Revisiting Accountability in Accountable Tracing Signatures

Rei Naito[1], Kyoichi Asano[2], and Keita Emura*[1,2]

[1.]Kanazawa University, Japan.
[2.]AIST, Japan.

November 1, 2026

Abstract

Accountable tracing signatures (ATS), introduced by Kohlweiss and Miers (PoPETs 2015), classify each group member (user) as either traceable or non-traceable. A traceable user corresponds to one in conventional group signature schemes whose identity can be revealed by the opener from a signature, whereas a non-traceable user cannot be identified from any signature they produce. The opener, who issues a witness, can certify a user's status via the Account algorithm. In this paper, we formalize *accountability soundness*, which ensures that no two witnesses can simultaneously certify a user as both traceable and non-traceable. We show that the Kohlweiss-Miers ATS scheme admits conflicting witnesses that violate this property. Since accountability is a fundamental aspect of ATS, as reflected in its name, such conflicting witnesses undermine the intended guarantees of the primitive. To address this issue, we propose a patched version of the Kohlweiss-Miers ATS scheme that provides accountability soundness. We also discuss the applicability of our attack to the lattice-based ATS scheme (Ling et al., CT-RSA 2019) and the isogeny-based ATS scheme (Shaw-Dutta, Indocrypt 2021), together with preliminary investigations of possible countermeasures. We emphasize that all our attacks fall outside the original security models.

1 Introduction

1.1 Group Signatures and Accountability

Group signatures (GS) [15] are well known as privacy-enhanced signatures. A signer can generate a signature on behalf of a group, and a verifier can verify the signature anonymously; that is, the verifier can confirm that the signer is a group member without revealing the identity of the individual signer. A special authority, called the opener (or group manager), can identify the signer using its secret key. For example, the opener identifies the winner of an anonymous auction or a signer exhibiting misbehavior, while non-target signers remain anonymous. At first glance, GS appears to provide a well-balanced combination of anonymity and traceability. However, there remains an issue regarding the authority of the opener: since the opener can identify the signer without restriction, it can, in principle, identify the signer of any signature. Therefore, how to appropriately restrict the authority of the opener while preserving both anonymity and traceability has been widely studied.

*Corresponding Author: k-emura@se.kanazawa-u.ac.jp

One approach to reducing the authority of the opener is to introduce accountability. Graf et al. [22] define accountability as a security notion that *allows for obtaining undeniable cryptographic proof of misbehavior, thereby incentivizing honest behavior*. Because the notion of accountability varies depending on the party to whom an actor is held accountable, several different formulations are possible. Consequently, unlike anonymity or traceability, accountability itself is not usually formalized as a security property. Various forms of accountability have been studied in the context of group and threshold signature schemes, each addressing different aspects of authority and the prevention of misuse. Accountable ring signatures (ARS) [12], which imply fully dynamic group signatures (FDGS) [8,12], introduce accountability by enabling public verification of tracing results, thereby preventing the opener from falsely attributing a signature to a user other than the actual signer. Furthermore, a valid tracing proof provides public evidence that the identified signer is indeed the actual signer, thereby contributing to signer accountability as well. Similarly, group signatures with deniability have been proposed [23,30], which introduce accountability from another perspective by ensuring that a given user is not the signer of a signature. In contrast, accountable tracing signatures (ATS) [26, 29, 34][1] focus on the accountability of the opener with respect to users' traceability status. They provide a mechanism to certify whether a user is traceable or non-traceable, preventing the opener from arbitrarily misrepresenting this classification. Group signatures with message-dependent opening (GS-MDO) [27, 33][2] achieve another form of accountability by restricting the conditions under which opening is permitted. By introducing two authorities, the opener and the admitter, GS-MDO ensures that tracing can only be performed under legitimate conditions (i.e., the admitter issues a message-dependent token, and the opener can then identify signers of signatures on that message), thereby preventing arbitrary use of the opening capability. From this perspective, traceable signatures [25,28,35] can be seen as providing a form of user-dependent linking, where a trapdoor enables one to determine whether two or more signatures were produced by the same signer. Set pre-constrained group signatures (SPC) [4] provide a different perspective on accountability by restricting the scope of traceability to a predefined set of messages. In SPC, even a malicious opener cannot trace arbitrary signatures; tracing is only possible when the signed message belongs to a pre-specified set (e.g., illegal content). This ensures that the tracing capability is inherently constrained and cannot be abused beyond its intended purpose. Claimable ring signatures [32] provide a form of signer accountability in which the signer of a signature can claim that they generated the signature. Threshold signatures with private accountability (TAPS) [9] address accountability in the context of collaborative signing. While hiding the identities of participating signers during normal operation, TAPS enables the revelation and verification of the actual subset of signers involved in generating a signature when necessary, ensuring that no party can falsely claim or deny participation. To sum up, providing accountability, which can be enforced at different levels and from various perspectives, is a widely researched area.

1.2 Accountable Tracing Signatures

In this paper, we focus on ATS, proposed by Kohlweiss and Miers [26]. As mentioned in the previous subsection, each group member (user) is categorized as either traceable or non-traceable. A traceable user corresponds to a user in conventional GS schemes whose identity can be traced

[1] In this paper, ATS stands for accountable tracing signature. As a remark, Boneh et al. [9–11] use ATS to denote accountable threshold signatures, in which a threshold signature reveals the identities of the participants involved in generating the signature.

[2] As another form of accountability in GS-MDO, admitter accountability has been studied [36]. Through public verifiability of tokens, the admitter can demonstrate that a token was honestly generated. For example, the GS-MDO scheme by Ohara et al. [31] provides admitter accountability.

by the opener from a signature generated by that user. In contrast, a non-traceable user cannot be identified as the signer by the opener from any signature produced by that user. Although users do not know whether they are traceable or non-traceable, the opener, who issues a witness w^{escrw}, can prove a user's status via the $\mathsf{Account}$ algorithm. More precisely, each user receives a certificate cert when joining the group, and

$$\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w^{\mathsf{escrw}}, \mathsf{tr}) = 1$$

indicates that the user associated with cert is traceable if $\mathsf{tr} = 1$, and non-traceable if $\mathsf{tr} = 0$, where gpk denotes the group public key. Kohlweiss and Miers [26] proposed a pairing-based ATS scheme. Later, Ling et al. [29] proposed a lattice-based ATS scheme, and Shaw and Dutta [34] proposed an isogeny-based ATS scheme.

1.3 Our Motivation

As mentioned above, the goal of accountability in ATS is to ensure the legitimacy of traceability status. In particular, it is required that the opener cannot forge or alter the status of group members (users) once it has been determined. For example, if there exist two witnesses w_0^{escrw} and w_1^{escrw} such that both certify that a user associated with cert is simultaneously traceable and non-traceable, i.e.,

$$\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_1^{\mathsf{escrw}}, 1) = 1,$$

then the accountability of the opener is violated. However, the existing security notions formalized in [29] (correctness, anonymity under tracing, anonymity with accountability, traceability, non-frameability, and trace-obliviousness) do not include a definition that guarantees that no adversary can produce such conflicting witnesses. Therefore, the goal of this paper is as follows:

- To formalize *accountability soundness*, which guarantees that no two witnesses can certify that a user is both traceable and non-traceable.
- To investigate whether existing ATS schemes satisfy accountability soundness.

1.4 Our Contribution

In this paper, we demonstrate that there exist conflicting witnesses in the Kohlweiss-Miers ATS scheme [26]. Given that the term "accountable" is included in the name ATS, it is clear that guaranteeing accountability constitutes a fundamental aspect of ATS. The existence of such conflicting witnesses undermines this foundation and can therefore be regarded as an issue that must be addressed. By considering the situation that an adversary can produce conflicting witnesses, we formalize accountable soundness, and propose a patched version of the Kohlweiss-Miers ATS scheme that provides accountable soundness.

Technical Overview. The core building block of ATS is key-oblivious encryption (KOE) and Kohlweiss and Miers proposed an ElGamal-type KOE scheme. Informally, KOE is a kind of updatable public key encryption (UPKE) [1–3, 20, 24]. Unlike UPKE, no time information is defined in KOE. KOE and their idea of the KOE-based ATS construction are informally explained as follows.

- Generate a public key and secret key pair for $\mathsf{tr} = 0$ (non-traceable) $(\mathsf{pk}_0, \mathsf{sk}_0)$ and erase sk_0 (this key erasure is proceed in a trusted setup assumption), and generate a public key and secret key pair for $\mathsf{tr} = 1$ (traceable) $(\mathsf{pk}_1, \mathsf{sk}_1)$.

