose not to store them. Participants were informed of their right to withdraw from the study at any time, and no withdrawal requests had been received at the time of submission. At the end of the study, we provide participants with information about the alias mechanism to raise awareness.

Acknowledgment

We would like to thank the anonymous reviewers for their valuable comments, which improved the quality of the paper. This work was sponsored by the National Key Research and Development Program of China under grant No. 2024YFF0618800, and the National Natural Science Foundation of China (62302101, 62402114). Min Yang is a faculty of Shanghai Institute of Intelligent Electronics & Systems and Engineering Research Center of Cyber Security Auditing and Monitoring, and Shanghai Collaborative Innovation Center of Intelligent Visual Computing, Ministry of Education, China.

References

[1] Gmail. (2025) Send emails from a different address or alias. *https://support.google.com/mail/answer/22370*.

[2] K. Shen, C. Wang, M. Guo, X. Zheng, C. Lu, B. Liu, Y. Zhao, S. Hao, H. Duan, Q. Pan, and M. Yang, "Weak Links in Authentication Chains: A Large-scale Analysis of Email Sender Spoofing Attacks," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 2021, pp. 3201–3217.

[3] J. Ma, L. Chen, K. Xue, B. Luo, X. Huang, M. Ai, H. Zhang, D. S. L. Wei, and Y. Zhuang, "FakeBehalf: Imperceptible Email Spoofing Attacks against the Delegation Mechanism in Email Systems," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 1243–1260.

[4] M. I. Ashiq, W. Li, T. Fiebig, and T. Chung, "You've Got Report: Measurement and Security Implications of DMARC Reporting," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 4123–4137.

[5] eclipso Mail Europe International. (2025) eclipso mail europe. privacy focused. *https://www.eclipso.eu/*.

[6] Yandex. (2025) Yandex mail. *https://360.yandex.com/mail/*.

[7] C. Support. (2025) A variation of this email address is already taken in our system. only one variation is allowed. *https://developers.cloudflare.com/support/troubleshooting/http-status-codes/cloudflare-1xxx-errors/#error-1104-a-variation-of-this-email-address-is-already-taken-in-our-system-only-one-variation-is-allowed*.

[8] Npmjs. (2025) Build amazing things. *https://www.npmjs.com/*.

[9] PyPI. (2025) Find, install and publish python packages with the python package index. *https://pypi.org/*.

[10] M. Wu, G. Hong, W. Mai, X. Wu, L. Zhang, Y. Pu, H. Chai, L. Ying, H. Duan, and M. Yang, "Exposing the hidden layer: Software repositories in the service of seo manipulation," in *Proceedings of the IEEE/ACM 47th International Conference on Software Engineering*, 2025.

[11] Anonymous. (2025) Originmail. *https://anonymous.4open.science/r/OriginMail-7A47*.

[12] Gmail. (2025) Dots don't matter in gmail addresses. *https://support.google.com/mail/answer/7436150*.

[13] ——. (2025) Getting someone else's mail. *https://support.google.com/mail/answer/10313*.

[14] 2925Mail. (2025) How to get a sub-account? *https://www.2925.com/helpcenter/#/questions?id=1&cid=1*.

[15] Yandex. (2025) Additional addresses. *https://yandex.com/support/mail/web/preferences/about-sender/additional-addresses.html*.

[16] Yahoo. (2025) Create, use, edit, or delete temporary email addresses in new yahoo mail. *https://help.yahoo.com/kb/SLN28815.html*.

[17] Proton. (2025) Types of email addresses and aliases. *https://proton.me/support/addresses-and-aliases#additional*.

[18] Zoho. (2025) Zoho - secure business email hosting for your organization. *https://www.zoho.com/mail/*.

[19] 139Mail. (2025) China mobile 139 mail. *https://mail.10086.cn/*.

[20] Aliyun. (2025) Alibaba mail. *https://mail.aliyun.com/*.

[21] D. J. C. Klensin, "Simple Mail Transfer Protocol," RFC 5321, Oct. 2008. [Online]. Available: *https://www.rfc-editor.org/info/rfc5321*

[22] P. Resnick, "Internet Message Format," RFC 5322, Oct. 2008. [Online]. Available: *https://www.rfc-editor.org/info/rfc5322*

[23] P. P. Fraud. (2025) Free email spam test. *https://www.ipqualityscore.com/email-deliverability/email-spam-test-checker/*.

[24] Mailmeteor. (2025) Spam checker. *https://mailmeteor.com/spam-checker*.

[25] Mail.ru. (2025) Mail.ru. *https://mail.ru/*.

[26] Microsoft. (2025) Microsoft outlook (formerly hotmail): Free email and calendar. *https://www.microsoft.com/en-us/microsoft-365/outlook/email-and-calendar-software-microsoft-outlook*.

[27] iCloud. (2025) icloud mail - apple icloud. *https://www.icloud.com/mail/*.

[28] U. Mail. (2025) 2925 mail - unlimited mail. *https://www.2925.com/*.

[29] Google. (2025) Email - gmail - google. *https://mail.google.com/mail/u/0/*.

[30] Proton. (2025) Proton mail: Get a free email account with privacy and encryption. *https://proton.me/mail*.

[31] Runbox. (2025) Secure and private email hosting services by runbox. *https://runbox.com/*.

[32] GitHub. (2025) Github. *https://github.com/*.

[33] Cloudflare. (2025) Cloudflare: Connect, protect, and build everywhere. *https://www.cloudflare.com/*.

[34] X. Corp. (2025) X. *https://x.com/*.

[35] g1879. (2025) Drissionpage. *https://github.com/g1879/DrissionPage*.

[36] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," in *Proceedings of the 24th Network and Distributed System Security Symposium (NDSS)*, 2019.

[37] Microsoft. (2025) Microsoft. *https://www.microsoft.com/*.

[38] Meta. (2025) Facebook connect with friends and the world around you on facebook. *https://www.facebook.com/*.

[39] ——. (2025) Instagram. *https://www.instagram.com/*.

[40] Netflix. (2025) Unlimited movies, tv shows, and more. *https://www.netflix.com/*.

[41] Pinterest. (2025) Pinterest. *https://www.pinterest.com/*.

[42] Adobe. (2025) The ultimate creative ai solution. *https://www.adobe.com/*.

[43] Vimeo. (2025) Do more with video. *https://vimeo.com/*.

[44] Spofity. (2025) Spofity. *https://open.spotify.com/*.

[45] I. Zoom Communications. (2025) Zoom. *https://www.zoom.com/*.

[46] Tiktok. (2025) Tiktok. *https://www.tiktok.com/explore*.

[47] Gandi. (2025) A domain name for a secure online space. *https://www.gandi.net/*.

[48] Unity. (2025) Go create. *https://unity.com/*.

[49] ChatGPT. (2025) create-account. *https://auth.openai.com/create-account*.

[50] GitHub. (2022) Rest api. *https://docs.github.com/en/rest/using-the-rest-api/rate-limits-for-the-rest-api?apiVersion=2022-11-28*.

[51] Marphownio. (2024) Repseo-package-list. *https://github.com/Marphownio/RepSEO_Classifier/tree/main/RepSEO-package-list/npm*.

[52] Prolific. (2025) Prolific. *https://www.prolific.com/*.

[53] M. Minaei, M. Mondal, and A. Kate, "Empirical understanding of deletion privacy: Experiences, expectations, and measures," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 3415–3432.

[54] I. Datatracker. (2025) Web-based working group email archives. *https://datatracker.ietf.org/list/wg/*.

[55] kernel.org. (1970) All of lore.kernel.org. *https://lore.kernel.org/all/*.

[56] J. Chen, V. Paxson, and J. Jiang, "Composition Kills: A Case Study of Email Sender Authentication," in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, 2020, pp. 2183–2199.

[57] H. Hu and G. Wang, "End-to-End Measurements of Email Spoofing Attacks," in *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, 2018, pp. 1095–1112.

[58] S. Kitterman, "Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1," RFC 7208, Apr. 2014. [Online]. Available: *https://www.rfc-editor.org/info/rfc7208*

[59] M. Kucherawy, D. Crocker, and T. Hansen, "DomainKeys Identified Mail (DKIM) Signatures," RFC 6376, Sep. 2011. [Online]. Available: *https://www.rfc-editor.org/info/rfc6376*

[60] M. Kucherawy and E. Zwicky, "Domain-based Message Authentication, Reporting, and Conformance (DMARC)," RFC 7489, Mar. 2015. [Online]. Available: *https://www.rfc-editor.org/info/rfc7489*

[61] N. Bennett, R. Sowards, and C. Deccio, "Spfail: discovering, measuring, and remediating vulnerabilities in email sender validation," in *Proceedings of the 22nd ACM Internet Measurement Conference*, ser. IMC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 633–646.

[62] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in *Proceedings of the Anti-Phishing Working Groups 2nd Annual ECrime Researchers Summit*, ser. eCrime '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 60–69.

[63] A. Bergholz, J. H. Chang, G. Paass, F. Reichartz, and S. Strobel, "Improved phishing detection using model-based features," in *International Conference on Email and Anti-Spam*, 2008.

[64] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proceedings of the 16th International Conference on World Wide Web*, ser. WWW '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 649–656.

[65] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in *Proceedings of the 2007 ACM Workshop on Recurring Malcode*, ser. WORM '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 1–8.

[66] C. Whittaker, B. Ryner, and M. Nazif, "Large-scale automatic classification of phishing pages." in *Ndss*, vol. 10, 2010, p. 2010.

[67] G. Ho, A. Cidon, L. Gavish, M. Schweighauser, V. Paxson, S. Savage, G. M. Voelker, and D. Wagner, "Detecting and Characterizing Lateral Phishing at Scale," in *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, 2019, pp. 1273–1290.

[68] G. Ho, A. Sharma, M. Javed, V. Paxson, and D. Wagner, "Detecting Credential Spearphishing in Enterprise Settings," in *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, 2017, pp. 469–485.

[69] G. Stringhini and O. Thonnard, "That ain't you: Blocking spearphishing through behavioral modelling," in *Proceedings of the 12th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment - Volume 9148*, ser. DIMVA 2015. Berlin, Heidelberg: Springer-Verlag, 2015, p. 78–97.

[70] S. Duman, K. Kalkan-Cakmakci, M. Egele, W. Robertson, and E. Kirda, "Emailprofiler: Spearphishing filtering with header and stylometric features of emails," in *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1, 2016, pp. 408–416.

[71] M. Khonji, Y. Iraqi, and A. Jones, "Mitigation of spear phishing attacks: A content-based authorship identification framework," in *2011 International Conference for Internet Technology and Secured Transactions*, 2011, pp. 416–421.

[72] ericmutta. (2023) registry. *https://docs.npmjs.com/cli/v11/using-npm/registry*.

[73] Github. (2022) List public repositories. *11-28#list-public-repositories*. *https://docs.github.com/zh/rest/repos/repos?apiVersion=2022-*

Appendix A
Methodology Appendix

A. Identity in email providers

1) Username Requirements: As noted in our alias policy analysis (Section III-A), providers that support character-based aliasing typically rely on special characters to distinguish aliases from the base address. So, we first analyzed the username character constraints across providers during account registration to determine whether certain characters might be reserved for alias generation.

Of the 28 providers analyzed, 2980Mail and 2925Mail were the only ones that restricted usernames to letters and digits only, without support for special characters. All other providers allow one or more of the following three special characters: underscore (_), dot (.), and hyphen (-). Underscore (_) was the most widely supported, allowed by 21 providers, while dot (.) was accepted by 18 providers and Hyphen (-) was supported by 13 providers. Notably, four Chinese providers in our dataset supported only the underscore, whereas most providers from other countries typically supported all three characters.

2) Ignored and Invalid Characters: 20 in 28 providers ignore the backslash(\) during address interpretation – effectively treating *user\name@domain.com* identically to *username@domain.com*, only three providers treat the address with backslash as an invalid address. After excluding these two situations, there are still 12 providers that support other email aliases. Besides, 10 characters consistently led to invalid email addresses across providers. These include: `@`, parentheses `()`, square brackets `[]`, semicolon `;`, double quotes `"`, comma`,`, and angle brackets <>.

B. Alias Multiplicity Abuse Dataset

To evaluate real-world alias usage, we try to collect email addresses from the accounts of platforms. However, email addresses are often considered private information for most internet platforms. After surveying 40 well-known different-type platforms, we found that only a few platforms allow users to decide whether their email is public. Only npm and GitHub make users' email addresses public for the purpose of open-source software security and traceability. For npm, we crawled the package indexs and accessed each package's metadata by npm API [72]. We extracted register email addresses from `_npmUser` , `author`, and `maintainer` field. For GitHub, we used the query API [73] with different languages to get repositories. We then use repository API [50] to get commit histories. For each commit, it includes the committer's homepage URL, if the homepage is accessible, we can verify the email address of the commit belongs to a GitHub user.

In total, we collected 3,310,406 npm packages and 2,219,000 GitHub repositories between 01/01/2009 and 02/28/2025, getting 539,105 unique npm users and 1,602,342 addresses from 1,282,532 Github accounts, as one account can bind several addresses.

C. User Study

Participant statistics. We recruited 304 participants for our user study, and the sample exhibited diverse demographic characteristics: 60.81% are male and 39.19% are female. 87.50% participants are 18-50 years old. Most of the participants have a bachelor degree (48.03%) or a master degree (28.95%), followed by those with a phd degree (12.17%) and highschool degree (10.86%).

Sender generation. We investigated how the email alias mechanism influences users' ability to identify phishing emails, with a key step being the generation of emails from various aliasing schemes to create sender variations.

To achieve this, we selected six email providers representing different aliasing mechanisms: 1) Gmail, the most well-known alias mechanism, where users may be familiar with its rules; 2) Outlook and ProtonMail, which have similar but slightly different aliasing rules compared to Gmail, potentially leading to partial recognition but uncertain judgments; 3) 2925Mail and Eclipso, which use unique aliasing mechanisms, likely cause users to misidentify them as entirely unrelated phishing emails; 4) Yahoo, which does not support aliasing, serves as a baseline for comparison.

We first asked participants how frequently they use email in a week to know their familiarity with email systems. Then, we provided them with six known contacts in the platform's address book. Each participant was asked to assess whether the emails originated from these known contacts. We designed a 15-email evaluation task divided into four progressive stages to assess participants' understanding of email aliasing mechanisms, as shown in Table VI. Stage 1 (Email 1) served as an attention check using an exact address match from the participant's predefined contacts, we also checked if the participant could understand the quiz. Stage 2 (Emails 2-3) tested basic alias awareness through two Gmail aliases, as the Gmail alias is the most well-known one, including plus-suffix alias and dot-infix alias. Stage 3 (Emails 4-8) examined alias generalization by applying Gmail-style syntax to five other providers (3 legitimate, 2 phishing). Stage 4 (Emails 9-15) evaluated comprehension of unconventional aliasing formats across providers (4 legitimate, 3 phishing), including one case-sensitivity test. This phased approach allowed us to systematically measure how users' awareness of email aliasing scales from familiar to unfamiliar scenarios while maintaining a balanced legitimate/phishing ratio in later stages.

To capture participants' genuine reactions and knowledge about aliasing mechanisms, we did not inform them beforehand that the study focused on email aliases. Only at the end of the experiment did we ask whether they were aware of email aliases and then reveal the true purpose of the study. Importantly, no actual phishing emails were sent—participants were only tasked with assessing whether a sender was trustworthy.

Instruction to participants. We educated all participants in our user study about email aliasing mechanisms at the end of the survey:

> Thank you for participating in this survey! Introduction to Email Aliases: An alias is an additional email address associated with your

primary email address. All emails sent to these aliases will be automatically forwarded to your primary email address. This feature not only helps you protect your real email address and reduce the risk of spam and phishing, but also allows you to effectively manage and classify emails in your inbox by creating dedicated aliases for different scenarios. Some emails support some very convenient aliases, such as Gmail's + suffix alias. If your main email is user@gmail.com, you can use aliases such as user+shopping@gmail.com, user+newsletters@gmail.com or user+work@gmail.com to register for different services. All emails sent to these aliases will be forwarded to user@gmail.com. Additionally, some email services allow you to create completely separate alias addresses that also forward email to your primary email address, but do not reveal your real email address when displayed externally.

APPENDIX B
ARTIFACT APPENDIX

This appendix provides a complete roadmap for reproducing the research artifacts associated with our paper. We contribute the *OriginMail* tool to help platforms resolve alias confusion and disclose vulnerabilities to affected stakeholders. It can extract the base email addresses of a syntactic alias, *e.g.,* you can get *alice@gmail.com* while your input is *al.ice@gmail.com* or *alice+test@gmail.com*. It supports alias rules of 28 email providers. We also use it to find real-world abuse case in Github and npm.

A. Description & Requirements

1) How to access: The code for the artifact, our core tool *OriginMail* can be accessed at: https://github.com/labrynth/OriginMail or https://doi.org/10.5281/zenodo.16735091.

2) Hardware dependencies: None.

3) Software dependencies: The code was tested on MacOS and using Python 3.9. It can run on any platform with Python installed.

B. Artifact Installation & Configuration

- *[Python Version]*: Python 3.9+ (recommended)
- *[Download and unzip the repository]*:
```
unzip originmail.zip
cd originmail/
```
- *[Core Artifact (OriginMail)]*: Requires only Python built-in libraries (no additional installation needed).

C. Major Claims

The following are the major claims we make for the artifact available and artifact functional badge.

- (C1): *OriginMail* can accurately identifies alias emails from 28 providers and extracts their base addresses, based on the alias mechanisms learned in Section III-B. This is proven by the experiment (E1).

D. Evaluation

Please run the following experiments to verify the claims.

1) Experiment (E1): Individual Alias Detection [1 human-minute + 1 compute-minutes]: Validate *OriginMail*'s core functionality by detecting alias emails from single inputs.

- *[Preparation]* Ensure `originmail.py` and `test.py` exist in folder `originmail/src`. No additional data required (input is provided via command line.
- *[Execution]* Run the following command to test alias detection for single email address:
```
python src/originmail.py
alice+1@gmail.com
```
We also provide a script to demo all 28 email provider's alias address:
```
python src/test.py
```
- *[Results]* Expected output format:
```
Origin email for alice+1@gmail.com is
alice@gmail.com
```
Verify the output matches the alias rules in Section III-B of the paper.