An Investigation of Identity-Account Inconsistency in Single Sign-On

Guannan Liu
University of Delaware
Newark, Delaware, USA
gliu@udel.edu

Xing Gao
University of Delaware
Newark, Delaware, USA
xgao@udel.edu

Haining Wang
Virginia Tech
Arlington, Virginia, USA
hnw@vt.edu

ABSTRACT

Single Sign-On (SSO) has been widely adopted for online authentication due to its favorable usability and security. However, it also introduces a single point of failure since all service providers fully trust the identity of a user created by the SSO identity provider. In this paper, we investigate the identity-account inconsistency threat, a new SSO vulnerability that can cause the compromise of online accounts. The vulnerability exists because current SSO systems highly rely on a user's email address to bind an account with a real identity, but ignore the fact that email addresses might be reused by other users. We reveal that under the SSO authentication, such inconsistency allows an adversary controlling a reused email address to take over associated online accounts without knowing any credentials like passwords. Specifically, we first conduct a measurement study on the account management policies for multiple cloud email providers, showing the feasibility of acquiring previously used email accounts. We further perform a systematic study on 100 popular websites using the Google business email service with our own domain address and demonstrate that most online accounts can be compromised by exploiting this inconsistency vulnerability. To shed light on email reuse in the wild, we analyze the commonly used naming conventions that lead to a wide existence of potential email address collisions, and conduct a case study on the account policies of U.S. universities. Finally, we propose several useful practices for end-users, service providers, and identity providers to protect against this identity-account inconsistency threat.

CCS CONCEPTS

• **Security and privacy** → **Authentication**; **Authorization**; **Access control**; Web application security.

KEYWORDS

Single Sign-On, Authentication, Email Account

ACM Reference Format:
Guannan Liu, Xing Gao, and Haining Wang. 2021. An Investigation of Identity-Account Inconsistency in Single Sign-On. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3450085

1 INTRODUCTION

Over the past decade, Single Sign-On (SSO) has been widely used [1] for its favorable usability and security assurance to reduce password fatigue [2] and security risk of accessing third-party websites. Many email providers and social media services, including Google, Microsoft, and Facebook, have provided free Web SSO solutions. The original design of SSO is to allow users to perform cross-domain authentication using a federated identity. With much work concentrating on the security enhancement of the federated identity and its management systems [3, 4], the majority of today's online Service Providers (SPs) entirely outsource the account authentication process to reliable and trusted Identity Providers (IdPs). Typically, once the authentication on the IdP side is successful, an end-user is permitted to access the associated online account without additional security checks.

Since SSO serves as a single bridge between the user identity and online accounts, it also inevitably introduces a single point of failure in the overall authentication process. The vulnerabilities existed in SSO might allow attackers to compromise user accounts on service providers, and hence it has always been an attractive target to attackers [5–7]. A large amount of research efforts have been devoted to automatically disclose vulnerabilities and combat the logic flaws in the overall SSO authentication systems [8, 9].

In this paper, we present a new SSO vulnerability, the identity-account inconsistency threat, which can result in account compromises for service providers. Existing SSO systems heavily rely on a user's email address to bind an account with a real identity, and many service providers adopt email addresses as the primary account identification. In particular, IdPs usually provide a unique ID for SPs to verify the user identity by checking the corresponding information in the user account. If the ID is mismatched, SPs tacitly agree that a matched email address can verify the user's identity, and thus grant the access. However, in many scenarios, email addresses might actually be reused by other users, resulting in an inconsistency between the user's account and identity. Given the existing SSO authentication systems, such inconsistency can further allow an adversary controlling a reused email address to take over associated accounts on service providers.

Unlike traditional password recovery attacks [10, 11], SSO authentication does not require adversaries to crack or recover victims' passwords. Once an adversary obtains a recycled email address, IdPs recognize the adversary as the only legitimate user and owner of the email address. The problem is that such a scenario (i.e., an email address is reused by another user) is unknown by SPs. Therefore, the adversary could be able to legally use the SSO service provided by the IdP to hijack accounts on SPs (belonging to a victim) without knowing any credentials. As a result, all existing defense

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450085

mechanisms against password recovery attacks, such as two-factor authentication [12] and additional knowledge [13], have become futile.

To comprehensively understand the identity-account inconsistency threat in the wild, we conduct a systematic study on 100 popular websites using the Google business email service with our own domain address. We consider three different scenarios where inconsistency might happen and observe that 80 out of 100 websites are vulnerable to the inconsistency threat, implying that a reused email address can easily compromise those accounts without knowing passwords.

To demonstrate the possibility and feasibility of email reuse, we investigate the account management policies of multiple cloud email providers, including both public accounts and business accounts. Our study covers three top email providers in the U.S. accounting for a total of 85% of the market share [14] and one of the top email providers in China hosting more than 800 million active users [15]. We find that most email providers allow an email address re-registration once the email address has been abandoned. We also analyze several widely adopted email naming conventions to further understand the likelihood of email address reuse. Based on the employee history of the Metropolitan Transportation Authority (MTA) of the state of New York, we show that the email naming collision indeed has a relatively high probability to occur. We further conduct a case study on 50 randomly chosen U.S. universities and observe that 32 of them allow students to modify their email addresses, and 24 of them delete student email accounts after their graduation, which can potentially cause the reuse of email addresses.

Finally, we propose several useful practices to mitigate the identity-account inconsistency threat. Both SPs and IdPs should adopt necessary mechanisms to prevent online accounts from being compromised due to the disagreement between user identity and account information. Meanwhile, we suggest that end-users should be aware of the existence of email address reuse and use the SSO authentication with more caution than before.

The rest of this paper is structured as follows. Section 2 introduces the background of SSO, accounts, and identities. Section 3 gives an overview of the identity-account inconsistency threat and the threat model. Section 4 shows that email address reuse could happen in most email providers by studying email account management policies. Section 5 details the account matching procedure and its problems of causing an account compromise. Section 6 analyzes the possibility of email address reuse, and Section 7 discusses several mitigation practices from different aspects. Section 8 surveys related work, and finally, Section 9 concludes.

2 BACKGROUND

2.1 Single Sign-On System Overview

The primary purpose of deploying SSO authentication is to enable the use of federated user identities to log in to various online services. Google, for example, allows users to access its associated services with a single Gmail account. Universities also use banner systems enabling students and employees to access management systems and online resources with a single identity. Since the centralized user identification and authentication system can further improve account security [16], many well-reputed identity management services encourage users to redirect authentication requests to their servers [17]. While SSO is widely used by many applications, in this work, we primarily focus on the website/application login framework (i.e., Web SSO).

The SSO authentication commonly uses the Authorization Code Flow, which involves the token access and URL redirection across three major parties: *end-user*, *service provider*, and *identity provider*. End-users are individuals who attempt to login online services or accounts. SPs are the websites providing services for end-users. IdPs are the identity management systems responsible for providing authentication services to SPs. Typically, an end-user first submits a login request to an SP. The SP then redirects the end-user to visit the IdP authentication URL. After inputting the IdP account credentials, the user authorizes the IdP server to deliver the user identity and corresponding attributions to the SP through multiple SSO tokens. Then, the SP starts to identify an account associated with the received user identity and attributions. Once recognized a matched account, the end-user is granted permission to log in to this account. Otherwise, the SP starts to guide the end-user to establish a new account.

2.2 Account versus Identity

SSO enables end-users to authenticate to an online account with their identities. Particularly, an identity is a special type of account managed and maintained by the IdP server. In SSO, user identities serve as an authentication factor for user accounts in SPs.

Account refers to the digital entity being held in SPs to differentiate services for each end-user. End-users are required to establish at least one unique identifier for their accounts. Traditionally, this identifier is user-defined and referred to as a username or account ID. Since the proposal of Email-Based Identification and Authorization (EBIA) [18], SPs start to adopt email addresses to replace traditional usernames. Because the uniqueness of email addresses is strictly enforced, it ensures that each account receives an exclusive identifier.

Identity refers to a particular type of accounts managed by IdPs. In SSO, it is used for SPs as another account identifier. The user identity is issued by IdPs as the form of SSO tokens, and the information encapsulated in the identity should be kept as secret. The identity first contains several user attributions as supplementary information about the end-user. Popular user attributions include an email address (in the *"email"* field), a user's full name, and a preferred username. The IdP is responsible for controlling this type of information shared with SPs. Besides, SSO specifications require each identity to be assigned with a unique number or a set of characters (known as UserID), which is stored in the *"sub"* field. The unique UserID is created when an end-user registers the identity in IdPs, and can only be assigned to just one user identity (i.e., never be recycled).

2.3 Identity Management System

In SSO, the Identity Management System (IMS) offers a universal authentication scheme to online accounts using a single user identity. IMS integrates IdP services to administer identities for multiple end-users.

Email Providers. Since the proposal of EBIA, SPs start to endorse an email address as an account's username. Email providers then expand their services not only as message transfer agents but also as IMS for user identities. Many email providers offer public accounts free of charge, and an end-user with a registered account receives both email and identity services. Public email accounts are managed by the email providers, and end-users have limited management and customization capabilities.

Enterprise administrators can also register a paid business account on many email providers. They can add as many email addresses as they need within one domain name. Each email address in a business account represents one user in the organization, and every user also receives a digital identity along with the email address. Email providers grant the administrator with more flexible management capabilities. An administrator can set up email naming conventions for the organization's users (e.g., employees) in order to unify the format of email addresses across the entire organization. Some email providers also allow users to create aliases on top of their primary email addresses. This provides users an opportunity to obtain another email address without changing the primary address. Both aliased and primary email addresses share the same email inbox, and outgoing emails can carry either an aliased address or primary address as a sender.

Identity and Access Management (IAM) System. Many large-scale enterprises further invest in IAMs for managing employees and their digital identities. IAMs allow the IT department to establish more complex and flexible management policies. It can also achieve an automatic management procedure with well-defined management flows. In addition, IAMs include IdP services to provide employees with SSO capabilities for accessing many internal resources.

A typical example of an IAM is the banner system adopted by many universities. Students, faculties, and staffs are assigned with an identity (often called *NetID*) when they become affiliated members. With their digital identities, they have access to internal resources. Besides NetID, students and employees also receive educational email addresses (often end with `.edu`). If a university outsources its email service to an external email provider that also offers the SSO service as an IdP, everyone receives an additional digital user identity assigned by the email provider.

3 VULNERABILITY OVERVIEW

SSO requires both SPs and IdPs to follow particular implementation specifications to provide secure authentication services to end-users. The IdP is responsible for ensuring that the owner of an identity receives exclusive access, and provides the unique UserID and other user attributions to SPs. SPs should identify the associated account only with the provided identity. As long as SPs ensure that each identity is linked to a specific account, the account identification process should be fairly straightforward.

However, even the SSO requirements specify detailed authentication flows and security specifications for both SPs and IdPs, the actual implementations and system configurations may have diverged from the original designs. In particular, inconsistency might exist when part of a user's information is updated. In this section, we first present the threat model, and then introduce the inconsistency between user identities and accounts, which generates potential vulnerabilities and might lead to serious security consequences.

3.1 Threat Model

As email address has been widely adopted by most SPs to identify an account for their online services, we assume that users also utilize their email addresses for account registration on SPs. Such an email address could belong to a public email account in email providers, such as *Gmail* and *Hotmail*, or a business account (like *Google G Suite*) paid by the user's organization in its own domain. The organization can further use IAM systems for email and identity management. The email provider provides both email and identity services so that users can use SSO to authenticate accounts provided by SPs. The accounts on SPs can be either registered directly on SPs or through SSO.

In both cases (i.e., public email and business email), email address reuse could happen when an email address previously held by a user now belongs to another user. We present a detailed analysis of this reuse's possibilities and feasibilities in Section 4. For public emails, the email provider might delete or recycle unused email accounts, which could be later registered by other users. For business emails, it happens more frequently as the enterprise often deletes the corresponding email accounts after employees depart.

The email address reuse might occur coincidentally: (1) A new user registers the email address simply based on the email naming convention. For example, it uses the initial letter of the first name and the last name as the username of the mailbox (i.e., the part before the @ symbol). (2) A user modifies the email address for particular reasons, such as divorcing or getting married, as the user might change the last name in the email address. (3) Many organizations like universities allow users to set up an alias email address by their own choices, which increases the possibilities of email address reuse. While the alias email cannot be used for identity, it can be used for registering accounts on SPs. Once the alias is changed, the expired address might be reused by other users as the primary email address.

The email address reuse might also occur intentionally: adversaries register or rename their accounts to one expired email address left by the victim in order to compromise the associated accounts on SPs. For example, malicious employees or students could have a motivation to revive expired email addresses that belong to their former colleagues or classmates. Adversaries could also reuse deleted emails from publicly available database leaks (i.e., the Use-After-FreeMail problem [19]).

Our threat model does not require adversaries to have special technical skills other than gaining reused email addresses from victims. To further verify the online accounts (on SPs) linked with the reused emails, adversaries can actively examine targeted SPs to check whether there exist online accounts associated with the reused email addresses, or they can regularly check inbox for any delivered emails from targeted SPs. Adversaries can also utilize existing tools that provide the service of listing all accounts associated with an email address. For example, if adversaries obtain reused email addresses hosted by either Google or Microsoft, they can easily obtain a list of accounts associated with the reused email

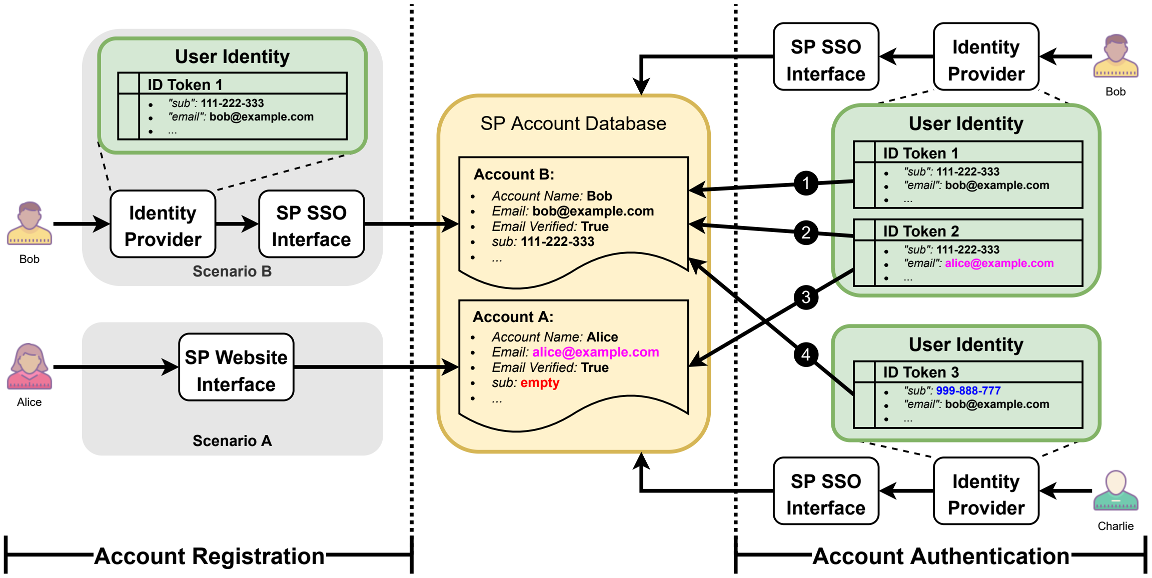

Figure 1: Identity-Account Relationship and Inconsistency.

addresses through *Deseat.me* [20]. Once the assumptions become valid, adversaries can begin to authenticate the associated online accounts (belonging to the victim) through SSO.

Difference from Password Recovery Attacks. Reused email addresses can be directly used to hijack online accounts by simply resetting or recovering an account password [10, 11]. While many effective mechanisms have been proposed to defend against password recovery attacks, such as two-factor authentication [12] and additional knowledge [13], those defenses are ineffective for countering the identity-account inconsistency threat. With the SSO authentication, adversaries do not need to crack or recover passwords. Adversaries can legally use the SSO service provided by the IdP to hijack accounts on SPs (belonging to the victim) without knowing any credentials. The reason is that IdPs recognize the adversary as the only legitimate user and owner of the email address once the adversary obtains a recycled email address, while such a scenario (i.e., the email address is reused by another user) is unknown to SPs. The root cause is the inconsistency between the identity and account in the existing SSO systems, with the details presented below.

3.2 Inconsistency

Most SPs adopt an email address as a primary account identifier and an optional UserID for SSO authentication. When a user registers a new online account, SPs store a combination of both information as the identifier in its database. As illustrated in Figure 1, two possible scenarios could happen: (1) User Alice registers the account directly on the SP website. In this scenario, since SSO is not involved, the SP cannot access the information of the UserID. Thus, the SP stores the new account with a valid email address provided by the user (e.g., `alice@example.com`) as well as an *empty* UserID (as Account A in Figure 1). (2) User Bob registers the account by SSO for the first time getting access to the SP. Then, both UserID and email address (e.g., `bob@example.com`) from the provided identity are recorded (as Account B in Figure 1).

On the other hand, for the identity information, the UserID is a non-recycled, unique number (or a set of characters) assigned by the IdP when establishing the identity. It always contains a valid value when a user requests an SSO authentication to an online account (as the case of the ID Token 1 in Figure 1). However, the email address is allowed to be modified or even reused based on the account management policies implemented on the IdP side. In particular, if the email address is changed by the user, the IdP updates the email information but keeps the same original UserID (as the case of the ID Token 2 in Figure 1). But if the email address is deleted and then reused by another user, the IdP assigns a new unique UserID to the user for a newly established identity (as the case of the ID Token 3 in Figure 1).

The inconsistency occurs when a user requests an SSO authentication to an online account, because the email address change only takes place internally in the IdP server, and SPs are not aware of the modification. The SP searches its account database to locate a specific account with the matched information based on the user identity containing a UserID and an email address from the IdP. In general, the SP can safely permit the access to the recognized account without any security concerns given a match on both UserID and email address. However, for a partial match (either UserID or email address), depending on the system configuration, the SPs might grant access to a wrong user.

Figure 1 illustrates four possible identity-account relationships. Case ❶ represents a normal case, where Bob attempts to login through SSO with the identical *"sub"* (i.e., UserID) and *"email"* fields in the SSO token as his online account. In this case, both the user identity and online account maintain consistency. In case ❷, Bob first changes his email address in the IdP, and attempts to sign in with his new email address. The SSO token and the online account

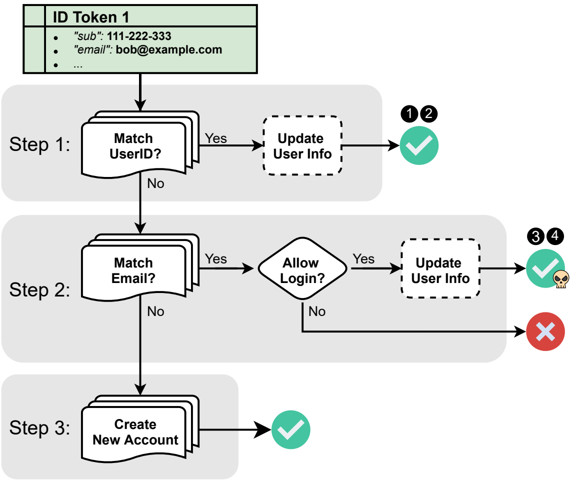

Figure 2: Overview of the Account Identification Procedure.

share the same *"sub"* but different *"email"*. In case ❸, Bob happens to change his email address to Alice's email. Since Alice registers her account without SSO, her online account holds an identical *"email"* with the SSO token but an empty *"sub"*. Instead, the SSO token contains a valid but unknown UserID. Another scenario is that Alice registers her account on SP using an email alias (`alice@example.com`). Once Bob obtains this email address as his primary email address, the inconsistency also occurs. Finally, in case ❹, the SSO token and online account have the same *"email"* but different *"sub"*. It happens when Bob deletes his email address in the IdP, and another user obtains this email address (e.g., `bob@example.com`) and reuses it in the IdP again.

The identity-account inconsistency occurs in cases ❷ ❸ ❹. In case ❷, though the inconsistency exists, it only affects the identity owner since the UserID cannot be re-assigned to others. However, in cases ❸ and ❹, the owner of the user identity is not guaranteed to be the same as the account owner due to the inconsistency. Thus, granting access in both cases results in a potential account compromise. Next, we detail how existing SSO systems handle such inconsistency.

3.3 Account Identification Flow and Problems

To understand the account identification in the existing SSO systems and the handling of the identity-account inconsistency, we describe the workflow of the OpenID Connect (OIDC) module [21], which is a popular open-source module enabling the Drupal platform to process user identities from several popular IdPs, including Facebook, Google, and LinkedIn. Note that different systems might have a different implementation on handling the inconsistencies. We present a detailed measurement study on 100 popular SPs in Section 5.

Figure 2 shows a detailed procedure of the account identification method. When an SP receives a user identity from a trusted IdP, the SP attempts to identify an existing account linked to the given identity. If no such account exists, SP begins the process of creating a new account for the user identity. This procedure includes three major steps:

Step 1. The first step is to identify an existing account associated with the given user identity. It first checks the UserID (stored in the *"sub"* field) to search for a matching account. Once a matching account (cases ❶ and ❷ in Figure 1) is recognized, the system performs a configuration check to determine if updating user attributions with a matching UserID is allowed. If it is permitted, SPs revise the information stored in the user account and modify the outdated information to align with the user identity. For example, in case ❷, the email address in the account database might be updated according to the identity information in the SSO token. Finally, the user authentication succeeds, and the user is granted access to the matching account, regardless of whether the user information can be updated or not.

Step 2. If there is no matching account found with the given UserID, the system performs another search for a potential match on the *"email"* field. However, since there is no matching account on UserID, any existing account with a matching email address should contain an empty (case ❸) or different *"sub"* field (case ❹). Dependent on the system configuration, the user might be granted access or denied to the identified account. If granted, the system performs another update on user attributions if it is allowed by system configuration. If denied, the user authentication fails, and an error message is shown on the website.

Step 3. If no account is found in either Step 2 or Step 3, the system begins the procedure of establishing a new account. The system creates a new account with the *"sub"* and *"email"* provided in the user identity. The system also asks the user to input additional account information if it is not provided by the identity.

Handling the Inconsistency. While the account matching procedure above may seem logically sound at first glance, a more in-depth analysis reveals that it may expose a vulnerability to account compromise attacks. Especially, it ignores the fact that email might be reused by different end-users. As a result, an email address, as an element of user attributions, cannot adequately represent the identity of an end-user. Granting access to an account with a matching *"email"* may end up with wrong users or even adversaries who attempt to compromise the victim's account. Furthermore, when an SP performs the user information update, it may also overwrite the *"sub"* field of the matched account. From the SSO authentication point of view, this process indicates that the matched account officially changes its associated user identity.

Secure vs. Insecure. For the inconsistency case ❷, OIDC grants user access since the UserID in the user identity remains the same as the *"sub"* field in the account. Some other systems (discussed in Section 5) instead guide the user to create a new account. In both scenarios, it might cause inconvenience for the user, but not lead to account compromise. Thus, we consider case ❷ as a **secure** policy. However, for cases ❸ and ❹, if the email address is reused by another user, this different user can gain access to the victim's online accounts. The skull mark in Figure 2 denotes both cases ❸ and ❹ as **insecure** implementations.

SAML. Security Assertion Markup Language (SAML) is a widely used open standard adopted by many IdPs like Google, allowing IdPs to pass authorization credentials to SPs. We also investigate the open-source SAML implementation in the Drupal platform. The SAML protocol allows an email address being used as a valid UserID. As a result, IdPs such as Google use the users' primary email

Account Type	**Email Provider**	**Default Naming Convention**	**Alias**	**Change**	**Delete (User)**	**Delete (System)**	**Reuse**	**Probation**	**Inactive Period**	**Disable**	**Price**[1]
Public	Gmail	User Defined[2]	✓	✗	✓	✗	✗	2-3 weeks	N/A	N/A	Free
	Hotmail	User Defined	✗	✗	✓	✓	✓	N/A	1 year	N/A	Free
	Yahoo!	User Defined[2]	✗	✗	✓	✓	✓	40-180 days	1 year	N/A	Free
	QQ	System Generated	✗	✗	✗	✓	✓	N/A	10 months	N/A	Free
Business	MS Office 365	User Defined	✗	✓	✓	✗	✓	30 days	N/A	Yes, paid	$5-20
	Google G Suite	Yes ({FN})[3]	✗	✓	✓	✗	✓	N/A	N/A	Yes, paid	$6-25
	Zoho Mail	Yes ({FN}.{LN})[3]	✗	✓	✓	✗	✓	N/A	N/A	Yes, paid	$1-4
	Amazon WorkMail	User Defined	✗	✓	✓	✗	✓	N/A	N/A	Yes, paid	$4

[1] Price unit: /user/month. [2] Email addresses must meet certain length requirement. [3] {FN}: First Name; {LN}: Last Name.

Table 1: Account Management Policy of Popular Email Providers for Different Account Types.

addresses as their default UserIDs [22]. This eliminates cases ❶ and ❷ shown in Figure 1, but still leaves the inconsistency cases ❸ and ❹, which are both insecure policies.

4 FEASIBILITY OF EMAIL REUSE IN IDPS

Since each identity managed by an IdP receives a unique and non-reusable UserID, the primary cause of the identity-account inconsistency is due to the modification of user attributions, especially the user's email address. To investigate the feasibility of two distinct user identities containing the same email address, we examine the account management policies adopted by both Email Providers and IAMs.

4.1 Email Providers as an IdP

Most email providers serve two major customer groups: general public and business enterprises. The account management policies for these two groups differ significantly: public accounts are mainly managed by email providers, but the business administrators have more capabilities of managing their business accounts. In our study, we select four popular email providers for both public and business accounts with built-in SSO capabilities to study their account management policies. Specifically, Gmail, Hotmail, and Yahoo! account for 85% of the email market in USA [14], and QQ mail supports more than 800 million active users in China [15]. The result is listed in Table 1.

Email Creation. All four public email providers offer free-of-charge email account services, and they adopt a one-to-one relationship between user identity and email address. The system generates a user identity when a new email address is registered, and removes the corresponding user identity when an email address is deleted. When registering email accounts, Gmail, Hotmail, and Yahoo! allow users to define their preferred email addresses as long as the addresses are not taken by others or against their naming requirements. By contrast, QQ assigns a unique number to every account, and the number itself is used as the username of the email address.

On the other hand, business accounts receive more flexibility. In general, account administrators can assign any email address to the user's account in their domains. Some email providers also implement built-in naming conventions to ease the account registration process. Compared with public accounts, business accounts further allow the administrators to temporally disable a user account. If an email is disabled, the user cannot receive more services, but the identity information is still kept in the database.

Change of Email Address. It is common that end-users request to modify their email addresses. For example, when people change their legal names, they prefer to reflect this change on their email addresses as well. For public accounts, none of the four email providers allow the direct modification of email addresses on user accounts. The only way is to create new email accounts. For business accounts, however, modifying email addresses is a rather simple and straightforward process. The account administrator can directly modify the email address in the user's setting page, and the change takes place instantaneously.

Deletion of Email Address. There are two possible scenarios for deleting an email account: deleted by the user/admin or deleted by the system. Most public account providers (except QQ in our experiments) allow end-users to delete their accounts. Providers such as Gmail and Yahoo! further adopt a small probation period prior to deletion in case users would like to recover them back. Hotmail deletes an email account along with all the corresponding user data immediately. Public email accounts may also be deleted by the provider due to inactiveness. Hotmail, Yahoo!, and QQ have explicitly stated in their websites that they regularly clean up inactive accounts within a period of time, and the deleted email addresses can be re-acquired by others.

For business accounts, administrators are fully responsible for removing any unused or redundant accounts. Since enterprises pay for each existing email address under their domain, email providers maintain the accounts regardless of their status (active or disabled). As for the probation period, only Microsoft 365 has a 30-day recovery policy; other email providers remove the account instantaneously.

Reuse of Email Address. The process of email creation and address modification also involves reusing a previously owned email address. Our study indicates that Hotmail and Yahoo! allow the same email address to be re-registered by other users. While QQ assigns an account number as an email address, the same number can still be claimed if the previous account is deleted. By contrast, Gmail prohibits the reuse of public email addresses. For business accounts, once an email address is deleted, it immediately becomes available for re-assignment. Notably, none of the four email providers remind the administrators that the email address has been used before.

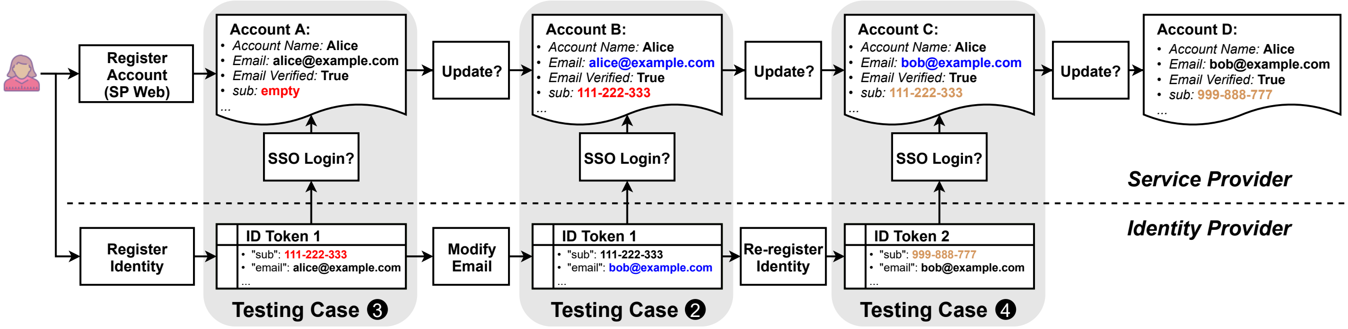

Figure 3: The Detailed Experiment Procedures for Understanding Account Matching in Popular SPs.

Summary. Overall, the account management policies adopted by both public and business accounts can potentially lead to an identity-account inconsistency. Neither end-users nor business administrators are aware of whether an email address has been previously used by others or not. Thus, once multiple individuals share the same preference on an email address, with the permitted reuse of email address, it is entirely possible for an end-user to have an email address that is previously owned by others. Although the email system can ensure a new identity and mailbox for this user, public SPs do not know the change of user identity caused by email address reuse. Similarly, permission to change a user's email address could also cause a problem: the administrator might assign a previously used email address to another user, who can further exploit the identity-account inconsistency to take control over the victim's online accounts.

Moreover, some policies might increase the occurrence of identity-account inconsistency. For example, the regular deletion of email accounts pollutes the pool of email addresses available for users. Since every business account is charged for a user-based monthly subscription fee (ranges from $1 to $25 per user, including disabled users), this billing method incentivizes account administrators to delete unused email accounts for cost reduction. Another critical factor is the naming convention. The system's default naming conventions suggest a universal format for all email addresses under one business domain. As a result, users might compete for the same email address if they share same attributions, such as their first or last names. We present a detailed discussion on naming conventions in Section 6.

4.2 IAM

To study the management flexibility and capability, we manually investigate five open-source IAM systems (i.e., OpenIAM, Keycloak, Gluu, Soffid, and feKara), including one system specifically designed for educational institutions (i.e., feKara). In general,the IAM system does not contain any restrictions on the account management policies, allowing the business administrators to fully customize them based on their own needs. An identity with basic user information and attributions (e.g., full name, position, and affiliation status) is initially created from the HR system or the admission system. The IAM generates the user identity based on the account creation policy, establishing new accounts on internal resources. For example, the IAM system registers an email address on a predefined internal or external email provider for a user. The email address is generated according to the naming convention that is predefined by the administrators.

We first run as an administrative account to test relevant features on email reuse. For example, we delete an existing account and then attempt to create a new account with the exact same email address immediately. We also observe that the administrator can configure the system allowing individual users to modify their information, including email addresses. Thus, we further test email reuse from a user's perspective. Overall, all five IAM systems allow account reuse (including email address) without notifying that the email address has been used before. Unlike email providers that might adopt a small probation period after deleting the email, all operations in these five IAM systems become valid immediately. If enabled, users can also change their email addresses to any available addresses.

5 ACCOUNT COMPROMISE IN THE WILD

To further investigate the possibility of compromising online accounts by exploiting the account-identity inconsistency vulnerability, we conduct a systematic study on 100 randomly selected popular SPs from the Alexa top 1,000 websites. We first establish a business email service on Google G Suite with our own domain name, which provides us the ability to add, modify, and delete user identities without any account management restrictions. Meanwhile, our domain name has never been registered by other organizations, ensuring that none of the SPs is hosting accounts associated with our testing email addresses. We then use the identity with an email address in our own domain to test the inconsistency cases ❷❸❹.

Figure 3 illustrates the whole procedure. Since each website incorporates its unique mechanism for user account registration and SSO authentication, in our experiment we manually investigate our selected SPs in batch. We first test case ❸, and use the related accounts to further test other cases. We create a user identity *Alice* with email address `alice@example.com` in the business email service provided by Google G Suite. This step creates a unique UserID (e.g., "`111.222.333`" in the *"sub"* field) for Alice. We then use the email address to register a new account on the target SP via the web interface (i.e., without SSO). As introduced in Section 3, registration via the web interface will establish an account with an *empty* UserID. However, the account in the target SP and the identity in the IdP share the same email address. Finally, we attempt to SSO login to the target SP using the identity in the IdP. If the login succeeds, we further check whether the target SP updates the stored account information based on the identity from SSO login. Particularly, we log in to Account A to check the SSO configuration in the account setting. If the Google account information appears as an authorized authentication method, we consider that the *"sub"* field is successfully updated.

We further test case ❷. On the IdP side, we modify the email address of the identity to `bob@example.com`. The UserID remains intact (i.e., still `"111-222-333"`). On the SP side, if the account information is updated in the previous step, the account should have the same UserID, but with an email address `bob@example.com`. Otherwise, we manually bind the account in the SP with the identity to update the related information. We then check whether the SP allows SSO login with the same UserID but a different email address. Again, if success, we check whether the account information is updated or not (by checking the email address).

Finally, we test the inconsistency case ❹. We delete the identity (i.e., the one with UserID `"111-222-333"`) on the IdP side, and register another identity reusing the same email address (i.e., `bob@example.com`). This generates another identity with a new UserID (e.g., `"999-888-777"`), but with all other information kept the same as the previous one. On the SP side, similar to testing case ❷, we make the account to have the same email address, but with a different UserID. We authenticate to the same SP by SSO and check if the login is allowed and if any user information is updated.

Notably, this case might modify the UserID on the SP side. If the UserID is modified, it is dangerous as the account now belongs to a new identity. If not, it might be even worse because two identities might be able to SSO login to the same account. We further conduct experiments to verify the update.

Results. According to our discussion in Section 3.3, SPs are vulnerable to the identity-account inconsistency threat if they allow SSO login for cases ❸❹. Otherwise, we consider they are not vulnerable since cases ❶❷ do not compromise accounts. Table 2 summarizes our results. Only 20 out of 100 SPs survive from the inconsistency threat (listed as Policy Types 3, 4, and 7). 79 out of 100 SPs are vulnerable to case ❸, which means they allow any identity with the same email address to SSO login to an account with *empty* UserID. Once the login succeeds, all those SPs update the *empty* UserID and bind the account to the identity. 52 SPs allow SSO login for case ❹: even with a different UserID, an identity with the same email address can SSO login to the SP's account, which clearly does not follow the SSO design specification. Besides, none of them update the UserID field.

For case ❷, 30 SPs deny the access of SSO login with a matching UserID but a different email address. Specifically, the SP might (1) only display an error page; or (2) ask the user to log in with the previous account to further modify SSO configuration; or (3) guide the user to register a new account. This does not follow the SSO specification and brings inconveniences to users who change their email addresses in the IdP. For the rest 70 SPs allowing SSO login with a matching UserID but a different email address, only 3 of them update the email address information. It might cause other problems. For example, a new user reuses the deleted email address might be able to recover password directly from the SP [11, 19].

Policy Type	Case ❷		Case ❸		Case ❹		Number of SPs
	Login	Update	Login	Update	Login	Update	
1	✓	✗	✓	✓	✓	✗	41
2	✓	✓	✓	✓	✓	✗	2
3	✓	✗	✗	✗	✗	✗	15
4	✓	✓	✗	✗	✗	✗	1
5	✓	✗	✓	✓	✗	✗	11
6	✗	✗	✓	✓	✓	✗	8
7	✗	✗	✗	✗	✗	✗	4
8	✗	✗	✓	✓	✗	✗	17
9	✗	✗	✗	✗	✓	✗	1
Total	✓:70 ✗:30	✓:3 ✗:97	✓:79 ✗:21	✓:79 ✗:21	✓:52 ✗:48	✓:0 ✗:100	100

Table 2: Account Matching Policy Adopted by Service Providers. Highlighted Types are Not Vulnerable to the Account-Identity Inconsistency Threat.

Our result also reveals another security problem caused by the identity-account inconsistency threat: two distinct user identities compete for a single online account. It exists when the SP allows SSO login for both cases ❷ and ❹ but without updating the related account information. In this case, as illustrated in Figure 3, an identity matching either the UserID (i.e., *"sub"*) or the email address can successfully SSO login. In total, we find that 41 out of 100 SPs (Type 1 in Table 2) are affected. We further conduct experiments on all those 41 SPs and confirm that both identities can SSO login to the account.

To summarize, we observe that more than half of popular SPs are vulnerable to the threat, which could be exploited to own a reused email and then compromise the victim's accounts via SSO authentication.

Ethics and Disclosures. Our experiment on SPs does not impose any ethical issues since the experiment is conducted in a controlled environment in which all user accounts are established by ourselves. We use the services provided by SPs as normal and legitimate users. However, our result indeed discloses that many SPs are potentially vulnerable to the identity-account inconsistency threat.

In addition, most of our experiments are conducted manually except for the batch investigation for our selected SPs. We have no intention of developing any automatic tools in our experiments because attackers may benefit from these tools to explore identity-account inconsistency in a more efficient and effective manner.

Such an inconsistency can be easily exploited by attackers with primitive technical skills. Therefore, for ethical considerations, we disclosed our findings to SPs affected by this threat. We received positive feedback from them who acknowledged the vulnerability, and some of them are actively working on implementing proper solutions.

6 POSSIBILITY OF EMAIL ADDRESS REUSE

In this section, we investigate the popular naming conventions adopted by the public to uncover why email address reuse could happen among users. We also present a case study on the account management policies, including the email naming convention adopted by many U.S. universities.

6.1 Email Naming Convention

Naming conventions serve as the primary guideline for people to choose email addresses. Moreland [23] suggested that the design of

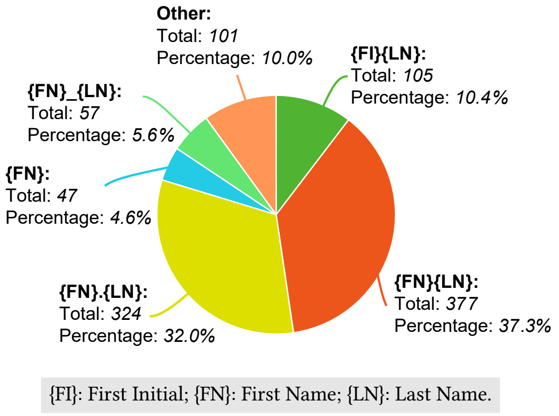

Figure 4: Fortune 1000 Email Naming Convention.

naming conventions should consider four critical aspects: *usability*, *security*, *administration*, and *audit*. Particularly, *usability* is the top concern for end-users.

Name-based convention is one of the popular email formats to achieve excellent usability [23]. A typical name-based convention consists of a user's initials, full name, and several digits of arbitrary or sequential numbers if necessary [24]. Many websites [25–27] publish various email naming conventions to provide guidelines for the public and businesses to select their preferred email address formats. Based on those reports, we summarize the most recommended email naming conventions (note that we use {} for better visual effects) as follows:

- {Firstname}@example.com
- {Firstinitial}{Lastname}@example.com
- {Firstname}.{Lastname}@example.com
- {Firstinitial}.{Lastname}@example.com

Large enterprises or organizations often adopt consistent naming conventions for their employees and affiliated members, enabling a fully automatic management system. We conduct an analysis on a database with email conventions endorsed by 971 businesses in the Fortune top 1000 company list in 2012 [28], and our result is shown in Figure 4. Note that some companies might adopt multiple naming conventions. The result indicates that the top five name-based email conventions count for 89.6% of the total formats. The top two most popular naming conventions, {Firstname}{Lastname} and {Firstname}.{Lastname}, are adopted by 661 organizations, which counts 68.1% of the total number of companies.

6.2 Email Address Collision

Given a large amount of population and only a few naming conventions, it is highly likely that different employees within an enterprise may compete for the same email address. To estimate the probability of email address collision at the organization/enterprise level, we investigate the employee history of the Metropolitan Transportation Authority (MTA) of New York State from its published employee payroll record [29]. The record provides a list of employees between 2012 and 2018. Table 3 presents the statistic information of the record, including the total number of employees and the number of employees who are retired or hired in each year. For example, in 2018, MTA had 80,147 employees (a reasonable size for many large enterprises), with 4,579 people retired from the company and 5,329 people recruited. The hire/retire information allows us to infer the possibility of email addresses reuse.

We assume that MTA uses the most recommended email naming conventions as discussed above. We first measure the probability for multiple employees receiving an identical address, and then present the possibility under different naming conventions for the year 2018 in Figure 5 (other years have similar results with less than 2% differences). Unsurprisingly, the {Firstname} convention has the highest probability: an astonishing 31% of the assigned emails would have a collision with 87% of the employees being affected. For example, there are 2,256 employees who have the same first name, Michael. For the {Firstinitial}{Lastname} case, about 18% of email addresses would collide, affecting 45% of employees. Even for the case {Firstname}{Lastname} using both firstname and lastname, it is still highly possible for the occurrence of email address collision: 6% of assigned email addresses would have a collision, affecting 12% of the employees. The results demonstrate that email address collision is inevitable in large enterprises given the current email naming conventions.

Next, we investigate the possibility of email addresses reuse due to the retire/hire procedure. We assume that an email address is released when an employee is retired, and it can be then reassigned to a newly hired employee in later years. In 2013, 4,592 employees were retired from MTA, which could cause the release of 1,341 email addresses for the case {Firstname}, 4,301 for the case {Firstinitial}{Lastname}, and 4,556 for the case {Firstname}{Lastname}. More than 5,000 employees joined MTA in each year from 2014 to 2018. We then study the probability of those released email addresses in 2013 to be reused by newly recruited employees from 2014 to 2018. Figure 6 illustrates the results. We can see that, at the end of 2018, 48.14%, 57.71%, and 65.77% of the email addresses could have been re-assigned for the cases {Firstname}{Lastname}, {Firstinitial}{Lastname}, and {Firstname}, respectively. Even for the naming convention involving both {Firstname} and {Lastname} that has the least chance of collision, 2,193 out of 4,556 email addresses could have been reused over six years. These results clearly indicate that email reuse in enterprises is highly possible, and those retired employees are vulnerable to the identity-account inconsistency threat.

6.3 Case Study

We further conduct a case study on the account management policies for students of U.S. universities, as most universities publish their policies online. We randomly select 50 universities, including public/private and national/regional universities with the student population ranging from thousands to tens of thousands.

Table 4 summarizes our survey. Among the 50 universities, three institutions host their own email servers (listed as N/A). Since we do not have further information on whether their email servers offer IdP services or not, we exclude these institutions from our result analysis. Other universities use either Google G Suite or Microsoft 365 as their email providers. We observe that many institutions adopt similar practices, and thus we categorize them into different types.

Year	Employee	Retire	Hire
2018	80,147	4,579	5,329
2017	79,397	5,129	6,914
2016	77,612	6,153	6,139
2015	77,626	6,882	6,887
2014	77,621	2,233	7,490
2013	72,364	4,592	5,074
2012	71,882	N/A	N/A

Table 3: MTA Employee Status Between 2012 and 2018.

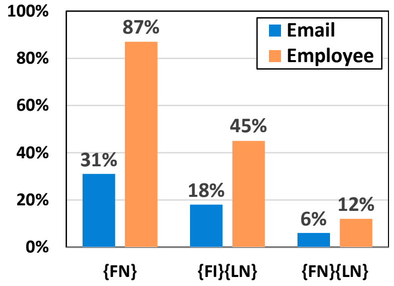

Figure 5: Potential Occurrence of Email Address Collision in 2018.

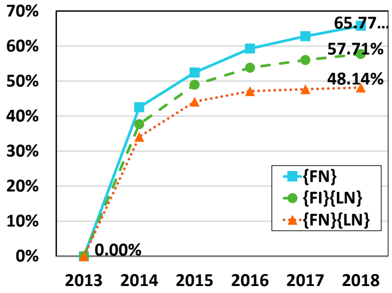

Figure 6: Potential Address Collision due to Retirement and Recruitment.

Among the rest 47 universities, 32 of them (types 1, 2, 3) allow students to change their email addresses by changing the NetIDs. Although most universities strongly discourage students to modify their email addresses without definite excuses, students may still choose to do so for various reasons, such as changing their legal names due to marriage or divorce. As a result, for those universities, students could reuse others' email addresses. In particular, 11 of them (type 2) delete students' email accounts immediately upon graduation, and two of them (type 6) maintain their email accounts unless the accounts become inactive. It further increases the possibility of email address reuse as those previously-used email addresses are available to newly admitted students. A student's identity generated by the university's email provider contains a distinct UserID and a recycled email address. Furthermore, eight universities (type 3) allow users to select their own email addresses without specific naming conventions. Such a policy offers students opportunities to intentionally obtain reused email addresses.

We also find that some universities (type 6) support alias email addresses. While they do not delete email accounts and changing email addresses is disallowed, alias email makes the identity-account inconsistency possible as students might use aliased email to register accounts in SPs. Only 13 universities (types 4, 5) are safe since they neither allow students to modify their email addresses, nor support alias email, and their email accounts remain active even after their students have been graduated.

We also check their naming conventions. We list the format as "System Assigned" for universities that do not explicitly mention the policy. Most of those universities apply name-based conventions to assign email addresses to their students. This result indicates that the identity-account inconsistency can indeed occur based on the existing email account management policies in U.S. universities.

7 MITIGATION

In order to mitigate the identity-account inconsistency threat, we propose several defensive practices. These methods aim to reduce the occurrence of identity-account inconsistency, as well as to prevent users from compromising a victim's accounts when the inconsistency takes place.

The identity-account inconsistency itself is complicated, involving both IdPs and SPs. Fully addressing it might require coordination between IdPs and SPs, which are usually from different enterprises. Besides, there is no universal implementation of the SSO authentication and email management systems. Thus, the defense practices introduced here are primitive and preliminary. The management/technical teams of the affected SPs should enforce appropriate defensive policies/mechanisms that best fit their system architectures and business needs.

Practice 1: *End-user should only use long-term identity to establish online accounts.* Long-term identities include IdP accounts that are possessed by the end-users themselves and are supposed to maintain usage for a long period of time, preferably lifetime. This ensures that the identities associated with their online accounts do not get expired and re-assigned to others. End-users can avoid using an organization's email address for registering online accounts on SPs. They should also pay extra attention to the account management policies so that they can avoid using temporary identities to establish accounts.

Practice 2: *When an identity is scheduled for removal, end-user should delete all associated accounts and erase all private data stored before the identity deletion date.* Despite the fact that many advanced security features have implemented in SSO, the identity-account inconsistency threat may continue to exist without users' awareness. Once their accounts are compromised, their personal information and private data will be exposed. Users should pay extra attention to those events that can result in the modification or deletion of their identities, and erase their private data and manually terminate all associated accounts.

Since SPs are responsible for identifying the account associated with provided identities, any logic flaws in the account identification procedure may introduce potential vulnerabilities. Thus, ensuring the authentication of a user to a specific account can effectively reduce the attack surface for adversaries to compromise online accounts.

Practice 3: *SPs should update user attributions, especially the email address, when an end-user gains access to an account with a matching UserID.* Email address is considered as one field of user attributions within an identity. SPs should also reflect the modification of user attributions on the user account. Thus, a modified email address in a user's identity should be able to overwrite the existing primary account identifier in the user's account. This also prevents a potential password recovery attack [11, 19] if the outdated email address is re-acquired by other users.

Type	Provider	NetID	Email Format	Change	Alias	Delete	Total
1	GSuite	✓	User Defined	✓[1]	✗	✗	8
2	GSuite	✓	(initials and #)	✗	✓	✗[2]	2
3	O365	✗	(initial, lastname, #)	✗	✗	✗	7
4	O365	✓	(firstname.lastname)	✓[1]	✗	✗	13
5	O365	✓	Assigned	✓[1]	✗	✓	11
6	O365	✓	Assigned	✗	✗	✗	6
7	N/A	✗	(firstname.lastname)	✗	✓	✓[2]	3

[1] By changing the NetID. [2] Must maintain active.

Table 4: Account Management Policy for 50 Universities.

IdPs are responsible for enforcing a secure baseline of administration policies for both public and business accounts. This baseline should be publicly available in order for the SP system to have a secure and robust implementation.

Practice 4: *IdPs should not allow the reuse of email addresses.* One of the primary causes of the identity-account inconsistency threat is the re-assignment of an email address. A simple and straightforward way of mitigating this threat is to prevent any distinct identities in IdPs from receiving the same email address. Such a practice will increase the naming entropy of an email account, while the usability could be balanced by allowing more flexible email naming conventions.

8 RELATED WORK

Account Security. Account security has been extensively studied for decades. Since a massive amount of sensitive information is stored in user accounts [11, 30], compromising accounts can lead to serious consequences. Email addresses have further attracted significant research attention. Hu et al. [31] presented that email spoofing can be used as an effective method for launching phishing attacks by impersonating trusted identities. Cidon et al. [32] presented the Business Email Compromise (BEC) threat to phish employees' private information and account credentials by impersonating management-level personnel. Although many works have been proposed to enhance account security by improving password strength [33, 34], many security vulnerabilities still exist, including password cracking [35, 36] and password recovery [11, 37]. Two-Factor Authentication (2FA) achieves strong account security by requiring users to provide additional authentication factors, such as biometrics [38], one-time password [12], or ambient sound [39]. However, it has its own limitation in usability [40]. Unlike previous works that compromise user accounts via password cracking or recovering, our proposed identity-account inconsistency threat shows that a normal reused email address can be exploited to compromise user accounts due to the security pitfalls in the SSO authentication scheme.

Reuse of Credentials and Identities. The reuse of credentials or identities can cause severe security problems. Hu et al. [41] recognized that online accounts registered through disposable email addresses, which are largely used by many people, could be easily hijacked for malicious purposes. Martindale [42] also demonstrated the possibility of compromising Facebook accounts by leveraging a reused cellphone number. Mariconti et al. [43] discovered the ability to impersonate Twitter users by adopting their profile names. Our work further complements these previous research efforts on understanding the security of reusing email addresses in the context of SSO authentication.

In particular, Gruss et al. [19] presented the Use-After-FreeMail problem. By reusing the deleted email addresses, attackers can compromise user accounts through password recovery. They showed that attackers can gather invalid email addresses from publicly available database leaks. Our work differs from them as we focus on the information inconsistency existed in the SSO systems. Also, our study shows that systems using both public and business email accounts suffer from the email reuse problem.

SSO Security. While both OpenID Connect [21] and OAuth 2.0 [44] describe some potential security problems and countermeasures, many SSO vulnerabilities and logic flaws have also been regularly reported. For example, intercepting HTTP cookies [45, 46] from the network has been one of the common attack methods to hijack user accounts. However, Chari et al. [47] showed that the SSO protocol is secure if a proper SSL encryption is applied to all communications. Gao et al. [48] illustrated the possibility of compromising user accounts through account binding. There are many other SSO vulnerabilities involved with network traffic [5], malicious IdP [7, 49], and software SDKs [50, 51]. Our work uncovers a new widely existing threat that allows adversaries to compromise a user's account. Meanwhile, it does not require adversaries to have particularly advanced skills or the capabilities to control the networking traffic to compromise the victim's accounts.

9 CONCLUSION

In this paper, we present the identity-account inconsistency threat in the SSO system caused by email address reuse. We demonstrate that, by reusing an email address, adversaries can compromise a victim's online account via the SSO authentication in several scenarios. We first show the feasibility for end-users to acquire previously used email accounts by studying the account management policies adopted by identity providers. Then, we explore the potential threats on 100 popular service providers and our results indicate that most online accounts can be compromised by exploiting the identity-account inconsistency vulnerability. To further demonstrate that email address reuse is highly possible in real life, we analyze the probability of email address collision based on several commonly used email naming conventions. We also conduct a case study on U.S. institutions, showing that the threat indeed exists. Finally, we propose useful practices to mitigate the identity-account inconsistency threat.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their insightful and detailed comments, which helped us improve the quality of this paper. This work was supported in part by the U.S. Army Research Office (ARO) grant W911NF-19-1-0049, Office of Naval Research (ONR) grant N00014-20-1-2153, and National Science Foundation (NSF) grant CNS-2054657.

REFERENCES

[1] M Lane and M Marie. The Adoption of Single Sign-On and Multifactor Authentication in Organisations - A Critical Evaluation Using TOE Framework. *Information in Motion*, 7:161, 2010.

[2] Graham Hayday. Security Nightmare: How Do You Maintain 21 Different Passwords? *Silicon.com*, 11, 2002.

[3] David W Chadwick. Federated Identity Management. In *Foundations of security analysis and design V*, pages 96–120. Springer, 2009.

[4] Gail-Joon Ahn and John Lam. Managing Privacy Preferences for Federated Identity Management. In *Workshop on Digital identity Management*, pages 28–36, 2005.

[5] Rui Wang, Shou Chen, and Xiaofeng Wang. Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services. In *IEEE Symposium on Security and Privacy*, 2012.

[6] Chuan Yue. The Devil is Phishing: Rethinking Web Single Sign-On Systems Security. In *USENIX Workshop on Large-Scale Exploits and Emergent Threats*, 2013.

[7] Phili Hu, Ronghai Yang, Yue Li, and Wing Cheong Lau. Application Impersonation: Problems of OAuth and API Design in Online Social Networks. In *ACM conference on Online social networks*, 2014.

[8] Yuchen Zhou and David Evans. SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities. In *23rd USENIX Security Symposium*, 2014.

[9] Chaoshun Zuo, Qingchuan Zhao, and Zhiqiang Lin. AuthScope: Towards Automatic Discovery of Vulnerable Authorizations in Online Services. In *ACM Conference on Computer and Communications Security*, 2017.

[10] Simone Raponi and Roberto Di Pietro. A Longitudinal Study on Web-sites PasswordManagement (in)Security: Evidence and Remedies. 2019.

[11] Yue Li, Haining Wang, and Kun Sun. Email as a Master Key: Analyzing Account Recovery in the Wild. In *IEEE Conference on Computer Communications*, pages 1646–1654. IEEE, 2018.

[12] Fadi Aloul, Syed Zahidi, and Wassim El-Hajj. Two Factor Authentication Using Mobile Phones. In *IEEE/ACS International Conference on Computer Systems and Applications*, pages 641–644. IEEE, 2009.

[13] John Brainard, Ari Juels, Ronald L Rivest, Michael Szydlo, and Moti Yung. Fourth-factor authentication: somebody you know. In *ACM Conference on Computer and Communications Security*, 2006.

[14] The Most Popular Email Providers in the U.S.A. https://blog.shuttlecloud.com/the-most-popular-email-providers-in-the-u-s-a/.

[15] QQ monthly active users. https://statstic.com/qq-monthly-active-users/.

[16] Eve Maler and Drummond Reed. The Venn of Identity: Options and Issues in Federated Identity Management. *IEEE Security & Privacy*, 2008.

[17] 12 Best Practices for User Account, Authorization and Password Management. https://cloud.google.com/blog/products/gcp/12-best-practices-for-user-account.

[18] Simson L. Garfinkel. Email-Based Identification and Authentication: An Alternative to PKI? *IEEE Security & Privacy*, 2003.

[19] Daniel Gruss, Michael Schwarz, Matthias Wübbeling, Simon Guggi, Timo Malderle, Stefan More, and Moritz Lipp. Use-After-FreeMail: Generalizing the Use-After-Free Problem and Applying it to Email Services. In *Asia Conference on Computer and Communications Security*, 2018.

[20] Deseat.me. https://www.deseat.me/.

[21] OpenID Connect Core 1.0 incorporating errata set 1. https://openid.net/specs/openid-connect-core-1_0.html.

[22] Set up your own custom SAML application. https://support.google.com/a/answer/6087519?hl=en.

[23] Troy Moreland. Definitive Guide to Account Username Conventions. *Identity Automation*.

[24] R. Witty and A. Allan. Best Practices in User ID Formation. *Gartner Research*, 2003.

[25] Email Naming Conventions. https://nathanives.com/email-naming-conventions/.

[26] 3 Rules to Choosing a Professional Email Address. https://fitsmallbusiness.com/professional-email-address/.

[27] How to Guess a Corporate Email Address. https://salesscripter.com/how-to-guess-a-corporate-email-address/.

[28] Fortune 1000 Companies List and Contact Info. https://booleanstrings.com/wp-content/uploads/2014/01/fortune1000-2012.xls/.

[29] Salary Information for State Authorities. https://data.ny.gov/Transparency/Salary-Information-for-State-Authorities/unag-2p27.

[30] Serge Egelman, Sakshi Jain, Rebecca S Portnoff, Kerwell Liao, Sunny Consolvo, and David Wagner. Are You Ready to Lock? Understanding User Motivations for Smartphone Locking Behaviors. In *ACM Conference on Computer and Communications Security*, pages 750–761, 2014.

[31] Hang Hu and Gang Wang. End-to-End Measurements of Email Spoofing Attacks. In *USENIX Security Symposium*, pages 1095–1112, 2018.

[32] Asaf Cidon, Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser, and Alexey Tsitkin. High Precision Detection of Business Email Compromise. In *USENIX Security Symposium*, pages 1291–1307, 2019.

[33] Serge Egelman, Andreas Sotirakopoulos, Ildar Muslukhov, Konstantin Beznosov, and Cormac Herley. Does My Password Go up to Eleven? The Impact of Password Meters on Password Selection. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 2379–2388, 2013.

[34] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. Password strength: An empirical analysis. In *IEEE International Conference on Computer Communications*, pages 1–9. IEEE, 2010.

[35] Enze Liu, Amanda Nakanishi, Maximilian Golla, David Cash, and Blase Ur. Reasoning Analytically About Password-Cracking Software. In *IEEE Symposium on Security and Privacy*, pages 380–397. IEEE, 2019.

[36] Jeremiah Blocki, Benjamin Harsha, and Samson Zhou. On the Economics of Offline Password Cracking. In *IEEE Symposium on Security and Privacy*, pages 853–871. IEEE, 2018.

[37] Christina Garman, Kenneth G Paterson, and Thyla Van der Merwe. Attacks Only Get Better: Password Recovery Attacks Against RC4 in TLS. In *USENIX Security Symposium*, pages 113–128, 2015.

[38] Andrew Teoh Beng Jin, David Ngo Chek Ling, and Alwyn Goh. Biohashing: Two Factor Authentication Featuring Fingerprint Data and Tokenised Random Number. *Pattern Recognition*, 37(11):2245–2255, 2004.

[39] Nikolaos Karapanos, Claudio Marforio, Claudio Soriente, and Srdjan Capkun. Sound-Proof: Usable Two-Factor Authentication Based on Ambient Sound. In *USENIX Security Symposium*, pages 483–498, 2015.

[40] Thanasis Petsas, Giorgos Tsirantonakis, Elias Athanasopoulos, and Sotiris Ioannidis. Two-factor Authentication: Is the World Ready? Quantifying 2FA Adoption. In *European Workshop on System Security*, pages 1–7, 2015.

[41] Hang Hu, Peng Peng, and Gang Wang. Characterizing Pixel Tracking through the Lens of Disposable Email Services. In *IEEE Symposium on Security and Privacy*, 2019.

[42] I kinda Hacked a Few Facebook Accounts Using a Vulnerability They Won't Fix. https://medium.com/hackernoon/i-kinda-hacked-a-few-facebook-accounts-using-a-vulnerability-they-wont-fix-2f5669794f79.

[43] Enrico Mariconti, Jeremiah Onaolapo, Syed Sharique Ahmad, Nicolas Nikiforou, Manuel Egele, Nick Nikiforakis, and Gianluca Stringhini. What's in a Name?: Understanding Profile Name Reuse on Twitter. In *International Conference on World Wide Web*, 2017.

[44] OAuth 2.0 Threat Model and Security Considerations. https://tools.ietf.org/html/rfc6819.

[45] Firesheep. https://codebutler.com/2010/10/24/firesheep/.

[46] Mohammad Ghasemisharif, Amrutha Ramesh, Stephen Checkoway, Chris Kanich, and Jason Polakis. O single sign-off, where art thou? an empirical analysis of single sign-on account hijacking and session management on the web. In *USENIX Security Symposium*, 2018.

[47] Suresh Chari, Charanjit S Jutla, and Arnab Roy. Universally Composable Security Analysis of OAuth v2.0. *International Association for Cryptologic Research*, 2011.

[48] Xi Gao, Lei Yu, Houhua He, Xiaoyu Wang, and Yiwen Wang. A Research of Security in Website Account Binding. *Journal of Information Security and Applications*, 2020.

[49] Christian Mainka, Vladislav Mladenov, and Jörg Schwenk. Do Not Trust Me: Using Malicious IdPs for Analyzing and Attacking Single Sign-On. In *IEEE European Symposium on Security and Privacy*, pages 321–336. IEEE, 2016.

[50] Ronghai Yang, Wing Cheong Lau, Jiongyi Chen, and Kehuan Zhang. Vetting Single Sign-On SDK Implementations via Symbolic Reasoning. In *USENIX Security Symposium*, pages 1459–1474, 2018.

[51] Rui Wang, Yuchen Zhou, Shuo Chen, Shaz Qadeer, David Evans, and Yuri Gurevich. Explicating SDKs: Uncovering Assumptions Underlying Secure Authentication and Authorization. In *USENIX Security Symposium*, pages 399–314, 2013.