- For a user whose status is $\mathsf{tr} \in \{0,1\}$, randomize $\mathsf{pk}_{\mathsf{tr}}$ using a randomness r where $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_{\mathsf{tr}}; r)$ (alternatively, it is denoted as $(\mathsf{pk}', r) \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_{\mathsf{tr}})$).
  - For all plaintexts M, $\mathsf{KOE.Dec}(\mathsf{sk}_{\mathsf{tr}}, \mathsf{KOE.Enc}(\mathsf{pk}', M)) = M$ holds, i.e., the secret key before randomization can decrypt a ciphertext generated by the randomized public key.
  - No information of tr is revealed from pk'.
  - If a user is traceable, then cert contains $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_1; r)$. A user-related value is encrypted by pk' in the signing algorithm, and the opener who has sk_1 can decrypt the ciphertext and identity the user. On the other hand, if a user is non-traceable, then cert contains $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_0; r)$. Since sk_0 has been erased, nobody can identify the user.
- Set $w^{\mathsf{escrw}} = r$ as the witness. The $\mathsf{Account}$ algorithm outputs 1 if $\mathsf{pk}' = \mathsf{KOE.KeyRand}(\mathsf{pk}_{\mathsf{tr}}; w^{\mathsf{escrw}})$.

At a high level, our attack can be interpreted as simply "changing the order" of key randomization. Originally, pk' contained in cert is generated by the $\mathsf{KeyRand}$ algorithm using either pk_0 or pk_1. We generate pk' from pk_1 and then generate pk_0 from pk' as follows.

- Generate $(\mathsf{pk}_1, \mathsf{sk}_1)$ as usual. Generate $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_1; r_1)$ and $\mathsf{pk}_0 \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}'; r_0)$. Set $w_0^{\mathsf{escrw}} = r_0^{-1}$ and $w_1^{\mathsf{escrw}} = r_1$. Then, since $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_0; w_0^{\mathsf{escrw}})$ and $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_1; w_1^{\mathsf{escrw}})$ simultaneously hold, $\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_1^{\mathsf{escrw}}, 1) = 1$ holds.
  - The crucial point is the user can recognize that "I am a non-traceable user" via w_0^{escrw}, but the opener can identify the user using sk_1.
  - Our attack is reminiscent of equivocal commitments (e.g., [13]), which allow the committer to generate a commitment to any value using a secret key. Unlike equivocal commitments, however, our attack does not require any secret key.

We emphasize that the setting employed in the Kohlweiss-Miers ATS scheme has prevented to trace a non-traceable user by assuming that $(\mathsf{pk}_0, \mathsf{sk}_0)$ is generated and sk_0 is erased in the trusted setup algorithm, and later (i.e., after pk_0 is fixed) the opener generates $(\mathsf{pk}_1, \mathsf{sk}_1)$. Nevertheless, this key generation order restriction is not sufficient to prevent our attack.

- After pk_0 is fixed, generate $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_0; r_0)$ and $\mathsf{pk}_1 \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}'; r_1)$.
- Then, for $w_0^{\mathsf{escrw}} = r_0$ and $w_1^{\mathsf{escrw}} = r_1^{-1}$, $\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_1^{\mathsf{escrw}}, 1) = 1$ holds.
- Although the adversary does not know the corresponding secret key and cannot identify a traceable user, the attack shows that conflicting witnesses exist.

Again, given that the term "accountable" is included in the name ATS, it is clear that guaranteeing accountability constitutes a fundamental aspect of ATS. The existence of such conflicting witnesses undermines this foundation and can therefore be regarded as an issue that must be addressed. Later, we will provide a patched version of the Kohlweiss-Miers ATS scheme that provides accountability soundness. Briefly, knowledge of sk_1 is proven in a zero-knowledge manner, and an extractor can be used to recover sk_1, which suffices to break the plaintext indistinguishability under key randomization security of the KOE scheme.

Other ATS Schemes. The Kohlweiss-Miers ATS scheme is based on pairings and is therefore vulnerable to Shor's algorithm. From the perspective of post-quantum cryptography (PQC), it is highly desirable to explore ATS schemes that are secure against quantum adversaries. Since the lattice-based ATS scheme [29] and the isogeny-based ATS scheme [34] also employ KOE as a building block in a similar manner, one may expect that they likewise do not provide accountability soundness. We discuss the applicability of our attack to the lattice-based ATS scheme and the isogeny-based ATS scheme, and present preliminary investigations of possible countermeasures. We emphasize that all our attacks fall outside the original security models.

- For the lattice-based ATS scheme [29], we show that our attack can be applied in Section 6. Specifically, we demonstrate that conflicting witnesses can be constructed from the key-randomization procedure underlying the lattice-based KOE scheme. We further discuss possible countermeasures based on witness norm-size checking and rejection sampling, and examine the conditions under which such approaches may be effective. A rigorous characterization of these conditions and the design of a secure modified construction are left for future work.
- For the isogeny-based ATS scheme [34], we show that our attack can be applied in Section 7. If one follows the same approach as in the patched Kohlweiss-Miers ATS scheme, it is currently unclear how to recover the secret key using the strategy described above. We expect that an isogeny-based proof-of-knowledge technique, such as those in [5, 16–19], could be applicable in this setting. Designing a patched scheme is left for future work.

2 Preliminaries

In this section, we introduce KOE and ATS. We mainly introduce the definitions given in [29].

2.1 Key-Oblivious Encryption (KOE)

A KOE scheme $\mathsf{KOE} = (\mathsf{KOE.Setup}, \mathsf{KOE.KeyGen}, \mathsf{KOE.KeyRand}, \mathsf{KOE.Enc}, \mathsf{KOE.Dec})$ is defined as follows.

KOE.Setup: The setup algorithm takes a security parameter $\lambda \in \mathbb{N}$ as input, and outputs a common parameter $\mathsf{KOE.pp}$. Unless explicitly specified otherwise, all algorithms implicitly take $\mathsf{KOE.pp}$ as input.

KOE.KeyGen: The key generation algorithm takes $\mathsf{KOE.pp}$ as input, and outputs a pair of a public key and a secret key $(\mathsf{pk}, \mathsf{sk})$.

KOE.KeyRand: The key randomization algorithm takes pk as input, and outputs a public key pk'. If we explicitly denote the randomness r used in the algorithm, we write $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}; r)$. In contexts where the randomness r is made explicit, we write $(\mathsf{pk}', r) \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk})$. Otherwise, we denote it $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk})$.

KOE.Enc: The encryption algorithm takes pk and a plaintext M as input, and outputs a ciphertext ct.

KOE.Dec: The decryption algorithm takes sk and ct as input, and outputs M.

Correctness. For all $\lambda \in \mathbb{N}$, $\mathsf{KOE.pp} \leftarrow \mathsf{KOE.Setup}(1^\lambda)$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$, $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk})$, and M, it is required that $\mathsf{KOE.Dec}(\mathsf{sk}, \mathsf{KOE.Enc}(\mathsf{pk}', M)) = M$ holds.

Plaintext indistinguishability under key randomization is defined as follows. Let $\mathcal{A}$ be an adversary and $\mathcal{C}$ be the challenger.

Definition 1 (Plaintext Indistinguishability under Key Randomization)**.** *It guarantees that standard IND-CPA (indistinguishability against chosen plaitext attacks) security holds even when a ciphertext is generated using a randomized public key and the randomness used in the* $\mathsf{KOE.KeyRand}$ *algorithm is known to* $\mathcal{A}$*.* $\mathcal{C}$ *runs* $\mathsf{pp} \leftarrow \mathsf{KOE.Setup}(1^\lambda)$ *and* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$*, and sends* pk *to* $\mathcal{A}$*.* $\mathcal{A}$ *declares the challenge plaintexts* M_0^* *and* M_1^* *and a public key* pk^* *and a randomness* r^**.* $\mathcal{C}$ *returns* $\perp$ *if* $\mathsf{pk}^* \neq \mathsf{KOE.KeyRand}(\mathsf{pk}; r^*)$*.* $\mathcal{C}$ *flips a coin* $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ *and computes the challenge ciphertext* $\mathsf{ct}^* \leftarrow \mathsf{KOE.Enc}(\mathsf{pk}^*, M^*_{\mathsf{coin}})$*.* $\mathcal{A}$ *outputs* $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$*. We say that* KOE *provides plaintext indistinguishability under key randomization if* $\mathsf{Adv}^{\mathsf{INDr}}_{\mathsf{KOE},\mathcal{A}}(\lambda) := |\Pr[\mathsf{coin} = \mathsf{coin}'] - 1/2|$ *is negligible in* λ *for all probabilistic polynomial-time (PPT) adversaries* $\mathcal{A}$*.*

Other security notions are also defined. Key randomizability guarantees that no information of the source public key is revealed from the randomized public key. Key privacy under key randomization guarantees that no information about which randomized public key is used for encryption is revealed, even when the randomness used in the $\mathsf{KOE.KeyRand}$ algorithm is known to $\mathcal{A}$. See [29] for formal definitions.

ElGamal-Type KOE Scheme [26]. Under the decisional Diffie-Hellman (DDH) assumption, the KOE scheme provides key randomizability, plaintext indistinguishability under key randomization, and key privacy under key randomization.

$\mathsf{KOE.Setup}(1^\lambda)$**:** Let $\mathbb{G}$ be a group of prime order p, and let $g \in \mathbb{G}$ be a generator that provides λ-bit security. Output $\mathsf{KOE.pp} = (g, \mathbb{G}, p)$.

$\mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$**:** Sample $r, \gamma \xleftarrow{\$} \mathbb{Z}_p$ and return $\mathsf{pk} = (g^r, (g^r)^\gamma)$ and $\mathsf{sk} = (\mathsf{pk}, \gamma)$.

$\mathsf{KOE.KeyRand}(\mathsf{pk})$**:** Parse $\mathsf{pk} = (A, B)$. Sample $r \xleftarrow{\$} \mathbb{Z}_p$ (if r is not provided as an input to the algorithm), and output $\mathsf{pk}' = (A^r, B^r)$ (and r, if required).

$\mathsf{KOE.Enc}(\mathsf{pk}, M)$**:** Parse $\mathsf{pk} = (A, B)$. Sample $s \xleftarrow{\$} \mathbb{Z}_p$ and return $\mathsf{ct} = (A^s, B^s M)$.

$\mathsf{KOE.Dec}(\mathsf{sk}, \mathsf{ct})$**:** Parse $\mathsf{sk} = (\mathsf{pk}, \gamma)$. Compute $\mathsf{ct} = (C, D)$, and return D/C^γ.

2.2 Accountable Tracing Signatures (ATS)

An ATS scheme $\mathsf{ATS} = (\mathsf{ATS.Setup}, \mathsf{GKeyGen}, \mathsf{UKeyGen}, \mathsf{Enroll}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Open}, \mathsf{Judge}, \mathsf{Account})$ is defined as follows.

$\mathsf{ATS.Setup}$**:** The setup algorithm takes a security parameter $\lambda \in \mathbb{N}$ and outputs a common parameter $\mathsf{ATS.pp}$. Unless explicitly specified otherwise, all algorithms implicitly take $\mathsf{ATS.pp}$ as input.

$\mathsf{GKeyGen}$**:** The group key generation algorithm takes $\mathsf{ATS.pp}$ as input and outputs the group public key gpk, the issuing key ik, and the opening key ok. We explicitly denote by $\mathsf{gpk.pp}$ the common parameters used in generating gpk.

UKeyGen: The user key generation algorithm takes $\mathsf{ATS.pp}$ as input and outputs a user key pair $(\mathsf{upk}, \mathsf{usk})$.

Enroll: The enrollment algorithm takes gpk, ik, upk, and $\mathsf{tr} \in \{0,1\}$ as input. Here, $\mathsf{tr} = 0$ (resp. $\mathsf{tr} = 1$) indicates that the user is non-traceable (resp. traceable). The algorithm generates a certificate cert and a witness w^{escrw}, stores $(\mathsf{upk}, \mathsf{cert}, w^{\mathsf{escrw}}, \mathsf{tr})$, and outputs cert, which is assumed to be sent to the user associated with upk.

Sign: The signing algorithm takes gpk, cert, usk, and a message m to be signed, and outputs a signature Σ.

Verify: The verification algorithm takes gpk, m, and Σ as input and outputs 0 or 1.

Open: The opening algorithm takes gpk, ok, m, and Σ as input, and outputs a user public key upk and a proof Π_{Open}. In case of $\mathsf{upk} = \perp$, $\Pi_{\mathsf{Open}} = \perp$.

Judge: The judgment algorithm takes gpk, m, Σ, upk, and Π_{Open} as input, and outputs 0 or 1.

Account: The accountability algorithm takes gpk, cert, w^{escrw}, and tr, and outputs 0 or 1.

Correctness guarantees that signatures generated honestly using honestly generated keys are always valid (i.e., the Verify algorithm outputs 1). If a signature is generated using a certificate cert with $\mathsf{tr} = 1$, then the Open algorithm correctly identifies the signer, and the corresponding opening proof is accepted by the Judge algorithm. If a signature is generated using a certificate cert with $\mathsf{tr} = 0$, then the Open algorithm always outputs $(\mathsf{upk}, \Pi_{\mathsf{Open}}) = (\perp, \perp)$. The $\mathsf{Account}$ algorithm outputs 1 if cert and w^{escrw} are honestly generated with the corresponding value of tr.

We informally introduce the security notions as follows, see [29] for the formal definitions. The purpose of this introduction is to clarify that these notions do not imply accountability soundness (where the adversary outputs conflicting witnesses), which we define later.

Anonymity under Tracing: This is basically the same as the standard full CCA anonymity for group signatures [6]. An adversary $\mathcal{A}$ takes $(\mathsf{gpk}, \mathsf{ik})$ and declares $(\mathsf{cert}_0, \mathsf{cert}_1, \mathsf{usk}_0, \mathsf{usk}_1, M, w_0^{\mathsf{escrw}}, w_1^{\mathsf{escrw}}, \mathsf{tr} = 1)$, i.e., $\mathcal{A}$ is allowed to know all the keys required to generate signatures. Here, two challenge users are required to be traceable, i.e., $\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}_0, w_0^{\mathsf{escrw}}, 1) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}_1, w_1^{\mathsf{escrw}}, 1) = 1$. The challenger $\mathcal{C}$ returns the challenge signature on M using $(\mathsf{cert}_{\mathsf{coin}}, \mathsf{usk}_{\mathsf{coin}})$ where $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ is the challenge bit. $\mathcal{A}$ is allowed to access the Open algorithm. $\mathcal{A}$ outputs $\mathsf{coin}' \in \{0,1\}$ and wins if $\mathsf{coin} = \mathsf{coin}'$.

Anonymity with Accountability: An adversary $\mathcal{A}$ takes $\mathsf{ATS.pp}$ as input and is allowed to generate all values including ok. One restriction is that $\mathsf{gpk.pp} = \mathsf{ATS.pp}$ must be held (this relates the trusted setup). $\mathcal{A}$ declares $(\mathsf{cert}_0, \mathsf{cert}_1, \mathsf{usk}_0, \mathsf{usk}_1, M, w_0^{\mathsf{escrw}}, w_1^{\mathsf{escrw}}, \mathsf{tr} = 0)$, and two challenge users are required to be non-traceable, i.e., $\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}_0, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}_1, w_1^{\mathsf{escrw}}, 0) = 1$. The challenger $\mathcal{C}$ returns the challenge signature on M using $(\mathsf{cert}_{\mathsf{coin}}, \mathsf{usk}_{\mathsf{coin}})$ where $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ is the challenge bit. $\mathcal{A}$ outputs $\mathsf{coin}' \in \{0,1\}$ and wins if $\mathsf{coin} = \mathsf{coin}'$.

- At first glance, if $\mathcal{A}$ can produce conflicting witnesses $(w_0^{\mathsf{escrw}}, w_1^{\mathsf{escrw}})$, e.g., $\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}_0, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}_0, w_1^{\mathsf{escrw}}, 1) = 1$, then it may seem that $\mathcal{A}$ can break anonymity with accountability. This is because $\mathcal{A}$ generates cert_0 with $\mathsf{tr} = 1$ (traceable) but presents w_0^{escrw} claiming that the corresponding user is non-traceable. However, the

existence of conflicting witnesses does not imply that the user is actually traceable, in particular, it is not guaranteed that $\mathcal{A}$ can trace upk_0 via the Open algorithm. In fact, in our attack presented in Section 4, an adversary that can produce conflicting witnesses still cannot trace the target user. This demonstrates that anonymity with accountability does not imply accountability soundness.

Traceability: An adversary $\mathcal{A}$ takes gpk and declares (m, Σ) where $\mathsf{Verify}(\mathsf{gpk}, m, \Sigma) = 1$ but the Open algorithm outputs $(\mathsf{upk}, \Pi_{\mathsf{Open}}) = (\bot, \bot)$, or $\mathsf{upk} \neq \bot$ and $\Pi_{\mathsf{Open}} \neq \bot$ are rejected by the Judge algorithm. Unlike traceability for group signature [7], $\mathcal{A}$ does not obtain ok. $\mathcal{A}$ is allowed to access the Enroll oracle and receives cert with $\mathsf{tr} = 1$ (traceable). This ensures that the adversary does not hold both certificate and user secret key for a non-traceable user.

Non-Frameability: This is basically the same as non-frameability for group signatures [7]. An adversary $\mathcal{A}$ takes $\mathsf{ATS.pp}$ as input and is allowed to generate all values including ok. One restriction is that $\mathsf{gpk.pp} = \mathsf{ATS.pp}$ must be held (this relates the trusted setup). $\mathcal{A}$ outputs $(m, \Sigma, \mathsf{upk}, \Pi_{\mathsf{Open}})$ where $\mathsf{Verify}(\mathsf{gpk}, m, \Sigma) = 1$ and $\mathsf{Judge}(\mathsf{gpk}, m, \Sigma, \mathsf{upk}, \Pi_{\mathsf{Open}}) = 1$ hold, and the user who has upk is honest.

Trace-Obliviousness: This guarantees that each user cannot determine whether they are being traced or not. The adversary takes gpk as input and received cert which is generated by $\mathsf{tr} = \mathsf{coin} \xleftarrow{\$} \{0, 1\}$. $\mathcal{A}$ outputs $\mathsf{coin}' \in \{0, 1\}$ and wins if $\mathsf{coin} = \mathsf{coin}'$.

Blueprint Construction. In KOE-based ATS constructions [26, 29, 34], the $\mathsf{ATS.Setup}$ algorithm runs $\mathsf{KOE.pp} \leftarrow \mathsf{KOE.Setup}(1^\lambda)$ and $(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$, erases sk_0, and outputs $\mathsf{ATS.pp} = (\mathsf{pk}_0, \mathsf{KOE.pp})$. The $\mathsf{GKeyGen}$ algorithm, which takes $\mathsf{ATS.pp}$ as input, runs $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$ and outputs $\mathsf{gpk} = (\mathsf{ATS.pp}, \mathsf{pk}_1)$ and $\mathsf{ok} = \mathsf{sk}_1$ (other values are omitted here). Since $\mathsf{gpk.pp} = \mathsf{ATS.pp}$ is required, the public key pk_0 contained in gpk is identical to the one generated in the $\mathsf{ATS.Setup}$ algorithm. Due to this trusted setup, it is guaranteed that no party learns sk_0. The remaining components follow the so-called sign-then-encrypt-then-prove (SEP) paradigm [6, 7]. Here, ok is set to sk_1 of the KOE scheme. ik is the signing key of a signature scheme, while the corresponding verification key is included in gpk. $(\mathsf{upk}, \mathsf{usk})$ denotes a key pair of the signature scheme. To join the group, pk' is generated from $\mathsf{pk}_{\mathsf{tr}}$ via the $\mathsf{KOE.KeyRand}$ algorithm with randomness w^{escrw} where $\mathsf{tr} \in \{0, 1\}$. The certificate cert contains pk', upk, and a signature on $(\mathsf{pk}', \mathsf{upk})$ generated by ik. In the Sign algorithm, the user first generates a signature on the message using usk, and then encrypts both the signature and upk under pk'. The user further generates a non-interactive zero-knowledge (NIZK) proof showing that the encrypted signature is valid under upk, and that upk is signed by ik, without revealing upk. The final signature consists of the ciphertext and the NIZK proof. If pk' is derived from pk_1, then the opener can identify the user by decrypting the ciphertext contained in the signature using $\mathsf{ok} = \mathsf{sk}_1$. The $\mathsf{Account}$ algorithm takes cert, containing pk', w^{escrw}, and tr as input and outputs 1 if $\mathsf{pk}' = \mathsf{KOE.KeyRand}(\mathsf{pk}_{\mathsf{tr}}; w^{\mathsf{escrw}})$.

3 Accountability Soundness

In this section, we define accountability soundness, which is a reasonable notion that should be achieved by KOE-based ATS schemes, as it captures realistic threats while allowing for the trusted setup inherently required in such constructions.

Definition 2 (Accountability Soundness)**.** *Let $\mathcal{A}$ be an adversary and $\mathcal{C}$ be the challenger. $\mathcal{C}$ runs* $\mathsf{ATS.pp} \leftarrow \mathsf{ATS.Setup}(1^\lambda)$ *and gives* $\mathsf{ATS.pp}$ *to $\mathcal{A}$. $\mathcal{A}$ outputs* $(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, w_1^{\mathsf{escrw}})$*. $\mathcal{A}$ wins if*

$\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_1^{\mathsf{escrw}}, 1) = 1$ *hold and* $\mathsf{gpk.pp} = \mathsf{ATS.pp}$. *We say that* ATS *provides accountability soundness if the advantage* $\mathsf{Adv}^{\mathsf{AS}}_{\mathsf{ATS},\mathcal{A}}(\lambda) := \Pr[\mathcal{A}\ \textit{wins}]$ *is negligible in* λ *for all PPT adversaries* $\mathcal{A}$.

We can further strengthen accountability soundness, where $\mathcal{A}$ is allowed to produce ATS.pp. Concretely, $\mathcal{A}$ takes the security parameter $\lambda \in \mathbb{N}$ as input, and outputs $(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, w_1^{\mathsf{escrw}})$. $\mathcal{A}$ wins if $\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_1^{\mathsf{escrw}}, 1) = 1$ hold. Since it does not match the situation adopted in all KOE-based ATS schemes, we do not consider this stronger notion in this paper.

4 Security Analysis on ElGamal-type KOE-based ATS

In this section, we show that the Kohlweiss-Miers ATS scheme [26] does not provide accountable soundness, and later generalize the analysis. We emphasize that Kohlweiss and Miers did not define accountability soundness and thus our analysis is outside of their security models.

As mentioned in the technical overview section, our attack requires at least two times randomization processes: for pk_0, we generate $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_0; r_0)$ and $\mathsf{pk}_1 \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}'; r_1)$. In the ElGamal-type KOE scheme, for $\mathsf{pk}_0 = (A, B)$, $\mathsf{pk}' = (A^{r_0}, B^{r_0}) := (A', B')$ and $\mathsf{pk}_1 = (A'^{r_1}, B'^{r_1}) = (A^{r_0 r_1}, B^{r_0 r_1})$. Due to the algebraic structure of $\mathbb{G}$, it can be regarded as $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_1, r_1^{-1})$ where r_1^{-1} is computed modulo p (i.e., $r_1 r_1^{-1} \bmod p = 1$).

Theorem 1. *The Kohlweiss-Miers ATS scheme does not provide accountability soundness.*

Proof. Let the challenger $\mathcal{C}$ issues ATS.pp containing $\mathsf{pk}_0 = (A, B)$. We construct an adversary $\mathcal{A}$ that breaks accountability soundness as follows. $\mathcal{A}$ samples $r_0, r_1 \xleftarrow{\$} \mathbb{Z}_p$ and computes $\mathsf{pk}' = (A^{r_0}, B^{r_0}) := (A', B')$ and $\mathsf{pk}_1 = (A'^{r_1}, B'^{r_1}) = (A^{r_0 r_1}, B^{r_0 r_1})$. Set $w_0^{\mathsf{escrw}} = r_0$ and $w_1^{\mathsf{escrw}} = r_1^{-1} \bmod p$. $\mathcal{A}$ prepares all other values. Then, $\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_0^{\mathsf{escrw}}, 0) = \mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w_1^{\mathsf{escrw}}, 1) = 1$ holds since $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_0; w_0^{\mathsf{escrw}})$ and $\mathsf{pk}' \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}_1; w_1^{\mathsf{escrw}})$ hold. Thus, $\mathcal{A}$ breaks accountability soundness.

This can be simply extended for multiple users where for $i \in [n+1]$ where $n \in \mathbb{N}$ is the number of group members (users), set $\mathsf{pk}^{(i)} \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}^{(i-1)}; r_i)$ and set $\mathsf{pk}^{(0)} = \mathsf{pk}_0$ and $\mathsf{pk}_1 \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}^{(n)}; r_{n+1})$.

5 Patched Scheme

First, we explore what happens if $\mathcal{A}$ can produce conflicting witnesses that breaks accountability soundness. Let $\mathsf{pk}_0 = (A_0, B_0) = (g^{r_0}, (g^{r_0})^{\gamma_0})$, $\mathsf{sk}_0 = \gamma_0$, $\mathsf{pk}_1 = (A_1, B_1) = (g^{r_1}, (g^{r_1})^{\gamma_1})$ and $\mathsf{sk}_1 = \gamma_1$. If $(\mathsf{pk}_1, \mathsf{sk}_1)$ is generated by KOE.KeyGen algorithm by following the protocol description, then γ_0 and γ_1 are independently chosen. Thus, revealing sk_1 does not affect key erasure where sk_0 is erased in the ATS.Setup algorithm. