Anonymous Tokens with Designated-Reader Metadata Bit

Aisha Tu[†], Meng Jia[‡], Kun He[†*], Jing Chen[†], Ruiying Du[†]
[†]*Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University*
{aishatu, hekun, chenjing, duraying}@whu.edu.cn
[‡]*Department of Computing, The Hong Kong Polytechnic University*
mengjia@polyu.edu.hk

Abstract

Anonymous tokens with private metadata bit convey hidden signals to verifiers when presented by the user and are under discussion in standardization. Existing solutions only allow the token issuer to read the signals, which places a heavy burden on the issuer and makes it challenging to support issuer-hiding because verifiers have to contact the issuer. In this paper, we propose an anonymous token scheme with designated-reader metadata bit, allowing the user to specify an issuer-accepted verifier to read the signal from the token directly. We also extend our scheme to support reader-hiding, which conceals the user's intended verifier from the issuer and other verifiers, and issuer-hiding, which prevents exposure of the token issuer from verifiers. We prove the security of our constructions and report their performance.

1 Introduction

Anonymous tokens [15, 18, 28] play a vital role in privacy-preserving identity and access management. In brief, an anonymous token propagates the trust established between an issuer and a user (e.g., through authentication) to a verifier who does not know the user but trusts the issuer. This process is privacy-preserving, meaning that the token presented by the user cannot be linked to the previous interaction between the user and the issuer, known as *unlinkability*. In addition to academic interest, there has been a significant industry advance recently in anonymous tokens, such as Google Trust Tokens[1] and Cloudflare Privacy Pass[2].

Issuers may hesitate to continue interactions when a user is identified as malicious. This is because refusing to produce tokens will convey a negative signal to the user, prompting it to improve the attack or retaliate against the issuer. One solution is to embed *private metadata* into a token to indicate whether the user is trustworthy, without the user being able to learn which signal the token conveys.[3] Then, the verifier decides on actions (e.g., something milder and more obscure than an outright rejection) based on the private metadata. Note that the metadata not only conveys signals but also partitions users into subsets, making it easier to violate the unlinkability of anonymous tokens. Therefore, the embedded metadata should be kept short, with a single bit being optimal [13, 29].

Due to the private nature of the metadata bit, even if the anonymous token scheme itself supports public verification of tokens, reading the metadata bit can only be done in a private manner. Existing solutions [2, 6, 13, 29] only allow the issuer to read the embedded bit. However, this design may cause two problems. On the one hand, the issuer needs to stay online to respond to reading requests from verifiers. It incurs additional overhead for the issuer and may cause a bottleneck when numerous requests are made simultaneously. On the other hand, it is inherently difficult to achieve *issuer-hiding* [7, 9, 17, 24, 31, 36], which is a common requirement of anonymous authentication that prevents the identity of the issuer from revealing personal information of the user. This difficulty arises because the verifier must contact the issuer to obtain the private metadata bit.

In this paper, we aim to design an anonymous token scheme with a metadata bit that a user-specified reader (aka verifier) can directly read. It is not difficult to see that the reader specified by the user must be the verifier to whom the issuer is willing to convey signals. For verifiers that the issuer does not wish to signal, we also expect the issuer to provide regular anonymous token services, which means that the tokens should support *public verification*. In our basic construction, the issuer and the reader know each other's identities. In some cases, the user may not wish to disclose the reader's identity, as it could reveal the destination they intend to access. To address this issue, we extend the basic scheme to support *reader-hiding*, where the issuer and other verifiers cannot learn the reader's identity associated with a token. In other cases, revealing the issuer's identity may expose users' orga-

*Corresponding author
[1] https://github.com/google/anonymous-tokens
[2] https://blog.cloudflare.com/privacy-pass-standard
[3] https://datatracker.ietf.org/wg/privacypass/about/

nizational or regional affiliations or subject them to discrimination. To tackle this issue, we extend the basic scheme to support *issuer-hiding*, where the issuer's identity is hidden from all verifiers.

Our contributions are summarized as follows.

- We introduce the notion of anonymous tokens with designated-reader metadata bit, called HinToken, that also supports public verification. We also formalize the security requirements of our scheme.
- We present an anonymous token scheme with constant-size tokens. Then, we extend it to support reader-hiding and issuer-hiding, respectively. Moreover, we design an instantiation of non-interactive zero-knowledge proof (NIZK) systems as an alternative solution to prove the legitimacy of blinded public keys.
- We prove the security of our constructions under the generic-group model (GGM) and evaluate their implementation performance.

1.1 Applications

To motivate our HinToken scheme and its two extensions (i.e., reader-hiding HinToken and issuer-hiding HinToken), we provide several useful applications as follows. In all applications, the metadata bit can be used by the issuer to convey hidden signals to the user-specified reader.

Bot Protection. When users visit a website, anonymous tokens can replace CAPTCHAs to determine whether an action is performed by a human or a bot. Existing solutions already support public verification (e.g., Apple Private Access Tokens[4]) or private metadata (e.g., WIGC Private State Tokens[5]). HinToken supports both features simultaneously, namely, conveying signals such as the risk assessment of a user being a bot to a specific website.

Proof of Funds. Issuer-hiding HinToken can be used as proof of funds in situations such as visa applications. The bank needs to know the reader (e.g., to clarify the purpose of the proof) and confirm that the bank account balance of the user exceeds a specified amount before issuing a token. The token can convey signals such as the user credibility or debt status. Users may prefer not to disclose which bank holds their deposits, thus requiring issuer-hiding.

Content Delivery Networks (CDNs). A streaming media owner (i.e., the issuer) may host their private content on various CDN servers (i.e., the readers) that support publicly verifiable tokens such as Google Media CDN[6]. The signals in tokens can also convey information such as the user credibility or category. Users may select the most suitable CDN server for their needs (e.g., based on their region) without revealing this choice to the issuer, thus requiring reader-hiding.

[4] https://developer.apple.com/news/?id=huqjyh7k
[5] https://github.com/WICG/trust-token-api
[6] https://docs.cloud.google.com/media-cdn/docs

1.2 Technical Overview

The trivial idea to support *designated-reader* is to employ existing solutions for the private metadata bit and have each reader share a secret key with the issuer. However, it requires complex key management. Moreover, it is impossible to achieve reader-hiding and issuer-hiding, since the shared key connects the issuer and the reader. To solve these problems, each reader should have their own public-secret key pair. Then, the issuer can encrypt the metadata bit using the reader's public key. The technical challenge here is to design the encryption method for anonymous tokens. Specifically, the ciphertext should be controlled-malleable, and the legitimacy of the underlying plaintext should be easy to prove. These requirements are essential for the unforgeability of the embedded bit and unlinkability of the token (see Section 4 for details).

Aiming at this goal, we leverage the re-randomization and random self-reducibility (for reader-hiding) features of Diffie-Hellman instances and present an El Gamal-like encryption method that matches Σ-protocols [8] and structure-preserving signatures on equivalence classes (SPS-EQ) [21]. Specifically, our ciphertext is of the form $(h, g^r, y^r h^s) \in \mathbb{G}^3$, where g is a generator of the group $\mathbb{G}$, y is the public key of the reader, s is the metadata bit, h is a random group element, and r is a random scalar. This ciphertext is controlled-malleable to $(h^z, g^{rz}, y^{rz} h^{sz})$ under SPS-EQ, where z is a random scalar. To ensure plaintext legitimacy, we present an instantiation of NIZK systems based on Σ-protocols.

A consequence of a specified reader is that the identity of the reader will be exposed to the issuer in the issuance protocol. To achieve *reader-hiding*, we require that the encryption method works under blinded public keys, which means that the reader can correctly decrypt the ciphertext obtained by a blinded key. In our method, $(h, g^{r\gamma}, y^{r\gamma} h^s)$ is the ciphertext of s under the public key y, where r and γ are two random exponents. This ciphertext can also be viewed as obtained under the blinded public key y^γ and the help value g^γ. This approach achieves reader-hiding only if the issuer does not have the secret key x of the reader; otherwise, it can check whether $(g^\gamma)^x = y^\gamma$. It is reasonable to assume that the issuer does not know x or is colluding with the reader. We also give a collusion-resistant solution in Section 8.

Another issue with the reader-hiding solution above is that the user must prove the legitimacy of the public key; otherwise, the user can provide their own public key and read the embedded bit. Since issuers use tokens to convey signals to readers, it is reasonable to assume that the issuer has a list of readers willing to signal. Therefore, the user needs to prove that the targeted reader's blinded public key is derived from a member of the acceptable public key set of readers. This approach provides the issuer with better control and flexibility to specify readers. We employ SPS-EQ to sign the generator and the acceptable public key, i.e., (g, y), and blind them to

(g^γ, y^γ) with the randomization algorithm of SPS-EQ. Since SPS-EQ is unforgeable, (g^γ, y^γ) is associated with a legitimate (i.e., signed) public key. We also provide an alternative solution to demonstrate the legitimacy of the blinded public key, based on a proposed NIZK system in Section 8.

With a similar assumption and idea, we can extend the basic scheme to support *issuer-hiding*. Specifically, the reader defines an acceptable public key set of issuers and signs on each public key in the set. The user can randomize the specific public key and signature associated with the issuer, and the reader checks the legitimacy by verifying the signature.

1.3 Related Work

Anonymous tokens with private metadata bit. To convey a signal of whether a user is trustworthy, Kreuter et al. [29] proposed to embed a metadata bit in the token privately. They defined the security properties of such anonymous token schemes and provided a construction PMBT based on verifiable oblivious PRF (VOPRF) [27]. Chase et al. [13] found that the definition of privacy of metadata bit in [29] is faulty and presented a stronger notion. They also presented a construction called ATHM via algebraic MACs [14]. PMBT and ATHM only support private verification, meaning that the issuer and the verifier are the same. Their applicability is limited because the issuer cannot provide services to others. Benhamouda et al. [6] presented a publicly verifiable anonymous token scheme with private metadata bit, which is built on top of blind Schnorr signatures [22]. Baldimtsi et al. [2] introduced non-interactive anonymous tokens (NIAT) with private metadata bit inspired by non-interactive blind signatures (NIBS) [26], which is also publicly verifiable. However, all of these schemes only support the issuer in reading the embedded bit, and thus cannot reach issuer-hiding.

Issuer-hiding anonymous credentials. In anonymous credentials, one approach to issuer-hiding is to allow the verifier to define a set of acceptable issuers and require the user to prove that the presented credential is issued by a member of the set [7, 9, 31, 36]. However, most solutions rely on NIZK systems, which usually incur high overhead. Delegatable anonymous credential (DAC) [12, 17, 24, 32] is another way to reach issuer-hiding. DACs enable a root issuer to delegate the credential-issuing power, and the intermediate issuers in a delegation chain can be hidden from verifiers. However, verifiers cannot determine (un)acceptable intermediate issuers in a DAC chain on demand. In Section 8, we will discuss how to leverage the idea of DACs to reach issuer-hiding.

2 Preliminaries

We use $\xleftarrow{\$}$ to represent sampling a value uniformly at random from a set and use $\mathbf{x} = (x_1, \cdots x_n)$ to represent a vector of group/field elements. For a vector $\mathbf{x}$ and a scalar z, let $\mathbf{x}^z = (x_1^z, \cdots, x_n^z)$. We denote a (multiplicative) cyclic group with prime order p by $\mathbb{G}$ and $\mathbb{G} \setminus \{1_\mathbb{G}\}$ by $\mathbb{G}^*$. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be three groups with prime order p, and g_1 and g_2 be the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. We say that $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ is a bilinear group if there exists an efficient map (called pairing) $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ that satisfies (1) bilinearity: $\forall u \in \mathbb{G}_1$, $w \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$, $e(u^a, w^b) = e(u, w)^{ab}$; (2) non-degeneracy: $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$.

2.1 Assumptions

Definition 1 (Decisional Diffie-Hellman)**.** Let $\mathbb{G}$ be a cyclic group with prime order p and $g \in \mathbb{G}$ be a generator. The decisional Diffie-Hellman (DDH) assumption holds in $\mathbb{G}$ if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, there is a negligible function negl such that

$$\left|\Pr[\mathcal{A}(g, g^a, g^b, g^c) = 1] - \Pr[\mathcal{A}(g, g^a, g^b, g^{ab}) = 1]\right| \leq \mathsf{negl}(\lambda),$$

where in each case the probabilities are taken over the experiment in which $a, b, c \xleftarrow{\$} \mathbb{Z}_p$.

Definition 2 (Inverse Decisional Diffie-Hellman)**.** Let $\mathbb{G}$ be a cyclic group with prime order p and $g \in \mathbb{G}$ be a generator. The inverse decisional Diffie-Hellman (Inv-DDH) assumption holds in $\mathbb{G}$ if for any PPT adversary $\mathcal{A}$, there is a negligible function negl such that

$$\left|\Pr[\mathcal{A}(g, g^a, g^b) = 1] - \Pr[\mathcal{A}(g, g^a, g^{a^{-1}}) = 1]\right| \leq \mathsf{negl}(\lambda),$$

where in each case the probabilities are taken over the experiment in which $a, b \xleftarrow{\$} \mathbb{Z}_p$.

2.2 Structure-Preserving Signatures of Equivalence Classes

Let $\mathbf{m} \in (\mathbb{G}_i^*)^l$ $(i \in [1,2])$ be a vector of group elements where $l > 1$. Structure-preserving signatures on equivalence classes [21, 25] are used to sign an equivalence class $[\mathbf{m}]_{\mathsf{R}_m} = \{\mathbf{n} \mid (\mathbf{m}, \mathbf{n}) \in \mathsf{R}_m\}$ with the following equivalence relation R_m:

$$\mathsf{R}_m = \{(\mathbf{m}, \mathbf{n}) \in (\mathbb{G}_i^*)^l \times (\mathbb{G}_i^*)^l \mid \exists z \in \mathbb{Z}_p^* : \mathbf{n} = \mathbf{m}^z\} \subseteq (\mathbb{G}_i^*)^{2l}.$$

Note that R_m is an equivalence relation if and only if $\mathbb{G}_i$ is a group with prime order. Since for any $\mathbf{m} \in (\mathbb{G}_i^*)^l$, the randomization $\mathbf{m}^z$ should look random in $(\mathbb{G}_i^*)^l$, which does not hold when $\mathbf{m}[i] = 1_\mathbb{G}$ for some $i \in [l]$, so we exclude the element $1_\mathbb{G}$ from $\mathbb{G}$.

Definition 3 (SPS-EQ)**.** A structure-preserving signature scheme for equivalence relation R_m on $\mathbb{G}_i$ consists of the following PPT algorithms.

- $(pk, sk) \leftarrow \mathsf{KeyGen}(\mathsf{BG}, 1^l)$. Given a bilinear group BG and a vector length $l > 1$ as input, the key generation algorithm outputs a key pair (pk, sk).

$\underline{\mathsf{EUF\text{-}CMA}_{\mathcal{A}}(\mathsf{BG},l)}$:

1: $(pk,sk) \leftarrow \mathsf{KeyGen}(\mathsf{BG},1^l)$
2: $Q \leftarrow \emptyset$
3: $(\mathbf{m}^*,\sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(\mathsf{BG},pk)$
4: **return** $\forall \mathbf{m} \in Q, [\mathbf{m}]_{\mathsf{R}_m} \neq [\mathbf{m}^*]_{\mathsf{R}_m} \wedge \mathsf{Verify}(pk,\mathbf{m}^*,\sigma^*)$

$\underline{\mathcal{O}_{\mathsf{Sign}}(\mathbf{m})}$:

1: $Q \leftarrow Q \cup \{\mathbf{m}\}$
2: **return** $\mathsf{Sign}(sk,\mathbf{m})$

Figure 1: Existential Unforgeability Game under Chosen Message Attacks for SPS-EQ

- $\sigma \leftarrow \mathsf{Sign}(sk,\mathbf{m})$. Given the secret key sk and a message vector $\mathbf{m} \in (\mathbb{G}_i^*)^l$ as input, the signing algorithm outputs a signature σ.
- $(\mathbf{m}',\sigma') \leftarrow \mathsf{ChgRep}(pk,\mathbf{m},\sigma,\mu)$. Given the public key pk, a message vector $\mathbf{m} \in (\mathbb{G}_i^*)^l$ and the corresponding signature σ, and a scalar $\mu \in \mathbb{Z}_p^*$ as input, the randomization algorithm outputs another representative $\mathbf{m}' = \mathbf{m}^\mu$ of the equivalence class $[\mathbf{m}]_{\mathsf{R}_m}$ and the corresponding signature σ'.
- $b \leftarrow \mathsf{Verify}(pk,\mathbf{m},\sigma)$. Given the public key pk, a message vector $\mathbf{m} \in (\mathbb{G}_i^*)^l$, and a signature σ as input, the verification algorithm outputs $b = 1$ for a valid signature and $b = 0$ otherwise.
- $b \leftarrow \mathsf{VKey}(sk,pk)$. Given a secret key sk and a public key pk, the key verification algorithm outputs $b = 1$ for a valid key pair and $b = 0$ otherwise.

Definition 4 (Correctness)**.** An SPS-EQ scheme on $\mathbb{G}_i$ is correct if for any bilinear group BG, any $l > 1$, any $(pk,sk) \leftarrow \mathsf{KeyGen}(\mathsf{BG},1^l)$, any $\mathbf{m} \in (\mathbb{G}_i^*)^l$, and any $\mu \in \mathbb{Z}_p^*$, we have:

$$\mathsf{VKey}(sk,pk) = 1 \wedge$$

$$\Pr[\mathsf{Verify}(pk,\mathbf{m},\mathsf{Sign}(sk,\mathbf{m})) = 1] = 1 \wedge$$

$$\Pr[\mathsf{Verify}(pk,\mathsf{ChgRep}(pk,\mathbf{m},\mathsf{Sign}(sk,\mathbf{m}),\mu)) = 1] = 1.$$

Definition 5 (EUF-CMA)**.** An SPS-EQ scheme on $\mathbb{G}_i$ is existentially unforgeable under adaptive chosen-message attacks if for any PPT adversary $\mathcal{A}$, any bilinear group BG, and any $l > 1$, there is a negligible function negl such that

$$\mathsf{Adv}^{euf}_{\mathcal{A}}(\mathsf{BG},l) \stackrel{\text{def}}{=} \Pr[\mathsf{EUF\text{-}CMA}_{\mathcal{A}}(\mathsf{BG},l) = 1] \leq \mathsf{negl}(\lambda),$$

where the existential unforgeability game under chosen message attacks $\mathsf{EUF\text{-}CMA}_{\mathcal{A}}(\mathsf{BG},l)$ is defined in Figure 1.

$\underline{\mathsf{CH}_{\mathcal{A}}(\mathsf{BG})}$:

1: $\mathbf{m} \xleftarrow{\$} (\mathbb{G}_i^*)^l$
2: $b \xleftarrow{\$} \{0,1\}$
3: $\mathbf{m}^{(0)} \xleftarrow{\$} (\mathbb{G}_i^*)^l, \mathbf{m}^{(1)} \xleftarrow{\$} [\mathbf{m}]_{\mathsf{R}_m}$
4: $b' \leftarrow \mathcal{A}(\mathsf{BG},\mathbf{m},\mathbf{m}^{(b)})$
5: **return** $b' = b$

Figure 2: Class-Hiding Game for SPS-EQ

Definition 6 (Class-Hiding)**.** The message space $(\mathbb{G}_i^*)^l$ of an SPS-EQ scheme is class-hiding if for any PPT adversary $\mathcal{A}$ and any bilinear group BG, there is a negligible function negl such that

$$\mathsf{Adv}^{ch}_{\mathcal{A}}(\mathsf{BG}) \stackrel{\text{def}}{=} \Pr[\mathsf{CH}_{\mathcal{A}}(\mathsf{BG}) = 1] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where the class-hiding game $\mathsf{CH}_{\mathcal{A}}(\mathsf{BG})$ is defined in Figure 2.

Definition 7 (Perfect Adaption of Signatures under Malicious Keys [21])**.** An SPS-EQ scheme on $\mathbb{G}_i$ perfectly adapts signatures under malicious keys if for any $(pk,\mathbf{m},\sigma,\mu)$ such that

$$\mathbf{m} \in (\mathbb{G}_i^*)^l \wedge \mathsf{Verify}(pk,\mathbf{m},\sigma) = 1 \wedge \mu \in \mathbb{Z}_p^*,$$

we have that the output σ' of $\mathsf{ChgRep}(pk,\mathbf{m},\sigma,\mu)$ is a uniformly random element in the space of signatures, conditioned on $\mathsf{Verify}(pk,\mathbf{m}^\mu,\sigma') = 1$.

2.3 Mercurial Signatures

Mercurial signatures [17] extended SPS-EQ to support equivalence relations for message, public key, and secret key spaces. It can additionally perform joint randomization of public keys and the corresponding signatures.

Definition 8 (Mercurial Signature)**.** Based on SPS-EQ, a mercurial signature scheme for parameterized equivalence relations R_m, R_{pk}, R_{sk} consists of the following additional PPT algorithms.

- $sk' \leftarrow \mathsf{ConvertSK}(sk,\rho)$. Given a secret key sk and a scalar $\rho \in \mathbb{Z}_p^*$ as input, the secret key randomization algorithm outputs a new secret key $sk' \in [sk]_{\mathsf{R}_{sk}}$ such that $sk' = \rho \cdot sk$.
- $pk' \leftarrow \mathsf{ConvertPK}(pk,\rho)$. Given a public key pk and a scalar $\rho \in \mathbb{Z}_p^*$ as input, the public key randomization algorithm outputs a new public key $pk' \in [pk]_{\mathsf{R}_{pk}}$ such that $pk' = pk^\rho$.
- $\sigma' \leftarrow \mathsf{ConvertSig}(pk,\mathbf{m},\sigma,\rho)$. Given a public key pk, a message vector $\mathbf{m}$, the signature σ corresponding to pk and $\mathbf{m}$, and a scalar $\rho \in \mathbb{Z}_p^*$ as input, the joint randomization algorithm outputs an updated signature σ' that is valid for pk^ρ and $\mathbf{m}$.

A mercurial signature scheme is unforgeable and class-hiding in its message space and public key space. It is also original-hiding if the following property holds.

Definition 9 (Original-Hiding [17])**.** A mercurial signature scheme is original-hiding if for any $(pk, \mathbf{m}, \sigma, \mu, \rho)$ such that

$$\mathsf{Verify}(pk, \mathbf{m}, \sigma) = 1 \ \wedge \ \mu, \rho \in \mathbb{Z}_p^*,$$

then $\mathsf{ChgRep}(pk, \mathbf{m}, \sigma, \mu)$ outputs a uniformly random σ' which is valid for pk and a uniformly random $\mathbf{m}' \in [\mathbf{m}]_{\mathsf{R}_m}$, and $\mathsf{ConvertSig}(pk, \mathbf{m}, \sigma, \rho)$ outputs a uniformly random σ'' which is valid for a uniformly random $pk^{\rho} \leftarrow \mathsf{ConvertPK}(pk, \rho)$ and $\mathbf{m}$.

3 HinToken: Anonymous Tokens with Designated-Reader Metadata Bit

We begin with the definitions of an anonymous token scheme with a designated-reader metadata bit, named HinToken, that supports public verification. However, only the user-specified reader can obtain the metadata bit (i.e., a signal) embedded in the token. We consider HinToken and its two extensions, which support reader-hiding and issuer-hiding, respectively.

3.1 Syntax

Definition 10 (HinToken)**.** An anonymous token scheme with designated-reader metadata bit consists of the following PPT algorithms and protocol.

- $pp \leftarrow \mathsf{Setup}(1^{\lambda})$. Given a security parameter λ in unary form as input, the setup algorithm outputs a public parameter pp.
- $(pk_I, sk_I) \leftarrow \mathsf{IKeyGen}(pp)$. Given the public parameter pp as input, the issuer's key generation algorithm outputs a public-secret key pair (pk_I, sk_I).
- $(pk_R, sk_R) \leftarrow \mathsf{RKeyGen}(pp)$. Given the public parameter pp as input, the reader's key generation algorithm outputs a public-secret key pair (pk_R, sk_R).
- $t/\perp \leftarrow \mathsf{Issue}\langle \mathsf{U}(pk_I, pk_R), \mathsf{I}(sk_I, pk_R, s) \rangle$. Given the issuer's public key pk_I and the reader's public key pk_R as the input of the user U, and the issuer's secret key sk_I, the same pk_R, and a metadata bit $s \in \{0,1\}$ as the input of the issuer I, the token issuance protocol outputs a token t or a failure ($\perp$) for U and nothing for I.
- $b \leftarrow \mathsf{Verify}(pk_I, t)$. Given the issuer's public key pk_I and a token t as input, the token verification algorithm outputs $b = 1$ for a valid token and $b = 0$ otherwise.
- $d \leftarrow \mathsf{Read}(sk_R, pk_I, t)$. Given the reader's secret key sk_R, the issuer's public key pk_I, and a token t as input, the metadata bit reading algorithm outputs $d = 1$ for a positive signal, $d = 0$ for a negative signal, and $d = \perp$ for a failure.

We say that HinToken has a one-round token issuance protocol initiated by the user, if the protocol can be split into the following three algorithms:

- $(pmsg, st) \leftarrow \mathsf{User}_0(pk_I, pk_R)$;
- $psig \leftarrow \mathsf{Sign}(sk_I, pk_R, s, pmsg)$;
- $t/\perp \leftarrow \mathsf{User}_1(psig, st)$.

This split also simplifies the security definitions in the next subsection. In practice, the user needs to send the identity of the specified reader at the beginning of each execution of the token issuance protocol (e.g., through *pmsg*). The issuer then checks the legitimacy of the reader and fetches the public key pk_R to run Sign.

Definition 11 (Correctness)**.** HinToken is correct if for any security parameter λ and any $s \in \{0,1\}$, there is a negligible function negl such that

$$\Pr\left[\begin{array}{c|c} \mathsf{Verify}(pk_I, t) = 1 \ \wedge \\ \mathsf{Read}(sk_R, pk_I, t) = s \end{array} \middle| \begin{array}{c} pp \leftarrow \mathsf{Setup}(1^{\lambda}), \\ (pk_I, sk_I) \leftarrow \mathsf{IKeyGen}(pp), \\ (pk_R, sk_R) \leftarrow \mathsf{RKeyGen}(pp), \\ t \leftarrow \mathsf{Issue}\langle \begin{array}{c} \mathsf{U}(pk_I, pk_R), \\ \mathsf{I}(sk_I, pk_R, s) \end{array} \rangle \end{array}\right]$$

is no less than $1 - \mathsf{negl}(\lambda)$.

We now define an anonymous token scheme with designated-reader metadata bit that supports reader-hiding, meaning the reader's identity is not visible to the issuer during the token issuance protocol and to other verifiers during the verification algorithm. In HinToken, the issuer takes the reader's public key pk_R as input, thereby ensuring that the public key is legitimate. In the reader-hiding scheme, we require that there is an acceptable public key set of readers for legitimacy.

Definition 12 (Reader-Hiding HinToken)**.** A reader-hiding anonymous token scheme with designated-reader metadata bit is based on HinToken by the following variant method.

- $t/\perp \leftarrow \mathsf{Issue}\langle \mathsf{U}(pk_I, \mathcal{R}, pk_R), \mathsf{I}(sk_I, \mathcal{R}, s) \rangle$. Given the issuer's public key pk_I, a (possibly authenticated) public key set $\mathcal{R}$ of readers acceptable to the issuer, and a reader's public key pk_R as the input of the user U, and the issuer's secret key sk_I, the same $\mathcal{R}$, and a metadata bit $s \in \{0,1\}$ as the input of the issuer I, the token issuance protocol outputs a token t or a failure ($\perp$) for U and nothing for I.

We also define an anonymous token scheme with designated-reader metadata bit that supports issuer-hiding,

that is, the issuer's identity is not visible to verifiers (including the reader) during the token verification and metadata bit reading algorithms. Similar to the reader-hiding scheme, we require an acceptable public key set of issuers. Moreover, for clarity, we explicitly output the auxiliary information used to verify that a token is produced by a legitimate issuer rather than including it as part of the token.

Definition 13 (Issuer-Hiding HinToken)**.** An issuer-hiding anonymous token scheme with designated-reader metadata bit is based on HinToken by the following variant methods.

- $(t, aux)/\perp \leftarrow \mathsf{Issue}\langle \mathsf{U}(I, pk_I, pk_R), \mathsf{I}(sk_I, pk_R, s)\rangle$. Given a (possibly authenticated) public key set I of issuers acceptable to the reader, the issuer's public key pk_I, and the reader's public key pk_R as the input of the user U, and the issuer's secret key sk_I, the same pk_R, and a metadata bit $s \in \{0,1\}$ as the input of the issuer I, the token issuance protocol outputs a token t and the auxiliary information aux or a failure ($\perp$) for U and nothing for I.
- $b \leftarrow \mathsf{Verify}(I, (t, aux))$. Given the public key set I, a token t, and the auxiliary information aux as input, the token verification algorithm outputs $b = 1$ for a valid token or $b = 0$ for an invalid token.
- $d \leftarrow \mathsf{Read}(sk_R, I, (t, aux))$. Given the reader's secret key sk_R, the public key set I, a token t, and the auxiliary information aux as input, the metadata bit reading algorithm outputs $d = 1$ for a positive signal, $d = 0$ for a negative signal, and $d = \perp$ for a failure.

The correctness of reader-hiding and issuer-hiding schemes can be derived from Definition 11.

We note that in real-world systems, such as public key infrastructures [4, 10], acceptable readers and issuers can be authenticated by certificate chains pointing to a root authority. In this case, the input in the relevant algorithms of reader-hiding and issuer-hiding schemes only requires the public key of the root authority rather than the public key set of readers or issuers. However, the reader (or issuer) will lose full control over which issuers (or readers) are acceptable. We consider this case in Section 8.

3.2 Security Definitions

We demand that HinToken must satisfy unforgeability, unlinkability, and privacy of the metadata bit [29]. For reader-hiding and issuer-hiding schemes, we additionally define reader anonymity and issuer anonymity, respectively, and we omit the security definitions of the first three properties for simplicity.

We first define one-more unforgeability. This property guarantees that for any signal, an adversary cannot obtain more valid tokens than requested. The adversary is given access to

$\underline{\mathsf{OMUF}_{\mathcal{A},n}(\lambda)}$:
1: $pp \leftarrow \mathsf{Setup}(1^\lambda)$
2: $(pk_I, sk_I) \leftarrow \mathsf{IKeyGen}(pp)$
3: $(pk_R, sk_R) \leftarrow \mathsf{RKeyGen}(pp)$
4: $q_0 \leftarrow 0, q_1 \leftarrow 0$
5: $\{t_i\}_{i\in[n+1]} \leftarrow \mathcal{A}^{O_{\mathsf{Sign}}, O_{\mathsf{Read}}}(pp, pk_I, pk_R)$
6: **return** $\forall s \in \{0,1\}, q_s \le n \ \wedge$
$\forall i, j \in [n+1], t_i \ne t_j \ \wedge$
$\forall i \in [n+1], \mathsf{Verify}(pk_I, t_i) = 1 \ \wedge$
$\exists s \in \{0,1\}, \forall i \in [n+1], \mathsf{Read}(sk_R, pk_I, t_i) = s$

$\underline{O_{\mathsf{Sign}}(pk_{R'}, s, pmsg)}$:
1: **if** $s \notin \{0,1\}$ **then**
2: **return** $\perp$
3: $q_s \leftarrow q_s + 1$
4: **return** $\mathsf{Sign}(sk_I, pk_{R'}, s, pmsg)$

$\underline{O_{\mathsf{Read}}(t)}$:
1: **return** $\mathsf{Read}(sk_R, pk_I, t)$

Figure 3: One-More Unforgeability Game

a signing oracle O_{Sign}. For each metadata bit s (0 or 1), the adversary is allowed to request tokens at most n times, but should not generate $n+1$ valid tokens with the same bit. We also consider the collusion between users and readers and allow the adversary to specify not only s but also pk'_R as the input of O_{Sign}. Even in this case, the adversary cannot output $n+1$ valid tokens for the reader specified by the challenger.

As the token t in our syntax encapsulates both the message and the corresponding signature, we consider a variant of unforgeability distinct from that in [29], which restricts the adversary from producing tokens tied to the same message. In contrast, our notion merely requires that the adversary cannot generate a new signature for the same message. This difference is analogous to the distinction between existential unforgeability [23] and strong existential unforgeability [1] in digital signature schemes.

Definition 14 (One-More Unforgeability)**.** HinToken is one-more unforgeable if for any PPT adversary $\mathcal{A}$ and any $n \ge 0$, there is a negligible function negl such that

$$\mathsf{Adv}^{omuf}_{\mathcal{A},n}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathsf{OMUF}_{\mathcal{A},n}(\lambda) = 1] \le \mathsf{negl}(\lambda),$$

where the one-more unforgeability game $\mathsf{OMUF}_{\mathcal{A},n}(\lambda)$ is defined in Figure 3.

Unlinkability guarantees that even if the issuer and the reader collude, a token shown by a user cannot be linked to any execution of the token issuance protocol. The adversary is given access to O_{User_0} to open an execution of the protocol. We also provide the adversary with access to O_{User_1}, allowing it to link a part of the tokens. Even so, it cannot break the unlinkability of the remaining tokens (i.e., the challenge tokens).

$\underline{\mathsf{UNLINK}_{\mathcal{A},m}(\lambda)}$:

1: $pp \leftarrow \mathsf{Setup}(1^\lambda)$
2: $(pk_I, pk_R, st_{\mathcal{A}_0}) \leftarrow \mathcal{A}_0(pp)$
3: $Q \leftarrow \emptyset, q_0 \leftarrow 0, q_1 \leftarrow 0$
4: $(\{psig_i\}_{(i,*)\in Q}, st_{\mathcal{A}_1}) \leftarrow \mathcal{A}_1^{O_{\mathsf{User}_0}, O_{\mathsf{User}_1}}(st_{\mathcal{A}_0})$
5: **if** $Q = \emptyset$ **then**
6: **return** 0
7: **for** $(i, st_i) \in Q$ **do**
8: $t_i \leftarrow \mathsf{User}_1(psig_i, st_i)$
9: $j \xleftarrow{\$} Q, Q \leftarrow Q \backslash \{(j, st_j)\}$
10: Choose a random permutation ϕ of Q
11: $j' \leftarrow \mathcal{A}_2(t_j, \{t_{\phi(i)}\}_{(i,*)\in Q}, st_{\mathcal{A}_1})$
12: **return** $q_0 - q_1 \geq m \wedge j' = j$

$\underline{O_{\mathsf{User}_0}(pk_I, pk_R)}$:

1: $q_0 \leftarrow q_0 + 1$
2: $(pmsg_{q_0}, st_{q_0}) \leftarrow \mathsf{User}_0(pk_I, pk_R)$
3: $Q \leftarrow Q \cup \{(q_0, st_{q_0})\}$
4: **return** $pmsg_{q_0}$

$\underline{O_{\mathsf{User}_1}(psig_i)}$:

1: **if** $(i,*) \notin Q$ **then**
2: **return** $\perp$
3: $t_i \leftarrow \mathsf{User}_1(psig_i, st_i)$
4: **if** $t_i \neq \perp$ and $\mathsf{Verify}(pk_I, t_i)$ **then**
5: $Q \leftarrow Q \backslash \{(i, st_i)\}$
6: $q_1 \leftarrow q_1 + 1$
7: **return** t_i

Figure 4: Unlinkability Game

Kreuter et al. [29] consider the general case of κ-unlinkability that for m challenge tokens, the probability that the adversary can link the tokens is no greater than $\frac{\kappa}{m}$. Since the metadata embedded in tokens is a single bit determined by the adversary, the executions of the token issuance protocol will be partitioned into two subsets. Then, the adversary can read the bit from each challenge token and associate it with one subset of the executions. This means that the probability of outputting the correct answer is at least twice that of random guessing. Therefore, we focus on 2-unlinkable security.

Definition 15 (Unlinkability)**.** HinToken is unlinkable if for any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ and any $m > 0$, there is a negligible function negl such that

$$\mathsf{Adv}^{unlink}_{\mathcal{A},m}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathsf{UNLINK}_{\mathcal{A},m}(\lambda) = 1] \leq \frac{2}{m} + \mathsf{negl}(\lambda),$$

where the unlinkability game $\mathsf{UNLINK}_{\mathcal{A},m}(\lambda)$ is defined in Figure 4.

Privacy of metadata bit guarantees that any unspecified reader, including the user, cannot distinguish whether the

$\underline{\mathsf{PMB}_{\mathcal{A}}(\lambda)}$:

1: $pp \leftarrow \mathsf{Setup}(1^\lambda)$
2: $(pk_I, sk_I) \leftarrow \mathsf{IKeyGen}(pp)$
3: $(pk_R, sk_R) \leftarrow \mathsf{RKeyGen}(pp)$
4: $flag \leftarrow 0$
5: $b \xleftarrow{\$} \{0,1\}$
6: $b' \leftarrow \mathcal{A}^{O^b_{\mathsf{sign}}, O_{\mathsf{Sign}'}, O_{\mathsf{Read}}}(pk_I, pk_R)$
7: **return** $b' = b$

$\underline{O^b_{\mathsf{Sign}}(pmsg)}$:

1: $flag \leftarrow 1$
2: **return** $\mathsf{Sign}(sk_I, pk_R, b, pmsg)$

$\underline{O_{\mathsf{Sign}'}(s, pmsg)}$:

1: **if** $s \notin \{0,1\}$ **then**
2: **return** $\perp$
3: **return** $\mathsf{Sign}(sk_I, pk_R, s, pmsg)$

$\underline{O_{\mathsf{Read}}(t)}$:

1: **if** $flag$ **then**
2: **return** $\perp$
3: **return** $\mathsf{Read}(sk_R, pk_I, t)$

Figure 5: Private Metadata Bit Game

metadata bit embedded in a token is 0 or 1 [13]. For a metadata bit b chosen by the challenger, the adversary is given access to a signing oracle O^s_{Sign} to generate the challenge $psig$. Furthermore, we allow the adversary to obtain tokens with any bit by interacting with O_{Sign} and read the metadata bit from any token of its choice by interacting with O_{Read}. Note that, similar to the definition of CCA1 [5], once the challenge is generated, the adversary no longer has access to O_{Read}.

Definition 16 (Privacy of Metadata Bit)**.** HinToken provides private metadata bit if for any PPT adversary $\mathcal{A}$, there is a negligible function negl such that

$$\mathsf{Adv}^{pmb}_{\mathcal{A}}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathsf{PMB}_{\mathcal{A}}(\lambda) = 1] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where the private metadata bit game $\mathsf{PMB}_{\mathcal{A}}(\lambda)$ is defined in Figure 5.

In addition to HinToken's security properties, the reader-hiding HinToken scheme must also meet reader anonymity. This property guarantees that the identity (i.e., the public key) of the user-specified reader is hidden in the token issuance protocol. We assume that the issuer does not collude with all readers (i.e., it can collude with a subset of readers), otherwise the issuer may exhaustively search for the user-specified reader using readers' secret keys (i.e., sk_0 and sk_1 in Figure 6). In other words, reader anonymity guarantees that the adversary cannot determine which of two honest readers the user specified, even if all other readers and the issuer collude. This

$\mathsf{RA}_{\mathcal{A}}(\lambda)$:

1: $pp \leftarrow \mathsf{Setup}(1^\lambda)$
2: $(pk_0, sk_0), (pk_1, sk_1) \leftarrow \mathsf{RKeyGen}(pp)$
3: $\mathcal{R} \leftarrow (pk_0, pk_1)$
4: $(pk_I, st_{\mathcal{A}_0}) \leftarrow \mathcal{A}_0(pp, \mathcal{R})$
5: $b \xleftarrow{\$} \{0,1\}$
6: $(pmsg, st) \leftarrow \mathsf{User}_0(pk_I, \mathcal{R}, pk_b)$
7: $b' \leftarrow \mathcal{A}_1(pmsg, st_{\mathcal{A}_0})$
8: **return** $b' = b$

Figure 6: Reader Anonymity Game

assumption is reasonable, since such collusion involves too many participants and is therefore improbable to occur in practice. We discuss how to resist full collusion (i.e., pk_0 and pk_1 in Figure 6 are output by the adversary) in Section 8.2.

Definition 17 (Reader Anonymity)**.** A reader-hiding HinToken scheme provides reader anonymity if for any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there is a negligible function negl such that

$$\mathsf{Adv}^{ra}_{\mathcal{A}}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathsf{RA}_{\mathcal{A}}(\lambda) = 1] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where the reader anonymity game $\mathsf{RA}_{\mathcal{A}}(\lambda)$ is defined in Figure 6.

The issuer-hiding HinToken scheme must meet issuer anonymity. This property guarantees that the identity (i.e., the public key) of the issuer of the token is hidden in the token verification algorithm. The adversary is given access to a signing oracle O_{Sign} to obtain tokens with any bit issued by any issuer it chooses. Similar to reader anonymity, for the sake of reasonability and simplicity, we assume that the verifier does not collude with all issuers (i.e., it can collude with a subset of issuers), and discuss how to resist such collusion in Section 8.2.

Definition 18 (Issuer Anonymity)**.** An issuer-hiding HinToken scheme provides issuer anonymity if for any PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there is a negligible function negl such that

$$\mathsf{Adv}^{ia}_{\mathcal{A}}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathsf{IA}_{\mathcal{A}}(\lambda) = 1] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where the issuer anonymity game $\mathsf{IA}_{\mathcal{A}}(\lambda)$ is defined in Figure 7.

4 Basic Construction

We present a construction of anonymous tokens with metadata bit for a user-specified (and issuer-accepted) reader,

$\mathsf{IA}_{\mathcal{A}}(\lambda)$:

1: $pp \leftarrow \mathsf{Setup}(1^\lambda)$
2: $(pk_0, sk_0), (pk_1, sk_1) \leftarrow \mathsf{IKeyGen}(pp)$
3: $I \leftarrow (pk_0, pk_1)$
4: $(pk_R, st_{\mathcal{A}_0}) \leftarrow \mathcal{A}_0(pp, I)$
5: $b \xleftarrow{\$} \{0,1\}, s \xleftarrow{\$} \{0,1\}$
6: $(t, aux) \leftarrow \mathsf{Issue}\langle \mathsf{U}(I, pk_b, pk_R), \mathsf{I}(sk_b, pk_R, s)\rangle$
7: $b' \leftarrow \mathcal{A}_1^{O_{\mathsf{Sign}}}((t, aux), st_{\mathcal{A}_0})$
8: **return** $b' = b$

$O_{\mathsf{Sign}}(pk_i, s')$:

1: **if** $i \notin \{0,1\}$ or $s' \notin \{0,1\}$ **then**
2: **return** $\perp$
3: **return** $\mathsf{Issue}\langle \mathsf{U}(I, pk_i, pk_R), \mathsf{I}(sk_i, pk_R, s')\rangle$

Figure 7: Issuer Anonymity Game

called HinToken. Then, we extend it to support reader-hiding, named HinToken-R, in Section 5, and to support issuer-hiding, named HinToken-I, in Section 6.

To embed a metadata bit in the token that can only be read by the user-specified reader, a straightforward approach is to encrypt this bit using the reader's public key. For the privacy of metadata bit, the encryption should be done by the issuer. Then, there are two potential violations of unlinkability. First, a malicious issuer may exploit the security property (e.g., semantic security [8]) of the encryption scheme to encrypt messages other than 0 or 1, thereby identifying users. Therefore, the issuer must prove the legitimacy of its plaintext in zero-knowledge. Second, this ciphertext can be used to mark the execution of the token issuance protocol. Thus, the ciphertext should be rerandomizable [35], i.e., the user can convert the ciphertext into another ciphertext that is indistinguishable from a fresh encryption of the same plaintext.

The issuer needs to sign the ciphertext to constitute a token that must also be randomized; otherwise, unlinkability will be compromised. To this end, we adopt SPS-EQ [21, 25], which enables the message and corresponding signature to be randomized together, ensuring the indistinguishability of the old and new signatures. To prevent users from obtaining multiple tokens from a single execution of the issuance, we fix the first element of the message vector, such that each equivalence class has a unique representative in a valid token.

Note that blind signature [15, 34] is a common way to construct anonymous token schemes, which allows users to blind a message and obtain the signature from the signer. However, since the ciphertext is added to the message by the issuer, blind signatures are no longer applicable. This situation is similar to NIBS [26], where the message is chosen by the signer and randomized by the user. However, in HinToken, users need to specify the reader, so it is inherently interactive. Using NIBS requires additional random oracles (beyond the

Fiat-Shamir transformation [20]) and users' public-private key pairs, which complicates our construction. We can render HinToken non-interactive if the reader is fixed.

4.1 Embedding Metadata Bit

An SPS-EQ scheme allows users to randomize a signature σ on a message vector $\mathbf{m} \in (\mathbb{G}_i^*)^l$ ($i \in \{1,2\}$) by a scalar $\mu \in \mathbb{Z}_p^*$ and obtain a new signature σ' on $\mathbf{m}' = \mathbf{m}^\mu$. Since the ciphertext is a part of the message vector, we require encryption to produce ciphertext on $\mathbb{G}_i^*$. Moreover, it can be randomized by a scalar and keep the plaintext unchanged under such randomization. To this end, we design an encryption method on $\mathbb{G}_1$ that is similar to the El Gamal encryption [19].

For a metadata bit $s \in \{0,1\}$, a generator $g_1 \in \mathbb{G}_1$, and a reader's public key $y = g_1^x$, the ciphertext is the triple $(h, g_1^r, y^r h^s)$, where $h \xleftarrow{\$} \mathbb{G}_1$ and $r \xleftarrow{\$} \mathbb{Z}_p$. To ensure unlinkability, h must be selected by the user. To decrypt a ciphertext (t_2, t_3, t_4), the user-specified reader computes $h' \leftarrow t_3^{-x} t_4$ using its secret key x. Then, the plaintext is 0 if $h' = 1_{\mathbb{G}_1}$ and 1 if $h' = t_2$. The correctness of this encryption method is straightforward. Moreover, the ciphertext can be randomized to $(h^z, g_1^{rz}, y^{rz} h^{sz})$ by a scalar z, which is a ciphertext of the same plaintext s.

To ensure the correctness of the ciphertext, the issuer needs to prove that the plaintext it encrypted is either 0 or 1. To this end, we adopt an NIZK system. Let pp_g be the description of $\mathbb{G}_1$. The issuer needs to provide a valid proof for the following language $\mathcal{L}_{pp_g}$:

$$\mathcal{L}_{pp_g} = \left\{ (g_1, y, h, u_s, v_s) \in \mathbb{G}_1^5 \;\middle|\; \begin{array}{l} \exists (r,s) \in \mathbb{Z}_p \times \{0,1\}, \\ \text{s.t. } u_s = g_1^r, v_s = y^r h^s \end{array} \right\}.$$

Although many NIZK systems can be used in HinToken, we present a simple and efficient instantiation $\Pi_v = (\mathsf{Prove}, \mathsf{Vrfy})$ for $\mathcal{L}_{pp_g}$ in Figure 8, where $\mathsf{H_g} : \{0,1\}^* \to \mathbb{Z}_p$ is a hash function. It is similar to the OR proof of Σ-protocols [8]. Completeness is straightforward.

Our instantiation provides special soundness. For two accepted transcripts $(\gamma, c_s, \alpha_s, c_{1-s}, \alpha_{1-s})$ and $(\gamma, c_s', \alpha_s', c_{1-s}', \alpha_{1-s}')$ such that either $c_s \neq c_s'$ or $c_{1-s} \neq c_{1-s}'$, we can extract the witness (r,s). Specifically, $s = 0$ if $c_s \neq c_s'$ and $s = 1$ otherwise. Then, r can be extracted by computing $r = ((\alpha_0 - \alpha_0')(c_0 - c_0')^{-1} - \gamma s) \cdot (1-2s)^{-1}$.

Our instantiation is special honest verifier zero-knowledge. The simulator samples $\gamma, \alpha_0, c_0, \alpha_1, c_1 \xleftarrow{\$} \mathbb{Z}_p$. Then, it computes $u_0^* \leftarrow g_1^{\alpha_0} u_0^{-c_0}$, $v_0^* \leftarrow y^{\alpha_0} v_0^{-c_0}$, $u_1^* \leftarrow g_1^{\alpha_1} u_1^{-c_1}$, and $v_1^* \leftarrow y^{\alpha_1} v_1^{-c_1}$. Finally, it programs the random oracle to reply with $c_0 + c_1$ when queried on $(g_1, y, h, u_s, v_s, \gamma, u_s^*, v_s^*, u_{1-s}^*, v_{1-s}^*)$. In this way, the simulator can generate a transcript whose distribution is indistinguishable from a real one.

$\mathsf{Prove}(pp_g, (g_1, y, h, u_s, v_s), (r,s))$

1: $\gamma \xleftarrow{\$} \mathbb{Z}_p$
2: $u_{1-s} \leftarrow g_1^\gamma u_s^{-1}, v_{1-s} \leftarrow y^\gamma h v_s^{-1}$
3: $\alpha_1, c_1 \xleftarrow{\$} \mathbb{Z}_p$
4: $u_1^* \leftarrow g_1^{\alpha_1} u_1^{-c_1}, v_1^* \leftarrow y^{\alpha_1} v_1^{-c_1}$
5: $r_0 \xleftarrow{\$} \mathbb{Z}_p$
6: $u_0^* \leftarrow g_1^{r_0}, v_0^* \leftarrow y^{r_0}$
7: $c \leftarrow \mathsf{H_g}(g_1, y, h, u_s, v_s, \gamma, u_s^*, v_s^*, u_{1-s}^*, v_{1-s}^*)$
8: $c_0 \leftarrow c - c_1, \alpha_0 \leftarrow r_0 + c_0(\gamma s + (1-2s)r)$
9: $\pi_v \leftarrow (\gamma, c_s, \alpha_s, c_{1-s}, \alpha_{1-s})$
10: **return** π_v

$\mathsf{Vrfy}(pp_g, (g_1, y, h, u_s, v_s), \pi_v)$

1: $\pi_v = (\gamma, c_s, \alpha_s, c_{1-s}, \alpha_{1-s})$
2: $u_{1-s} \leftarrow g_1^\gamma u_s^{-1}, v_{1-s} \leftarrow y^\gamma h v_s^{-1}$
3: $u_s^* \leftarrow g_1^{\alpha_s} u_s^{-c_s}, v_s^* \leftarrow y^{\alpha_s} v_s^{-c_s}$
4: $u_{1-s}^* \leftarrow g_1^{\alpha_{1-s}} u_{1-s}^{-c_{1-s}}, v_{1-s}^* \leftarrow y^{\alpha_{1-s}} v_{1-s}^{-c_{1-s}}$
5: $c \leftarrow \mathsf{H_g}(g_1, y, h, u_s, v_s, \gamma, u_s^*, v_s^*, u_{1-s}^*, v_{1-s}^*)$
6: **if** $c = c_s + c_{1-s}$ **then**
7: **return** 1
8: **return** 0

Figure 8: Proof of Plaintext Legitimacy

4.2 Construction Details

Let $\mathsf{EQ} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{ChgRep}, \mathsf{Verify}, \mathsf{VKey})$ be an SPS-EQ scheme on $\mathbb{G}_1$. The details of HinToken are as follows.

- Setup. Let $\mathsf{BG} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ be a bilinear group. The public parameter is $pp \leftarrow \mathsf{BG}$.
- IKeyGen. Set $l = 4$ and output the issuer's key pair $(pk_I, sk_I) \leftarrow \mathsf{EQ.KeyGen}(\mathsf{BG}, 1^l)$.
- RKeyGen. Sample $x \xleftarrow{\$} \mathbb{Z}_p$ and compute $y \leftarrow g_1^x \in \mathbb{G}_1$. Output the reader's key pair $(pk_R, sk_R) = (y, x)$. Let pp_g be the description of $\mathbb{G}_1$. We assume that it is implicted in the key pair.
- Issue. Let $pk_R = y$. The details of the token issuance protocol are as follows.

U: Sample $h \xleftarrow{\$} \mathbb{G}_1$ and $z \xleftarrow{\$} \mathbb{Z}_p^*$, compute $g_1^* \leftarrow g_1^{z^{-1}}$, and set $st = (y, h, z, g_1^*)$.

U $\Rightarrow$ I: Send $pmsg = (g_1^*, h)$.

I: Sample $r \xleftarrow{\$} \mathbb{Z}_p$ and compute:
$u_s \leftarrow g_1^r, v_s \leftarrow y^r h^s$,
$\pi_v \leftarrow \Pi_v.\mathsf{Prove}(pp_g, (g_1, y, h, u_s, v_s), (r,s))$,
$\sigma^* \leftarrow \mathsf{EQ.Sign}(sk_I, (g_1^*, h, u_s, v_s))$.

I $\Rightarrow$ U: Send $psig = (u_s, v_s, \pi_v, \sigma^*)$.

U: If $\Pi_{\mathrm{v}}.\mathsf{Vrfy}(pp_g,(g_1,y,h,u_s,v_s),\pi_v) = 1$ and $\mathsf{EQ.Verify}(pk_I,(g_1^*,h,u_s,v_s),\sigma^*) = 1$, output: $t \leftarrow \mathsf{EQ.ChgRep}(pk_I,(g_1^*,h,u_s,v_s),\sigma^*,z)$; otherwise, output $\perp$.

- Verify. Parse t as $((t_1,t_2,t_3,t_4),\sigma)$. If $t_1 = g_1$ and $\mathsf{EQ.Verify}(pk_I,(t_1,t_2,t_3,t_4),\sigma) = 1$, output 1; otherwise, output 0.
- Read. Parse sk_R as x and t as $((t_1,t_2,t_3,t_4),\sigma)$. If $\mathsf{Verify}(pk_I,t) = 0$, output $\perp$. Compute $h^* \leftarrow t_3^{-x}t_4$. If $h^* = 1_{\mathbb{G}}$, output 0; if $h^* = t_2$, output 1; else, output $\perp$.

Note that the message vector in a valid token is in the form of $(g_1, h^z, g_1^{rz}, y^{rz}h^{sz})$. The correctness of HinToken is based on the correctness of our encryption method and the SPS-EQ scheme EQ.

4.3 Security

We prove the one-more unforgeability, unlinkability, and privacy of metadata bit of HinToken in Appendix A.

Theorem 1. If the SPS-EQ scheme EQ is existentially unforgeable under adaptive chosen-message attacks, then HinToken is one-more unforgeable.

Theorem 2. If the NIZK system Π_{v} provides special soundness, the SPS-EQ scheme EQ perfectly adapts signatures under malicious keys, and the inverse DDH assumption holds in $\mathbb{G}_1$, then HinToken is unlinkable.

Theorem 3. If the NIZK system Π_{v} provides special honest verifier zero-knowledge, the SPS-EQ scheme EQ is existentially unforgeable under adaptive chosen-message attacks, and the DDH assumption holds in $\mathbb{G}_1$, then HinToken provides private metadata bit.

5 Reader-Hiding Construction

We extend HinToken to support reader-hiding, named HinToken-R. An intuitive way to achieve reader-hiding is for the user to mask the reader's public key so that the issuer encrypts the metadata bit blindly. In fact, we take reader-hiding into consideration when designing the encryption method in Section 4. Recall that in our method, $(h, g_1^r, y^r h^s)$ and $(h, g_1^{r\gamma}, y^{r\gamma}h^s)$ are two ciphertexts of the same plaintext under the same key, where $\gamma \in \mathbb{Z}_p$ is a random number. The latter is also the ciphertext under g_1^γ and the public key y^γ. Since y^γ is uniformly distributed, we treat it as the blinded public key.

To prevent abuse, the user should prove that the public key it specified is within an acceptable range defined by the issuer. A trivial solution is to employ an NIZK system to prove that the user knows the witness (j,γ) for the statement $(y,\{y_1,\ldots,y_q\})$ such that $y = y_j^\gamma$. Although we can design a dedicated instantiation (e.g., a variant of the one we proposed in Section 8) of such an NIZK system, this approach usually leads to high computational and communication overheads.

Another solution is to randomize y_j in an authenticated manner, which is exactly what SPS-EQ can provide. Specifically, for a message vector $(g_1,y_j) \in (\mathbb{G}_1^*)^2$ and $(g^*,y^*) \xleftarrow{\$} (\mathbb{G}_1^*)^2$, $((g_1,y_j),(g_1^\gamma,y_j^\gamma))$ and $((g_1,y_j),(g^*,y^*))$ are indistinguishable (i.e., class-hiding) assuming that the issuer and the reader do not collude (as in Definition 17). Let σ_j be a signature of the message vector (g_1,y_j). Then, the user can obtain the signature σ_j^* on (g_1^γ,y_j^γ) using the randomization algorithm of SPS-EQ. In this way, we can ensure the legitimacy of the reader's public key without zero-knowledge proofs. Combining it with HinToken can obtain a construction of HinToken-R.

5.1 Generating Authenticated Set

Assume that the reader public keys acceptable to the issuer are $(y_1,\ldots,y_q) \in (\mathbb{G}_1^*)^q$. The details of generating an authenticated set are as follows.

- $\mathsf{IKeyGen}_2(\mathsf{BG})$. Set $l = 2$ and output the issuer's key pair $(pk_{I2}, sk_{I2}) \leftarrow \mathsf{EQ.KeyGen}(\mathsf{BG}, 1^l)$.
- $\mathsf{SetGen}(y_1,\ldots,y_q)$. For each acceptable key y_i where $i \in [q]$, compute $\sigma_i \leftarrow \mathsf{EQ.Sign}(sk_{I2},(g_1,y_i))$. Output $\mathcal{R} = \{(y_i,\sigma_i)\}_{i\in[q]}$.

$\mathcal{R}$ is generated using SetGen and published by the issuer in the initial stage of the system. The issuer can also update $\mathcal{R}$ using SetGen at any time. We omit these two algorithms in the syntax of HinToken-R for simplicity.

5.2 Construction Details

Combining HinToken and the authenticated set $\mathcal{R}$, we can obtain the construction of our reader-hiding HinToken scheme. In this construction, Setup, IKeyGen, RKeyGen, Verify, and Read are the same as the basic construction. The token issuance protocol is as follows. Note that the first step of this protocol can be done only once for $\mathcal{R}$ when the user receives it. Then, the user does not need to verify $((g_1,y_j),\sigma_j)$ in each execution of the protocol.

- Issue. Let $pk_R = y_j \in \{y_1,\ldots,y_n\}$ and σ_j be the corresponding signature in $\mathcal{R}$, i.e., $(y_j,\sigma_j) \in \mathcal{R}$.

  U: If $\mathsf{EQ.Verify}(pk_{I2},(g_1,y_j),\sigma_j) = 0$, output $\perp$.
  Sample $h \xleftarrow{\$} \mathbb{G}_1$ and $z,\gamma \xleftarrow{\$} \mathbb{Z}_p^*$, and compute: $(g^*,y^*,\sigma_j^*) \leftarrow \mathsf{EQ.ChgRep}(pk_{I2},(g_1,y_j),\sigma_j,\gamma)$.
  Set $g_1^* \leftarrow g_1^{z^{-1}}$ and $st = (g^*,y^*,h,z,g_1^*)$.

  U $\Rightarrow$ I: Send $pmsg = (g_1^*,h,g^*,y^*,\sigma_j^*)$.

I: If $\mathsf{EQ.Verify}(pk_{I2},(g^*,y^*),\sigma_j^*)=0$, abort.

Sample $r \xleftarrow{\$} \mathbb{Z}_p$ and compute:
$u_s \leftarrow (g^*)^r, v_s \leftarrow (y^*)^r h^s$,
$\pi_v \leftarrow \Pi_{\mathrm{v}}.\mathsf{Prove}(pp_g,(g^*,y^*,h,u_s,v_s),(r,s))$,
$\sigma^* \leftarrow \mathsf{EQ.Sign}(sk_I,(g_1^*,h,u_s,v_s))$.

I $\Rightarrow$ U: Send $psig=(u_s,v_s,\pi_v,\sigma^*)$.

U: If $\Pi_{\mathrm{v}}.\mathsf{Vrfy}(pp_g,(g^*,y^*,h,u_s,v_s),\pi_v)=1$ and $\mathsf{EQ.Verify}(pk_I,(g_1^*,h,u_s,v_s),\sigma^*)=1$, output: $t \leftarrow \mathsf{EQ.ChgRep}(pk_I,(g_1^*,h,u_s,v_s),\sigma^*,z)$; otherwise, output $\perp$.

We prove the security of HinToken-R in Appendix B.

6 Issuer-Hiding Construction

In this section, we extend HinToken to support issuer-hiding, named HinToken-I. We still adopt the idea of masking the issuer's public key when a user presents its tokens. To this end, we need to consider two issues. First, to avoid abuse, the legitimacy of the blinded issuer's public key must be proved. Similar to HinToken-R in Section 5, each verifier defines a set of acceptable issuer public keys (denoted as I) and requires the user to prove that the token it presented is issued by a member of this set. Second, the token must be randomized along with the blinded key and can only be verified by this key; otherwise, a verifier can exhaustively search for the issuer using the public keys in I.

In HinToken, the issuer's key pair is generated using the key generation algorithm of SPS-EQ. As observed in mercurial signatures [17, 24], SPS-EQ supports joint randomization of public keys and their corresponding signatures while maintaining verifiability. Leveraging this feature, we allow users to randomize the issuer's public key and the signature in a token before presenting it. This solves the second issue.

To solve the first issue, we note that the solution in Section 5 also applies to HinToken-I. Specifically, for each acceptable issuer's public key pk_i, the reader signs and releases an SPS-EQ signature σ_i on it. When a user randomizes pk_j (to solve the second issue), it uses the same scalar to randomize σ_j, and the legitimacy of the blinded key is guaranteed by the unforgeability of SPS-EQ. Note that this solution downgrades our construction to one-more unforgeability defined in [29] rather than Definition 14. To resile strong security, we can use the technique in Section 4 to ensure that there is only one legitimate randomization of σ_j. However, this will significantly increase the overhead of our construction, so we opt to satisfy traditional one-more unforgeability.

The shortcomings of this construction are twofold. On the one hand, the reader needs to generate an additional key pair of SPS-EQ. Moreover, the message vector consists of elements of $\mathbb{G}_2^*$ rather than $\mathbb{G}_1^*$ as in Section 4 and 5, which means we need another instantiation of SPS-EQ. On the other hand, this construction cannot be extended to support reader-hiding. This is because the randomization of σ_i can be verified using the reader's public key. Then, anyone (e.g., other verifiers) observing the token can learn about the specified reader. We solve these shortcomings in Section 8.

6.1 Generating Authenticated Set

Assume that the issuer public keys acceptable to the reader are $(pk_1,\ldots,pk_q) \in (\mathbb{G}_2^*)^{4q}$. Let EQ_2 be an SPS-EQ scheme on $\mathbb{G}_2$. The details of generating an authenticated set are as follows.

- $\mathsf{RKeyGen}_2(\mathsf{BG})$. Set $l=4$ and output the reader's key pair $(pk_{R2},sk_{R2}) \leftarrow \mathsf{EQ}_2.\mathsf{KeyGen}(\mathsf{BG},1^l)$.
- $\mathsf{SetGen}(pk_1,\ldots,pk_q)$. For each acceptable key pk_i where $i \in [q]$, compute $\sigma_i \leftarrow \mathsf{EQ}_2.\mathsf{Sign}(sk_{R2},pk_i)$. Output $I=\{(pk_i,\sigma_i)\}_{i\in[q]}$. We assume that pk_{R2} is implicted in I.

I is also generated using SetGen and published by the reader in the initial stage of the system. We omit these two algorithms in the syntax of HinToken-I for simplicity.

6.2 Construction Details

Combining HinToken and the authenticated set I, we can obtain a construction of our issuer-hiding HinToken scheme. Let EQ be an extended SPS-EQ scheme with additional support for ConvertPK and ConvertSig, which is actually an instantiation of mercurial signature in [17]. Let EQ_2 be an SPS-EQ scheme on $\mathbb{G}_2$. Setup, IKeyGen, and RKeyGen are the same as the basic construction. The token issuance protocol is updated as follows. Similar to HinToken-R, the following first step can be done only once for I when the user receives it. Then, the user does not need to verify (pk_j,σ_j) in each execution of the protocol.

- Issue. Let $pk_I=pk_j \in (pk_1,\ldots,pk_q)$ and σ_j be the corresponding signature in I. After U obtains a token $t=((t_1,t_2,t_3,t_4),\sigma)$ from I through Issue of HinToken, it performs the following operations.

  U: If $\mathsf{EQ}_2.\mathsf{Verify}(pk_{R2},pk_j,\sigma_j)=0$, output $\perp$.
  Sample $\rho \xleftarrow{\$} \mathbb{Z}_p^*$ and compute:
  $\sigma^* \leftarrow \mathsf{EQ.ConvertSig}(pk_j,(t_1,t_2,t_3,t_4),\sigma,\rho)$,
  $(pk_j^*,\sigma_j^*) \leftarrow \mathsf{EQ}_2.\mathsf{ChgRep}(pk_{R2},pk_j,\sigma_j,\rho)$.
  Output $t \leftarrow ((t_1,t_2,t_3,t_4),\sigma^*)$ and $aux \leftarrow (pk_j^*,\sigma_j^*)$.

The token verification and metadata bit reading algorithms are updated as follows:

- Verify. Let $t=((t_1,t_2,t_3,t_4),\sigma^*)$ and $aux=(pk_j^*,\sigma_j^*)$. If $t_1=g_1$, $\mathsf{EQ}_2.\mathsf{Verify}(pk_{R2},pk_j^*,\sigma_j^*)=1$, and $\mathsf{EQ.Verify}(pk_j^*,(t_1,t_2,t_3,t_4),\sigma^*)=1$, output 1; otherwise, output 0.

Table 1: Complexity comparisons of HinToken and related schemes, where the group exponentiation and pairing operation are denoted by E and P, respectively, and the group element and scalar are denoted by $\mathbb{G}$ and $\mathbb{Z}_p$, respectively.

Scheme	IKeyGen	RKeyGen	Issue	Verify and Read	Comm. (in Issue)	Token Size
PMBT [29]	4E	–	27E	6E	$2\mathbb{G}$+$7\mathbb{Z}_p$	$2\mathbb{G}$+$1\mathbb{Z}_p$
ATHM [13]	5E	–	36E	2E	$4\mathbb{G}$+$8\mathbb{Z}_p$	$2\mathbb{G}$+$1\mathbb{Z}_p$
PVAT [6]	2E	–	28E	10E	$5\mathbb{G}$+$6\mathbb{Z}_p$	$3\mathbb{G}$+$5\mathbb{Z}_p$
NIAT [2]	4E	–	30E+5P	1E+5P	$4\mathbb{G}$+$7\mathbb{Z}_p$	$5\mathbb{G}$
HinToken	4E	1E	30E+7P	1E+7P	$7\mathbb{G}$+$5\mathbb{Z}_p$	$7\mathbb{G}$

- Read. Let $t = ((t_1,t_2,t_3,t_4),\sigma^*)$ and $sk_R = x$. If $\mathsf{Verify}(I,t,aux) = 0$, output $\perp$. Compute $h^* \leftarrow t_3^{-x} t_4$. If $h^* = 1_{\mathbb{G}}$, output 0; if $h^* = t_2$, output 1; else, output $\perp$.

We prove the security of HinToken-I in Appendix C.

7 Evaluation

Theoretical Complexity. We evaluate the computational costs in terms of the number of group exponentiations and pairing operations, and the communication costs in terms of the number of elements sent in Issue and in the token. The complexity comparison of HinToken and related schemes is shown in Table 1. Like other schemes, our construction has *constant* computational and communication costs. Note that HinToken is the only construction that supports user-specified readers and can be extended to support issuer-hiding.

The computational costs of HinToken are dominated by the underlying SPS-EQ scheme EQ. Specifically, our IKeyGen invokes EQ.Sign, which requires l group exponentiations for a vector of length l ($l = 4$ in HinToken). In Issue and Verify, the user needs to invoke EQ.Verify, which requires $l+3$ pairing operations. Moreover, the user computes 1 group exponentiation to generate *pmsg* and 13 group exponentiations (10 for the proof and 3 for EQ) to obtain the token t. The issuer computes 16 group exponentiations (2 for the ciphertext, 8 for the proof, and 6 for EQ) to generate *psig*.

For the communication costs, the signature of EQ consists of 3 group elements, so the token size in HinToken is $l+3$. During Issue, the user sends *pmsg*, which consists of 2 group elements, and the issuer replies *psig*, which consists of 5 group elements (2 for the ciphertext and 3 for the signature) and 5 scalars (for the proof).

Implementation Performance. We implemented our schemes in pure Rust (stable, version 1.88.0) using the pairing-friendly curve Bls12-381 [3]. For PMBT [29] and ATHM [13], we use the implementations provided by their authors.[7][8] Both implementations use curve25519-dalek, which does not support pairing but is much faster than Bls12-381. All implementations are benchmarked on a laptop with Apple M4 Pro chip and 24 GB RAM.

[7] https://github.com/mmaker/anonymous-tokens
[8] https://github.com/Microsoft/MacTok

Our benchmarks consider key generation, token issuance, and token verification. We compare the time costs of HinToken with PMBT and ATHM, and the results are shown in Table 2. We do not test the costs of Verify in PMBT and ATHM, since they are both privately verifiable, and the token validity will be verified when reading the bit. Our scheme has the worst performance, primarily due to the group exponentiation of Bls12-381 being slower than that of curve25519-dalek, as well as the pairing operation used in the SPS-EQ scheme. HinToken provides unique functionality and the time costs of HinToken are still acceptable in practice.

We also examine the performance of our reader-hiding and issuer-hiding HinToken schemes. Compared with HinToken, the additional overhead in HinToken-R mainly comes from EQ.ChgRep (in User_0) and EQ.Verify (in Sign) of the SPS-EQ signatures for the set $\mathcal{R}$ during the token issuance phase. Similarly, the additional overhead in HinToken-I mainly comes from EQ.ChgRep (in User_1) and EQ.Verify (in Verify) of the SPS-EQ signatures for the set I during the token issuance and verification phase. Two SPS-EQ signatures need to be verified in Verify, and we perform these two verifications in parallel in our implementation to optimize the performance.

In Table 2, we also show the token size in all schemes. Compared with others, our token size is larger but acceptable in practice. We omit the first element of the message (i.e., g_1) to reduce the token size.

8 Other Extensions and Discussion

8.1 Double-Hiding Constructions

Recall that HinToken-R cannot integrate HinToken-I in Section 6 to support reader-hiding since I in the latter construction reveals the reader's identity. We provide an alternative approach to achieve issuer-hiding and extend HinToken-R to support both reader-hiding and issuer-hiding, named HinToken-RI.

We still leverage mercurial signatures [17, 24] to randomize the issuer's public key along with the token. Instead of using SPS-EQ in Section 6, we require the user to prove the legitimacy of the blinded public key via an NIZK system. Specifically, assume that the public key set of issuers accept-

Table 2: Benchmarks for HinToken (on Bls12-381) and related schemes (on curve25519-dalek).

Scheme	KeyGen		Issuance			Verification		Token Size
	IKeyGen	RKeyGen	User_0	Sign	User_1	Verify	Read	
PMBT [29]	59.5 µs	–	29.6 µs	187.9 µs	230.1 µs	–	60.0 µs	360 B
ATHM [13]	45.9 µs	–	26.2 µs	220.8 µs	244.7 µs	–	36.9 µs	360 B
HinToken	1057.6 µs	59.1 µs	123.2 µs	1269.5 µs	1170.6 µs	2729.8 µs	53.6 µs	1008 B
HinToken-R	1106.2 µs	60.1 µs	709.8 µs	3795.4 µs	1204.0 µs	2786.9 µs	53.1 µs	1008 B
HinToken-I	1118.0 µs	61.9 µs	130.7 µs	1310.9 µs	3363.5 µs	2962.5 µs	56.1 µs	1008 B

able to the verifier is $I = \{pk_i\}_{i\in[q]} \in (\mathbb{G}_2^*)^{4q}$, $pk_I = pk_j \in I$, and the scalar used in randomization is ρ. We require the user to provide a valid proof for the following language $\mathcal{L}_I$.

$$\mathcal{L}_I = \left\{ \begin{array}{c|c} I \in \mathbb{G}_2^{4q} & \exists pk_j \in \mathbb{G}_2^4, \rho \in \mathbb{Z}_p^* \\ pk_j^* \in \mathbb{G}_2^4 & \text{s.t. } pk_j \in I, pk_j^* = (pk_j)^\rho \end{array} \right\}.$$

For each $pk_i \in I$, let $pk_i = (X_{i1}, \ldots, X_{i4}) \in \mathbb{G}_2^4$. We have $pk_j^* = (X_{j1}^*, \ldots, X_{j4}^*)$ where $X_{jk}^* = X_{jk}^\rho$ ($k \in [4]$). Given a one-hot vector $\mathbf{b} = (b_1, \ldots, b_n) \in \mathbb{Z}_p^n$ where $b_j = 1$ and $b_i = 0, \forall i \in [n] \setminus \{j\}$, we have $X_{jk} = \prod_{i=1}^n X_{ik}^{b_i} = \mathbf{X}_k^{\mathbf{b}}$, $X_{jk}^* = X_{jk}^\rho = \mathbf{X}_k^{\mathbf{u}}$, where $\mathbf{X}_k = (X_{1k}, \ldots, X_{nk})$ and $\mathbf{u} = (\rho b_1, \ldots, \rho b_n)$. Thus, we commit to $\mathbf{b}$ and $\mathbf{u}$ and split the proof into three parts. (1) $\mathbf{b}$ is a one-hot vector. (2) There is a randomness ρ such that $\mathbf{u} = \rho \cdot \mathbf{b}$. (3) $X_{jk}^* = \mathbf{X}_k^{\mathbf{u}}$ ($k \in [4]$). The third part can be done in batch proof.

We design an instantiation of NIZK system Π_I for $\mathcal{L}_I$ based on Bulletproofs [11]. The details of the above three parts will be introduced in Appendix D.

8.2 HinToken with Stronger Privacy

Our constructions in Section 5 and 6 reach reader and issuer anonymity based on an assumption that a malicious issuer (resp., reader) will not collude with all readers (resp., issuers). We believe this non-collusion assumption is reasonable, since a malicious issuer (resp., reader) needs to contact all readers (resp., issuers), which would be detected and significantly affect its reputation, making it unlikely in the real world.

We emphasize that, when all issuers and readers collude, HinToken-R and HinToken-I will degrade into our basic HinToken scheme, still satisfying unlinkability defined in Definition 15. To see it, note that in the definition of unlinkability, the adversary learns both *pmsg* (observed by the issuer in the scheme) and t (observed by the reader in the scheme) from the oracles, but it still cannot link the challenge tokens. In other words, unlinkability already accounts for such collusion.

In this section, we discuss how to extend HinToken-R and HinToken-I with stronger privacy, i.e., collusion-resistant. We believe that schemes in Section 5, 6, and this section represent a trade-off between performance and security.

Recall that in Section 5, the user randomizes the reader's public key y_j and g_1 with the same scalar γ. This ensures the correctness of decryption, but y_j will be immediately recognizable to an adversary who holds the corresponding secret key. To address this issue, the user additionally selects $w \xleftarrow{\$} \mathbb{Z}_p^*$ and computes $g^* \leftarrow g_1^{\gamma w^{-1}}$. In addition to computing $g_1^* \leftarrow g_1^{z^{-1}}$, the user computes $g_1^{**} \leftarrow g_1^{(wz)^{-1}}$ and $h^* \leftarrow h^{-w}$, and sets $pmsg = (g_1^*, h, y^*, g_1^{**}, h^*, g^*)$ (the legitimacy of y^* can be proved by an NIZK system similar to Appendix D). The issuer embeds a metadata bit s by computing $u_s \leftarrow (g^*)^r, v_s \leftarrow (y^*)^r$, and proves that u_s, v_s share the same randomness r in zero-knowledge. Since now (h, u_s, v_s) cannot be decrypt directly, the issuer has to sign two SPS-EQ signatures σ_1, σ_2 on message vectors $\mathbf{m}_1 = (g_1^*, h, v_s)$ and $\mathbf{m}_2 = (g_1^{**}, h^*, u_s)$, respectively. The user randomizes $(\sigma_1, \mathbf{m}_1)$ with the scalar z and $(\sigma_2, \mathbf{m}_2)$ with wz. Combining the two randomized message vectors, the metadata bit can be successfully read by the specified reader, thereby completing the construction of HinToken-R with stronger privacy.

For HinToken-I based on mercurial signatures in Section 6, the issuer's identity pk_j will be recognizable to the adversary with the corresponding secret key for the similar reasons. Griffy et al. [24] noted this weakness in mercurial signatures and provided a new mercurial signature scheme with stronger privacy, which can resist this collusion. The main idea is to ensure that each element of the public key is correctly generated over a distinct generator of the group. Inspired by this work, we can extend HinToken-I with stronger privacy.

8.3 Discussion

In addition to the solution introduced in Section 8.1, there is another way to implement both reader-hiding and issuer-hiding. In some scenarios, there is a root authority trusted by the reader. Then, the set I can be authenticated by this authority, and thus will not reveal the identity of the reader. This solution can support reader-hiding and issuer-hiding simultaneously, and can be extended to a hierarchical structure similar to PKIs by leveraging the idea of DACs.

Compared with the solutions such as [29] and [2], our scheme can be easily extended to support multiple metadata bits. This allows issuers to categorize users more finely. However, the number of metadata bits must be carefully chosen to avoid violating unlinkability.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. This work was partly supported by the National Cryptologic Science Fund of China (2025NCSF02027), the National Natural Science Foundation of China (62472323), the Major Program of Hubei Province (2023BAA27), and the Key R&D Program of Hubei Province (2024BAB018).

Ethical Considerations

In our anonymous token systems, issuers and readers (aka verifiers) are typically service providers in the real world, while users are those who request the service. Our schemes support unique properties, including designated-reader metadata bit, issuer-hiding, and reader-hiding. These properties may enable new applications beyond existing solutions, as discussed in Section 1.1. We acknowledge that our schemes may have positive or negative impacts on different stakeholders, which may not have been sufficiently considered during our research and we address this post facto. We now identify the key stakeholders for each application and provide a post-facto analysis of the potential impacts on them.

In scenarios such as bot protection, the basic HinToken allows the issuer to convey signals about users (web visitors) to a user-specified reader (web server). It can help the reader decide on actions, but it may also increase the user's concerns about unlinkability. In our schemes, we limit the metadata to a single bit and provide formal security proofs. The issuer-hiding HinToken can be applied to proofs of funds for visa applications. It can better protect user privacy (e.g., not revealing which bank holds the applicant's deposits), but it will also impose an additional auditing burden on the readers. The impacts on stakeholders of reader-hiding HinToken are similar. Although these two properties offer stronger anonymity for users, our schemes allow issuers (resp., readers) to independently choose and publicly declare which readers (resp., issuers) they accept, ensuring that anonymity cannot be easily abused. We believe that even with prior ethical analysis, we would adopt the same technical approach because current solutions strike a reasonable trade-off between new properties and security, minimizing negative impacts on stakeholders.

Open Science

We follow the open science policy of USENIX Security and release the source code of our work at https://doi.org/10.5281/zenodo.17946122.

References

[1] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In *EUROCRYPT*, pages 83–107, 2002.

[2] Foteini Baldimtsi, Lucjan Hanzlik, Quan Nguyen, and Aayush Yadav. Non-interactive anonymous tokens with private metadata bit. *Cryptology ePrint Archive*, 2025.

[3] Paulo SLM Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In *SCN*, pages 257–267, 2002.

[4] David Basin, Cas Cremers, Tiffany Hyun-Jin Kim, Adrian Perrig, Ralf Sasse, and Pawel Szalachowski. Arpki: Attack resilient public-key infrastructure. In *CCS*, pages 382–393, 2014.

[5] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In *CRYPTO*, pages 26–45, 1998.

[6] Fabrice Benhamouda, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. Publicly verifiable anonymous tokens with private metadata bit. *Cryptology ePrint Archive*, 2022.

[7] Jan Bobolz, Fabian Eidens, Stephan Krenn, Sebastian Ramacher, and Kai Samelin. Issuer-hiding attribute-based credentials. In *CANS*, pages 158–178, 2021.

[8] Dan Boneh and Victor Shoup. *A graduate course in applied cryptography*. 2020.

[9] Daniel Bosk, Davide Frey, Mathieu Gestin, and Guillaume Piolle. Hidden issuer anonymous credential. In *PoPETs*, pages 571–607, 2022.

[10] Stefan Brands. *Rethinking public key infrastructures and digital certificates: building in privacy*. 2000.

[11] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *SP*, pages 315–334, 2018.

[12] Jan Camenisch, Manu Drijvers, and Maria Dubovitskaya. Practical uc-secure delegatable credentials with attributes and their application to blockchain. In *CCS*, pages 683–699, 2017.

[13] Melissa Chase, F Betül Durak, and Serge Vaudenay. Anonymous tokens with stronger metadata bit hiding from algebraic macs. In *CRYPTO*, pages 418–449, 2023.

[14] Melissa Chase, Sarah Meiklejohn, and Greg Zaverucha. Algebraic macs and keyed-verification anonymous credentials. In *CCS*, pages 1205–1216, 2014.

[15] David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, pages 199–203, 1983.

[16] David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In *CRYPTO*, pages 89–105, 1992.

[17] Elizabeth C Crites and Anna Lysyanskaya. Delegatable anonymous credentials from mercurial signatures. In *CT-RSA*, pages 535–555, 2019.

[18] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. In *PoPETs*, 2018.

[19] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[20] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.

[21] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *Journal of Cryptology*, 32(2):498–546, 2019.

[22] Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind schnorr signatures and signed elgamal encryption in the algebraic group model. In *EUROCRYPT*, pages 63–95, 2020.

[23] Shafi Goldwasser, Silvio Micali, and Ronald L Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[24] Scott Griffy, Anna Lysyanskaya, Omid Mir, Octavio Perez Kempner, and Daniel Slamanig. Delegatable anonymous credentials from mercurial signatures with stronger privacy. In *ASIACRYPT*, pages 296–325, 2024.

[25] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In *ASIACRYPT*, pages 491–511, 2014.

[26] Lucjan Hanzlik. Non-interactive blind signatures for random messages. In *EUROCRYPT*, pages 722–752, 2023.

[27] Stanislaw Jarecki, Aggelos Kiayias, and Hugo Krawczyk. Round-optimal password-protected secret sharing and t-pake in the password-only model. In *ASIACRYPT*, pages 233–253, 2014.

[28] Ioanna Karantaidou, Omar Renawi, Foteini Baldimtsi, Nikolaos Kamarinakis, Jonathan Katz, and Julian Loss. Blind multisignatures for anonymous tokens with decentralized issuance. In *CCS*, pages 1508–1522, 2024.

[29] Ben Kreuter, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. Anonymous tokens with private metadata bit. In *CRYPTO*, pages 308–336, 2020.

[30] Daniel Lewin and Salil Vadhan. Checking polynomial identities over any field: Towards a derandomization? In *STOC*, pages 438–447, 1998.

[31] Omid Mir, Balthazar Bauer, Scott Griffy, Anna Lysyanskaya, and Daniel Slamanig. Aggregate signatures with versatile randomization and issuer-hiding multi-authority anonymous credentials. In *CCS*, pages 30–44, 2023.

[32] Omid Mir, Daniel Slamanig, Balthazar Bauer, and René Mayrhofer. Practical delegatable anonymous credentials from equivalence class signatures. In *PoPETs*, 2023.

[33] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO*, pages 129–140, 1991.

[34] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.

[35] Manoj Prabhakaran and Mike Rosulek. Rerandomizable rcca encryption. In *CRYPTO*, pages 517–534, 2007.

[36] The-Anh Ta, Xiangyu Hui, and Sid Chi-Kin Chau. Ring referral: Efficient publicly verifiable ad hoc credential scheme with issuer and strong user anonymity for decentralized identity and more. In *SP*, pages 184–202, 2025.

A Security Proofs for HinToken

Theorem 1. If the SPS-EQ scheme EQ is existentially unforgeable under adaptive chosen-message attacks, then $\mathsf{HinToken}$ is one-more unforgeable.

Proof. Let $n \geq 0$. Assume that there is a PPT adversary $\mathcal{A}$ with an advantage $\mathsf{Adv}^{omuf}_{\mathcal{A},n}(\lambda)$ in game $\mathsf{OMUF}_{\mathcal{A},n}(\lambda)$, then we can use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for game $\mathsf{EUF\text{-}CMA}_{\mathcal{B}}(\mathsf{BG}, n)$ as follows.

1. $\mathcal{B}$ is given BG, pk, and access to O_{Sign}.
2. $\mathcal{B}$ sets $pp \leftarrow \mathsf{BG}$, $pk_I \leftarrow pk$, $q_0 \leftarrow 0$, and $q_1 \leftarrow 0$, and generates a reader's key pair $(pk_R, sk_R) \leftarrow \mathsf{RKeyGen}(pp)$, where $sk_R = x$ and $pk_R = y$. Then, $\mathcal{B}$ runs $\mathcal{A}(pp, pk_I, pk_R)$.
3. When $\mathcal{A}$ queries its signing oracle on $(pk'_R, s, pmsg)$, where $pk'_R = y'$ and $pmsg = (g_1^*, h)$, $\mathcal{B}$ returns $\perp$ if $s \notin \{0,1\}$. Otherwise, $\mathcal{B}$ computes u_s, v_s, and π_v as in $\mathsf{HinToken}$. Then, $\mathcal{B}$ queries O_{Sign} on the message $\mathbf{m} = (g_1^*, h, u_s, v_s)$ and obtains σ^*. Finally, $\mathcal{B}$ sets $q_s \leftarrow q_s + 1$ and returns $(u_s, v_s, \pi_v, \sigma^*)$ to $\mathcal{A}$.

4. When $\mathcal{A}$ queries its reading oracle on a token t, $\mathcal{B}$ returns $\mathsf{Read}(sk_R, pk_I, t)$.

5. When $\mathcal{A}$ outputs tokens $\{t_i\}_{i\in[n+1]}$, $\mathcal{B}$ randomly selects a token from them as its output if $\mathcal{A}$ wins, and aborts otherwise.

We now compute the advantage $\mathsf{Adv}^{euf}_{\mathcal{B}}(\mathsf{BG}, l)$ of $\mathcal{B}$. It is not difficult to see that if $\mathcal{A}$ does not win, $\mathcal{B}$ will not win either. Therefore, we have

$$\begin{aligned}\mathsf{Adv}^{euf}_{\mathcal{B}}(\mathsf{BG}, l) &= \Pr[\mathcal{B} \text{ wins} \mid \mathcal{A} \text{ wins}] \cdot \Pr[\mathcal{A} \text{ wins}] \\ &= \Pr[\mathcal{B} \text{ wins} \mid \mathcal{A} \text{ wins}] \cdot \mathsf{Adv}^{omuf}_{\mathcal{A},n}(\lambda).\end{aligned}$$

Without loss of generality, assume that when $\mathcal{A}$ wins, $q_0 = q_1 = n$, and the output of the metadata bit reading algorithm on all those tokens is s. We split the message vectors $\mathcal{B}$ submitted to O_{Sign} into two sets:

$$\begin{aligned}\mathcal{M}_0 &= \left\{ \left(g_1^{*(i)}, h^{(i)}, g_1^{r(i)}, y^{r(i)} \right) \right\}_{i\in[n]}, \\ \mathcal{M}_1 &= \left\{ \left(g_1^{*(i)}, h^{(i)}, g_1^{r(i)}, y^{r} h^{(i)} \right) \right\}_{i\in[n]}.\end{aligned}$$

Since $\mathsf{HinToken}$ restricts the message vector in a valid token to have g_1 as the first element, each valid token must consist of a unique representative of an equivalence class and the corresponding signature on the representative. We consider the following two cases.

First, there is at least one message vector in the tokens $\{t_i\}_{i\in[n+1]}$ that does not belong to any equivalence classes of message vectors in $\mathcal{M}_0$ and $\mathcal{M}_1$. In this case, $\mathcal{B}$ wins if it outputs the tokens corresponding to the new message vectors. Since $\mathcal{B}$ randomly selects a token in $\{t_i\}_{i\in[n+1]}$, the probability of $\mathcal{B}$ succeeding in this case is at least $\frac{1}{n+1}$.

Second, the message vectors in the tokens $\{t_i\}_{i\in[l+1]}$ all belong to equivalence classes of message vectors in $\mathcal{M}_0$ or $\mathcal{M}_1$. Since there are only n message vectors in $\mathcal{M}_s$, at least one message vector in $\mathcal{M}_{1-s}$ should be explained to an equivalence class of message vectors in the form of $\mathcal{M}_s$. Let $(g_1^*, h, g_1^r, y'^r h^{1-s})$ be that message vector in $\mathcal{M}_{1-s}$. The probability that r chosen by the issuer satisfies $y'^r h^{1-s} = y^r h^s$ is $\frac{1}{p}$. Thus, the probability that this case occurs is at most $\frac{n}{p}$ by union bound. In this case, $\mathcal{B}$ will never win the game.

Combine the above two cases and denote the occurrence of the second case as coll, we have

$$\begin{aligned}\Pr[\mathcal{B} \text{ wins} \mid \mathcal{A} \text{ wins}] &= \Pr[\mathcal{B} \text{ wins} \mid \overline{\mathsf{coll}}] \cdot \Pr[\overline{\mathsf{coll}} \mid \mathcal{A} \text{ wins}] \\ &\geq \frac{1}{n+1}(1 - \Pr[\mathsf{coll} \mid \mathcal{A} \text{ wins}]) \\ &\geq \frac{p-n}{p(n+1)}.\end{aligned}$$

In conclusion, the advantage $\mathsf{Adv}^{euf}_{\mathcal{B}}(\mathsf{BG}, l)$ of $\mathcal{B}$ in $\mathsf{EUF\text{-}CMA}_{\mathcal{B}}(\mathsf{BG}, l)$ is

$$\mathsf{Adv}^{euf}_{\mathcal{B}}(\mathsf{BG}, l) \geq \frac{p-n}{p(n+1)} \mathsf{Adv}^{omuf}_{\mathcal{A},n}(\lambda).$$

Since EQ is existentially unforgeable under adaptive chosen-message attacks, there is a negligible function negl such that $\mathsf{Adv}^{euf}_{\mathcal{B}}(\mathsf{BG}, l) \leq \mathsf{negl}(\lambda)$. Thus, we have

$$\mathsf{Adv}^{omuf}_{\mathcal{A},n}(\lambda) \leq \frac{p(n+1)}{p-n} \mathsf{negl}(\lambda),$$

which implies that the advantage of the adversary $\mathcal{A}$ in $\mathsf{OMUF}_{\mathcal{A},n}(\lambda)$ is also negligible. □

Theorem 2. If the NIZK system Π_{v} provides special soundness, the SPS-EQ scheme EQ perfectly adapts signatures under malicious keys, and the inverse DDH assumption holds in $\mathbb{G}_1$, then $\mathsf{HinToken}$ is unlinkable.

Proof. The proof is done by a hybrid argument. Let $\mathcal{E}$ be a witness extractor of Π_{v} and $m > 0$.

Hy_0 This is the game $\mathsf{UNLINK}_{\mathcal{A},m}(\lambda)$ defined in Figure 4.

Hy_1 When $\mathcal{A}_0$ outputs an issuer's public key pk_I, we obtain the issuer's secret key sk_I in GGM. This hybrid and Hy_0 are identical to the adversary.

Hy_2 We replace $\mathsf{User}_1(psig, st)$ (for generating challenge tokens and answering O_{User_1}) with the following procedure. Parse $psig = (u_s, v_s, \pi_v, \sigma^*)$ and $st = (h, z)$. If π_v and σ^* pass verification, we compute $\sigma \leftarrow \mathsf{EQ.Sign}(sk_I, (g_1, h^z, u_s^z, v_s^z))$ and output $t = ((g_1, h^z, u_s^z, v_s^z), \sigma)$ instead of invoking $\mathsf{EQ.ChgRep}$. Since EQ perfectly adapts signatures under malicious keys, this hybrid and Hy_1 are indistinguishable to the adversary. That is, we have

$$\left| \mathsf{Adv}^{\mathsf{Hy}_1}_{\mathcal{A},m}(\lambda) - \mathsf{Adv}^{\mathsf{Hy}_2}_{\mathcal{A},m}(\lambda) \right| \leq q_0 \cdot \mathsf{negl}_1(\lambda),$$

where negl_1 is the advantage of the adversary in EQ.

Hy_3 We use $\mathcal{E}$ to extract the witness (r, s) of $\mathcal{A}_1$ from π_v if it is valid. Since the special soundness of Π_{v}, this hybrid and Hy_2 are indistinguishable to the adversary. That is, we have

$$\left| \mathsf{Adv}^{\mathsf{Hy}_2}_{\mathcal{A},m}(\lambda) - \mathsf{Adv}^{\mathsf{Hy}_3}_{\mathcal{A},m}(\lambda) \right| \leq q_0 \cdot \mathsf{negl}_2(\lambda),$$

where negl_2 is the advantage of the adversary in Π_{v}.

Hy_4 Instead of signing on (g_1, h^z, u_s^z, v_s^z) in Hy_2, we output $t \leftarrow \mathsf{EQ.Sign}(sk_I, (g_1, h^z, g_1^{zr}, y^{zr} h^{zs}))$. This hybrid and Hy_3 are identical to the adversary.

Hy_5 Instead of signing on $(g_1, h^z, g_1^{zr}, y^{zr} h^{zs})$ in Hy_4, we output $t \leftarrow \mathsf{EQ.Sign}(sk_I, (g_1, h', g_1^{zr}, y^{zr} (h')^s))$, where $h' \xleftarrow{\$} \mathbb{G}_1$. This hybrid and Hy_4 are indistinguishable to the adversary if the DDH assumption hold in $\mathbb{G}_1$. That is, we have

$$\left| \mathsf{Adv}^{\mathsf{Hy}_4}_{\mathcal{A},m}(\lambda) - \mathsf{Adv}^{\mathsf{Hy}_5}_{\mathcal{A},m}(\lambda) \right| \leq q_0 \cdot \mathsf{Adv}^{ddh}_{\mathcal{B}_1}(\lambda).$$

Hy_6 When $\mathcal{A}_0$ outputs a reader's public key $pk_R = y$, we also obtain the reader's secret key x such that $y = g_1^x$ in GGM. Then, instead of signing on $(g_1, h', g_1^{zr}, y^{zr}(h')^s)$ in Hy_5, we output $t \leftarrow \mathsf{EQ.Sign}(sk_I, (g_1, h', g_1^{zr}, g_1^{zrx}(h')^s))$. This hybrid and Hy_5 are identical to the adversary.

Hy_7 To answer the query of $\mathcal{A}_1$ to O_{User_0}, we choose $g_1^* \xleftarrow{\$} \mathbb{G}_1$ instead of computing $g_1^* \leftarrow g_1^{z^{-1}}$. This hybrid and Hy_6 are indistinguishable to the adversary if the inverse DDH assumption hold in $\mathbb{G}_1$. That is, we have

$$\left|\mathsf{Adv}^{\mathsf{Hy}_6}_{\mathcal{A},m}(\lambda) - \mathsf{Adv}^{\mathsf{Hy}_7}_{\mathcal{A},m}(\lambda)\right| \leq q_0 \cdot \mathsf{Adv}^{inv}_{\mathcal{B}_2}(\lambda).$$

We now prove that Hy_4 and Hy_5 are indistinguishable. This is done by a sequence of hybrids $\{\mathsf{Hy}_{5,i}\}_{i\in\{0,\ldots,q_0\}}$. In $\mathsf{Hy}_{5,i}$, the first i invocations to User_1 uses a random h', and the others still use h^z. Then, $\mathsf{Hy}_{5,0}$ are Hy_4 are identical, and Hy_{5,q_0} are Hy_5 are identical. For $i \in [q_0]$, if the adversary can distinguish between $\mathsf{Hy}_{5,i-1}$ and $\mathsf{Hy}_{5,i}$, then we can use the adversary to construct an adversary $\mathcal{B}_1$ for the DDH game. Specifically, $\mathcal{B}_1$ receives a tuple (g_1, g_1^*, h', h) and uses it to answer the i-th invocation. At the end, $\mathcal{B}_1$ outputs the bit output by the adversary. Note that if $\mathcal{B}_1$ receives a DH-tuple, then the adversary is in $\mathsf{Hy}_{5,i-1}$; otherwise, the adversary is in $\mathsf{Hy}_{5,i}$. Therefore, the advantage of the adversary is no greater than the advantage of $\mathcal{B}_1$.

The proof for the indistinguishability between Hy_6 and Hy_7 is similar. Specifically, $\mathcal{B}_2$ receives a tuple (g_1, g_1^z, g_1^*) and uses it to invoke the i-th $\mathsf{EQ.Sign}$ on $(g_1, h', (g_1^z)^r, (g_1^z)^{rx}(h')^s)$. At the end, $\mathcal{B}_2$ outputs the bit output by the adversary. The advantage of the adversary to distinguish two sub-hybrids is no greater than the advantage of $\mathcal{B}_2$.

Since the tokens are random message vectors (conditioned on the correctness of Read) and corresponding signatures in Hy_7, the advantage of $\mathcal{A}$ is $\mathsf{Adv}^{\mathsf{Hy}_7}_{\mathcal{A},m}(\lambda) = \frac{2}{q_0 - q_1}$. Therefore,

$$\mathsf{Adv}^{unlink}_{\mathcal{A},m}(\lambda) = \mathsf{Adv}^{\mathsf{Hy}_0}_{\mathcal{A},m}(\lambda) \leq \frac{2}{q_0 - q_1} + q_0 \cdot \mathsf{negl}(\lambda),$$

where negl absorbs the above negligible functions. □

Theorem 3. If the NIZK system Π_{v} provides special honest verifier zero-knowledge, the SPS-EQ scheme EQ is existentially unforgeable under adaptive chosen-message attacks, and the DDH assumption holds in $\mathbb{G}_1$, then $\mathsf{HinToken}$ provides private metadata bit.

Proof. The proof is done by a hybrid argument. Let $\mathcal{S}$ be a simulator of Π_{v}.

Hy_0 This is the game $\mathsf{PMB}_{\mathcal{A}}(\lambda)$ defined in Figure 5.

Hy_1 We replace $\Pi_{\mathrm{v}}.\mathsf{Prove}$ in $\mathsf{Sign}_1(sk_I, pk_R, s, pmsg)$ (for answering $\mathcal{O}^b_{\mathsf{Sign}}$ and $O_{\mathsf{Sign}'}$) with the simulator $\mathcal{S}$. Since Π_{v} is zero-knowledge, this hybrid and Hy_0 are indistinguishable to the adversary. That is, we have

$$\left|\mathsf{Adv}^{\mathsf{Hy}_0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathsf{Hy}_1}_{\mathcal{A}}(\lambda)\right| \leq q \cdot \mathsf{negl}_1(\lambda),$$

where q is the number of times that $\mathcal{A}$ queries $O_{\mathsf{Sign}'}$ and $\mathcal{O}^b_{\mathsf{Sign}}$ and negl_1 is the advantage of the adversary in Π_{v}.

Hy_2 We set a list $\mathcal{L} = \emptyset$ at the beginning of the game. When $\mathcal{A}$ queries $O_{\mathsf{Sign}'}$ on $(s, pmsg)$, we obtain z in GGM, where $pmsg = (h, g_1^*)$ and $g_1^* = g_1^{-z}$, and add (z, h, u_s, v_s, s) to $\mathcal{L}$. This hybrid and Hy_1 are identical to the adversary.

Hy_3 When $\mathcal{A}$ queries O_{Read} on $t = ((t_1, t_2, t_3, t_4), \sigma)$ before the challenge is generated, if $\mathsf{EQ.Verify}(pk_I, t) = 1$ and there is a tuple $(z, h, u_s, v_s, s) \in \mathcal{L}$ such that $h^z = t_2$, $u_s^z = t_3$, and $v_s^z = t_4$, we return s to $\mathcal{A}$, otherwise, return $\perp$. Since EQ is existentially unforgeable under adaptive chosen-message attacks, this hybrid and Hy_2 are indistinguishable to the adversary. That is, we have

$$\left|\mathsf{Adv}^{\mathsf{Hy}_2}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathsf{Hy}_3}_{\mathcal{A}}(\lambda)\right| \leq \mathsf{Adv}^{euf}_{\mathcal{B}_1}(\mathsf{BG}, l).$$

Hy_4 Instead of selecting $x \xleftarrow{\$} \mathbb{Z}_p$ and generating a reader's public key $y \leftarrow g_1^x$, we directly select $y \xleftarrow{\$} \mathbb{G}_1$. This hybrid and Hy_3 are identical to the adversary.

Hy_5 Instead of computing $v_s \leftarrow y^r h^s$ when answering $O_{\mathsf{Sign}'}$ and $\mathcal{O}^b_{\mathsf{Sign}}$, we select $y' \in \mathbb{G}_1$ and compute $v_s \leftarrow y' h^s$. This hybrid and Hy_4 are indistinguishable to the adversary if the DDH assumption hold in $\mathbb{G}_1$. That is, we have

$$\left|\mathsf{Adv}^{\mathsf{Hy}_4}_{\mathcal{A},m}(\lambda) - \mathsf{Adv}^{\mathsf{Hy}_5}_{\mathcal{A},m}(\lambda)\right| \leq q \cdot \mathsf{Adv}^{ddh}_{\mathcal{B}_2}(\lambda).$$

The proof for the indistinguishability between Hy_4 and Hy_5 is done by a sequence of hybrids $\{\mathsf{Hy}_{5,i}\}_{i\in\{0,\ldots,q\}}$. In $\mathsf{Hy}_{5,i}$, the first i queries are answered with $y'h^s$, and the others are still answered with $y^r h^s$. Then, $\mathsf{Hy}_{5,0}$ are Hy_4 are identical, and Hy_{5,q_0} are Hy_5 are identical. For $i \in [q_0]$, if the adversary can distinguish between $\mathsf{Hy}_{5,i-1}$ and $\mathsf{Hy}_{5,i}$, then we can use the adversary to construct an adversary $\mathcal{B}_2$ for the DDH game. Specifically, $\mathcal{B}_2$ receives a tuple (g_1, y, g_1^r, y'). Then, y is set as the reader's public key, and the tuple is used to answer the i-th query. At the end, $\mathcal{B}_2$ outputs the bit output by the adversary. Note that if $\mathcal{B}_2$ receives a DH-tuple, then the adversary is in $\mathsf{Hy}_{5,i-1}$; otherwise, the adversary is in $\mathsf{Hy}_{5,i}$. Therefore, the advantage of the adversary is no greater than the advantage of $\mathcal{B}_2$.

Since the challenge contains a random t_4 in Hy_5, the advantage of $\mathcal{A}$ is $\mathsf{Adv}^{\mathsf{Hy}_5}_{\mathcal{A}}(\lambda) = \frac{1}{2}$. Therefore,

$$\mathsf{Adv}^{pmb}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathsf{Hy}_0}_{\mathcal{A}}(\lambda) \leq \frac{1}{2} + q \cdot \mathsf{negl}(\lambda),$$

where negl absorbs the above negligible functions. □

B Security Proofs for HinToken-R

In the one-more unforgeability game $\mathsf{OMUF}_{\mathcal{A},n}(\lambda)$ of reader-hiding HinToken, the adversary $\mathcal{A}$ is given access to an additional oracle, which allows $\mathcal{A}$ to add public keys of its choice into the set $\mathcal{R}$ that initially contains pk_R only. The signing oracle O_{Sign} then only takes $(s, pmsg)$ as input and returns $\mathsf{Sign}(sk_I, \mathcal{R}, s, pmsg)$. The proof for the one-more unforgeability of HinToken-R is similar to that of Theorem 1 with corresponding modifications.

Theorem 4. If the SPS-EQ scheme EQ is existentially unforgeable under adaptive chosen-message attacks, then HinToken-R is one-more unforgeable.

In the unlinkability game $\mathsf{UNLINK}_{\mathcal{A},m}(\lambda)$ of reader-hiding HinToken, the adversary $\mathcal{A}$ outputs the set $\mathcal{R}$ instead of pk_R, and all oracles uses $\mathcal{R}$ to answer queries. The proof for the unlinkability of HinToken-R is similar to that of Theorem 2 with corresponding modifications.

Theorem 5. If the NIZK system Π_{v} provides special soundness, the SPS-EQ scheme EQ perfectly adapts signatures under malicious keys, and the inverse DDH assumption holds in $\mathbb{G}_1$, then HinToken-R is unlinkable.

In the private metadata bit game $\mathsf{PMB}_{\mathcal{A}}(\lambda)$ of reader-hiding HinToken, the challenger outputs a (trusted) set $\mathcal{R}$ and corresponding key pairs (instead of (pk_R, sk_R)) that will be used in all oracles, and the reading oracle O_{Read} takes an additional input pk_R and checks whether this public key is legitimate before answering. The proof for the private metadata bit of HinToken-R additionally relies on the existential unforgeability of the underlying SPS-EQ scheme. Roughly speaking, this assumption prevents users from using public keys outside $\mathcal{R}$.

Theorem 6. If the NIZK system Π_{v} provides special honest verifier zero-knowledge, the SPS-EQ scheme EQ is existentially unforgeable under adaptive chosen-message attacks and is existentially unforgeable under adaptive chosen-message attacks, and the DDH assumption holds in $\mathbb{G}_1$, then HinToken-R provides private metadata bit.

Theorem 7. If the SPS-EQ scheme EQ perfectly adapts signatures under malicious keys and the DDH assumption holds in $\mathbb{G}_1$, then HinToken-R provides reader anonymity.

Proof. The proof is done by a hybrid argument.

Hy_0 This is the game $\mathsf{RA}_{\mathcal{A}}(\lambda)$ defined in Figure 6.

Hy_1 When $\mathcal{A}_0$ outputs an issuer's public key pk_{I2}, we obtain the issuer's secret key sk_{I2} in GGM. This hybrid and Hy_0 are identical to the adversary.

Hy_2 We replace $\mathsf{EQ.ChgRep}$ in $\mathsf{User}_0(pk_I, \mathcal{R}, pk_b)$ with the following procedure. Set $g^* \leftarrow g_1^{\gamma}$ and $y^* \leftarrow pk_b^{\gamma}$ and compute $\sigma_b^* \leftarrow \mathsf{EQ.Sign}(sk_{I2}, (g^*, y^*))$. Since EQ perfectly adapts signatures under malicious keys, this hybrid and Hy_1 are indistinguishable to the adversary. That is, we have

$$\left|\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_1}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_2}(\lambda)\right| \leq \mathsf{negl}_1(\lambda),$$

where negl_1 is the advantage of the adversary in EQ.

Hy_3 We choose $b \xleftarrow{\$} \{0,1\}$ before generating (pk_0, sk_0) and (pk_1, sk_1). This hybrid and Hy_2 are identical to the adversary.

Hy_4 We replace y^* with $y' \xleftarrow{\$} \mathbb{G}_1$ when invoking $\mathsf{EQ.Sign}$. This hybrid and Hy_3 are indistinguishable to the adversary if the DDH assumption hold in $\mathbb{G}_1$. That is, we have

$$\left|\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_3}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_4}(\lambda)\right| \leq \mathsf{Adv}_{\mathcal{B}}^{ddh}(\lambda).$$

If the adversary can distinguish between Hy_3 and Hy_4, then we can use the adversary to construct an adversary $\mathcal{B}$ for the DDH game. Specifically, $\mathcal{B}$ receives a tuple $(g_1, y, g_1^{\gamma}, y')$. Then, y is set as pk_b, and we compute $(pk_{1-b}, sk_{1-b}) \leftarrow \mathsf{RKeyGen}(pp)$. At the end, $\mathcal{B}$ outputs 1 if the adversary wins. Note that if $\mathcal{B}$ receives a DH-tuple, then the adversary is in Hy_3; otherwise, the adversary is in Hy_4. Therefore, the advantage of the adversary is no greater than that of $\mathcal{B}$.

Since y' is random in Hy_4, the advantage of $\mathcal{A}$ is $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_4}(\lambda) = \frac{1}{2}$. Therefore,

$$\mathsf{Adv}_{\mathcal{A}}^{ra}(\lambda) = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_0}(\lambda) \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where negl absorbs the above negligible functions. □

C Security Proofs for HinToken-I

In the one-more unforgeability game $\mathsf{OMUF}_{\mathcal{A},n}(\lambda)$ of issuer-hiding HinToken, the challenger outputs a (trusted) set I and corresponding key pairs (instead of (pk_I, sk_I)) that will be used in all oracles, and the signing oracle O_{Sign} takes an additional input pk_I and checks whether this public key is legitimate before answering. The input of Verify and Read need to be modified accordingly. Importantly, one of the winning conditions $\forall i, j \in [n+1], t_i \neq t_j$ should be changed to $\mathbf{t}_i \neq \mathbf{t}_j$, where $t_i = (\mathbf{t}_i, \sigma_i)$ and $t_j = (\mathbf{t}_j, \sigma_j)$. This change means that our one-more unforgeability for issuer-hiding HinToken is downgraded to that defined in [29]. The proof for the one-more unforgeability of HinToken-I additionally relies on the existential unforgeability of the underlying SPS-EQ scheme. Roughly speaking, this assumption prevents the user from using public keys outside I.

Theorem 8. If the mercurial signature scheme EQ and the SPS-EQ scheme EQ_2 are existentially unforgeable under adaptive chosen-message attacks, then HinToken is one-more unforgeable.

In the unlinkability game $\mathsf{UNLINK}_{\mathcal{A},m}(\lambda)$ of issuer-hiding HinToken, the adversary $\mathcal{A}$ outputs the set I instead of pk_I, and all oracles uses I to answer queries. The proof for the unlinkability of HinToken-I is similar to that of Theorem 2 with corresponding modifications.

Theorem 9. If the NIZK system Π_v provides special soundness, the mercurial signature scheme EQ perfectly adapts signatures under malicious keys, and the inverse DDH assumption holds in $\mathbb{G}_1$, then HinToken is unlinkable.

The modification of the private metadata bit game $\mathsf{PMB}_{\mathcal{A}}(\lambda)$ of issuer-hiding HinToken is similar to that in the one-more unforgeability game $\mathsf{OMUF}_{\mathcal{A},n}(\lambda)$ of issuer-hiding HinToken. The proof for the private metadata bit of HinToken-I is similar to that of Theorem 3 with corresponding modifications.

Theorem 10. If the NIZK system Π_v provides special honest verifier zero-knowledge, the mercurial signature scheme EQ is existentially unforgeable under adaptive chosen-message attacks, and the DDH assumption holds in $\mathbb{G}_1$, then HinToken provides private metadata bit.

Theorem 11. If the mercurial signature scheme EQ is original-hiding and the SPS-EQ scheme EQ_2 perfectly adapts signatures under malicious keys, then HinToken provides issuer anonymity.

Proof. The proof is done by a hybrid argument.

Hy_0 This is the game $\mathsf{IA}_{\mathcal{A}}(\lambda)$ defined in Figure 7.

Hy_1 When $\mathcal{A}_0$ outputs an issuer's public key pk_{R2}, we obtain the issuer's secret key sk_{R2} in GGM. This hybrid and Hy_0 are identical to the adversary.

Hy_2 We replace EQ_2.ChgRep in Issue with the following procedure. Set $pk_b^* \leftarrow pk_b^{\rho}$ and compute $\sigma_b^* \leftarrow \mathsf{EQ}_2.\mathsf{Sign}(sk_{I2}, pk_b)$. Since EQ_2 perfectly adapts signatures under malicious keys, this hybrid and Hy_1 are indistinguishable to the adversary. That is,

$$\left|\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_1}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_2}(\lambda)\right| \leq \mathsf{negl}_1(\lambda),$$

where negl_1 is the adversary's advantage in EQ_2.

Hy_3 We choose $b \xleftarrow{\$} \{0,1\}$ before generating (pk_0, sk_0) and (pk_1, sk_1). This hybrid and Hy_2 are identical to the adversary.

Hy_4 We replace EQ.ConvertSig in Issue with the following procedure. Compute $sk' \leftarrow \mathsf{EQ.ConvertSK}(sk_b, \rho)$ and $\sigma^* \leftarrow \mathsf{EQ.Sign}(sk', (t_1, t_2, t_3, t_4))$. Since EQ is original-hiding, this hybrid and Hy_3 are indistinguishable to the adversary. That is,

$$\left|\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_3}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_4}(\lambda)\right| \leq \mathsf{negl}_2(\lambda),$$

where negl_2 is the adversary's advantage in EQ.

Since σ^* is the signature for a random in pk_b^*, the advantage of $\mathcal{A}$ is $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_4}(\lambda) = \frac{1}{2}$. Therefore,

$$\mathsf{Adv}_{\mathcal{A}}^{ia}(\lambda) = \mathsf{Adv}_{\mathcal{A}}^{\mathsf{Hy}_0}(\lambda) \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where negl absorbs the above negligible functions. □

D NIZK of Blinded Key Legitimacy

For vectors of field elements $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$, we use $\mathbf{x} \circ \mathbf{y} = (x_1 y_1, \ldots, x_n y_n)$ for the entry-wise product and $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^n x_i y_i$ for the standard inner product. For the vector $\mathbf{g} = (g_1, \ldots, g_n) \in \mathbb{G}^n$ and $\mathbf{u} = (u_1, \ldots, u_n) \in \mathbb{Z}_p^n$, we define $\langle \mathbf{u}, \mathbf{g} \rangle = \prod_{i=1}^n g_i^{u_i} = \mathbf{g}^{\mathbf{u}}$.

Our instantiation of the NIZK system Π_I for $\mathcal{L}_I$ (see Section 8.1) is based on Bulletproofs [11]. Recall that the blinded key $pk_j^* = (X_{j1}^*, \ldots, X_{j4}^*)$ where $X_{jk}^* = X_{jk}^{\rho}$ ($k \in [4]$). For the committed vectors $\mathbf{b}$ and $\mathbf{u}$, we need to prove three parts. Based on these proofs, we can obtain an NIZK through the Fiat-Shamir transformation [20].

D.1 Proof for One-Hot Vector

The prover commits to $\mathbf{b}$ using the generalized Pedersen commitment [33]. In this part, the prover proves that $\mathbf{b}$ is a one-hot vector similarly to the range proof [11].

Let $\mathbf{d} \leftarrow \mathbf{b} - \mathbf{1}^n \in \mathbb{Z}_p^n$. The prover samples $r_d \in \mathbb{Z}_p$ and computes the commitment to $\mathbf{d}$ as $c_{\mathbf{d}} = h^{r_d} \mathbf{h}^{\mathbf{d}}$. The language $\mathcal{L}_b$ for proving that $\mathbf{b}$ is a one-hot vector committed in $c_{\mathbf{b}}$ is defined as follows:

$$\mathcal{L}_b = \left\{ \begin{array}{c} c_{\mathbf{b}}, c_{\mathbf{d}}, h \in \mathbb{G}_2, \\ \mathbf{g}, \mathbf{h} \in \mathbb{G}_2^n \end{array} \middle| \begin{array}{l} \exists\, \mathbf{b} \in \mathbb{Z}_p^n, r_b, r_d \in \mathbb{Z}_p, \text{ s.t.} \\ c_{\mathbf{b}} = h^{r_b} \mathbf{g}^{\mathbf{b}}, c_{\mathbf{d}} = h^{r_d} \mathbf{h}^{\mathbf{d}}, \\ \mathbf{d} = \mathbf{b} - \mathbf{1}^n \in \mathbb{Z}_p^n, \\ \mathbf{b} \circ \mathbf{d} = \mathbf{0}^n, \langle \mathbf{b}, \mathbf{1}^n \rangle = 1 \end{array} \right\}$$

The instantiation Π_b for $\mathcal{L}_b$ is shown in Figure 9. The key idea is to construct two special polynomials $l(x)$ and $r(x)$ using $\mathbf{b}$ and $\mathbf{d}$ as part of the coefficients, where the zero-coefficient of $\langle l(x), r(x) \rangle$ has a special form if and only if $\mathbf{b}$ is a one-hot vector.

Let $t(x) = \langle l(x), r(x) \rangle = \sum_{i=0}^2 t_i \cdot x^i$. Specifically, the zero-coefficient of $\langle l(x), r(x) \rangle$ is

$$\begin{aligned} t_0 = {} & \langle \mathbf{b}, \mathbf{y}^n \circ \mathbf{d} \rangle + z \cdot \langle \mathbf{b} - \mathbf{d}, \mathbf{y}^n \rangle + z^2 \cdot \langle \mathbf{b}, \mathbf{1}^n \rangle \\ & - z^2 \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle - z^3 \cdot \langle \mathbf{1}^n, \mathbf{1}^n \rangle. \end{aligned}$$

If $\mathbf{d} = \mathbf{b} - \mathbf{1}^n \in \mathbb{Z}_p^n \ \wedge \ \mathbf{b} \circ \mathbf{d} = \mathbf{0}^n \ \wedge \ \langle \mathbf{b}, \mathbf{1}^n \rangle = 1$, then $t_0 = z \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle + z^2 \cdot \langle \mathbf{1}^n, \mathbf{y}^n \rangle + z^2 - z^3 \cdot n$, which means that t_0 is a function of y and z and V can compute t_0 itself. Furthermore, P commits to t_1 and t_2 and proves $t(x) = \langle l(x), r(x) \rangle$ to V through polynomial identity check [30]. Note that instead of sending $\mathbf{l}, \mathbf{r}$ to V directly in Figure 9, P can commit them and prove $t = \langle \mathbf{l}, \mathbf{r} \rangle$ via the inner product argument to reduce the communication costs.

Statement: $(c_{\mathbf{b}}, c_{\mathbf{d}}, h, \mathbf{g}, \mathbf{h})$
Witness: $(\mathbf{b}, r_b, r_d)$

P: Sample $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p^n$, $r_\alpha, r_\beta \xleftarrow{\$} \mathbb{Z}_p$.
Compute $c_\alpha \leftarrow h^{r_\alpha}\mathbf{g}^\alpha, c_\beta \leftarrow h^{r_\beta}\mathbf{h}^\beta$.
P $\Rightarrow$ V: Send c_α, c_β.
V: Sample $y, z \xleftarrow{\$} \mathbb{Z}_p^*$.
V $\Rightarrow$ P: Send y, z.
P: Construct two degree 1 polynomials:

$$l(x) = \mathbf{b} - z \cdot \mathbf{1}^n + \alpha \cdot x,$$
$$r(x) = \mathbf{y}^n \circ (\mathbf{d} + z \cdot \mathbf{1}^n + \beta \cdot x) + z^2 \cdot \mathbf{1}^n.$$

Define $t(x) = \langle l(x), r(x) \rangle = \sum_{i=0}^{2} t_i \cdot x^i$.
Sample $s_1, s_2 \xleftarrow{\$} \mathbb{Z}_p$.
Compute $T_1 \leftarrow g_1^{t_1} h^{s_1}, T_2 \leftarrow g_1^{t_2} h^{s_2}$.
P $\Rightarrow$ V: Send T_1, T_2.
V: $x_0 \xleftarrow{\$} \mathbb{Z}_p^*$.
V $\Rightarrow$ P: Send x_0.
P: Compute $s_x \leftarrow s_1 x_0 + s_2 x_0^2, \mu_1 \leftarrow r_b + r_\alpha x_0, \mu_2 \leftarrow r_d + r_\beta x_0, \mathbf{l} \leftarrow l(x_0), \mathbf{r} \leftarrow r(x_0), t \leftarrow \langle \mathbf{l}, \mathbf{r} \rangle$.
P $\Rightarrow$ V: Send $s_x, \mu_1, \mu_2, t, \mathbf{l}, \mathbf{r}$.
V: For $i \in [n]$, set $h_i' \leftarrow h_i^{y^{-i+1}}$. Return 1 if
$t = \langle \mathbf{l}, \mathbf{r} \rangle \wedge g_1^t h^{s_x} = g_1^{t_0} T_1^{x_0} T_2^{x_0^2} \wedge$
$h^{\mu_1}\mathbf{g}^{\mathbf{l}} = c_{\mathbf{b}} \cdot c_\alpha^{x_0} \cdot g_1^{-z} \wedge$
$h^{\mu_2}\mathbf{h}'^{\mathbf{r}} = c_{\mathbf{d}} \cdot c_\beta^{x_0} \cdot \mathbf{h}'^{z \cdot \mathbf{y}^n + z^2 \cdot \mathbf{1}^n}$.
Return 0 otherwise.

Figure 9: Proof for One-Hot Vector

D.2 Proof of Randomness

In this part, the prover proves that there is a randomness ρ such that $\mathbf{u} = \rho \cdot \mathbf{b}$. Note that by computing $c_{\mathbf{b}}{}^\rho = h^{\rho \cdot r_b}\mathbf{g}^{\rho \cdot \mathbf{b}}$, the prover can obtain a Pedersen commitment on $\mathbf{u}$ using the randomness ρr_b. Thus, the prover commits $\mathbf{u}$ by computing $c_{\mathbf{u}} \leftarrow c_{\mathbf{b}}^\rho$. Let $g^* \leftarrow g_1^\rho$, the prover can reach the goal by proving g^* and $c_{\mathbf{b}}{}^\rho$ share the same exponent ρ. The language $\mathcal{L}_\rho$ is defined as follows:

$$\mathcal{L}_\rho = \left\{ g^*, c_{\mathbf{u}}, c_{\mathbf{b}} \in \mathbb{G}_2 \;\middle|\; \begin{array}{c} \exists \rho \in \mathbb{Z}_p^*, \\ \text{s.t. } g^* = g_1^\rho, c_{\mathbf{u}} = c_{\mathbf{b}}{}^\rho \end{array} \right\}.$$

The instantiation $\Pi_\rho = (\mathsf{Prove}, \mathsf{Vrfy})$ for $\mathcal{L}_\rho$ is shown in Figure 10, and it is actually a variant of the Chaum-Pedersen Σ-protocol [16].

D.3 Inner Product Argument

In this part, the prover proves for each $k \in [4]$, $X_{jk}^* = \mathbf{X}_k^{\mathbf{u}} = \langle \mathbf{u}, \mathbf{X}_k \rangle$. Specifically, the prover needs to provide a valid proof

$\mathsf{Prove}(pp_g, (g^*, c_{\mathbf{u}}, c_{\mathbf{b}}), \rho)$
1: $s \xleftarrow{\$} \mathbb{Z}_p$
2: $I_0 \leftarrow g_1^s, I_1 \leftarrow c_{\mathbf{b}}{}^s$
3: $c \leftarrow \mathsf{H}_{\mathsf{g}}(g_1, g^*, c_{\mathbf{u}}, c_{\mathbf{b}}, I_0, I_1)$
4: $\rho' \leftarrow r + cw$
5: $\pi_w \leftarrow (c, \rho')$
6: **return** π_w

$\mathsf{Vrfy}(pp_g, (g^*, c_{\mathbf{u}}, c_{\mathbf{b}}), \pi_w)$
1: $\pi_w = (c, \rho')$
2: $I_0' \leftarrow g_1^{\rho'} \cdot g^{*-c}, I_1' \leftarrow c_{\mathbf{b}}{}^{\rho'} \cdot c_{\mathbf{u}}{}^{-c}$
3: **if** $\mathsf{H}_{\mathsf{g}}(g_1, g^*, c_{\mathbf{u}}, c_{\mathbf{b}}, I_0', I_1') = c$ **then**
4: **return** 1
5: **return** 0

Figure 10: Proof of Randomness

for the following language $\mathcal{L}_x$:

$$\mathcal{L}_x = \left\{ \begin{array}{c} X_{jk}^*, c_{\mathbf{u}}, h \in \mathbb{G}_2, \\ \mathbf{g}, \mathbf{X}_k \in \mathbb{G}_2^n \end{array} \;\middle|\; \begin{array}{c} \exists \mathbf{u} \in \mathbb{Z}_p^n, \rho' \in \mathbb{Z}_p, \text{s.t.} \\ X_{jk}^* = \langle \mathbf{u}, \mathbf{X}_k \rangle, \\ c_{\mathbf{u}} = h^{\rho'}\mathbf{g}^{\mathbf{u}} \end{array} \right\},$$

where $\rho' = \rho r_b$ and ρ and r_b are known to the prover.

We design a proof system Π_{x} for $\mathcal{L}_x$ by an inner product argument based on Bulletproofs [11] for $\log n$ rounds to prove that $X_{jk}^* = \langle \mathbf{u}, \mathbf{X}_k \rangle$.

However, the original protocol in Bulletproofs [11] cannot be applied in Π_{x}. At the beginning of the original one, the prover separately divides $\mathbf{u}$ and $\mathbf{X}_k$ into two parts, such that $\mathbf{u} = \mathbf{u}_L | \mathbf{u}_R$, $\mathbf{X}_k = \mathbf{X}_L | \mathbf{X}_R$. Let $L_x \leftarrow \langle \mathbf{u}_L, \mathbf{X}_R \rangle$, $R_x \leftarrow \langle \mathbf{u}_R, \mathbf{X}_L \rangle$, $L_g \leftarrow \langle \mathbf{u}_L, \mathbf{g}_R \rangle$, $R_g \leftarrow \langle \mathbf{u}_R, \mathbf{g}_L \rangle$. The prover sends the inner product values to the verifier during the protocol. Since $\mathbf{u} = \rho \cdot \mathbf{b}$ and $\mathbf{b}$ is a one-hot vector, this values expose the non-zero index in $\mathbf{u}$, and the verifier can detect in which half the non-zero index lies. After $\log n$ rounds, the verifier can locate the specific index.

To ensure zero-knowledge, L_x, R_x, L_g and R_g cannot be sent to the verifier directly. The prover use $h^{r_1}, h^{r_2}, h^{r_1'}, h^{r_2'}$ for randomly chosen r_1, r_2, r_1', r_2' to blind these values, and send $h^{r_1} \cdot L_x, h^{r_2} \cdot R_x, h^{r_1'} \cdot L_g, h^{r_2'} \cdot R_g$ to the verifier. In the following $\log n - 1$ rounds, the prover can blind the related values in the same way.

At the very end of the protocol, denote the final values by $c_u = h^{r_u} g^u$, $X_f = h^{r_x} X^u$ for the single-element vector u, using the final scalar X and g. The prover needs to prove that the c_u and X_f share the same scalar u in zero-knowledge, which is a variant of the Chaum-Pedersen Σ-protocol [16]. Note that r_u and r_x can be computed from all randomness by P, and X and g can be generated from public values. With the above improvements, we finish our proof system Π_{x} for $\mathcal{L}_x$.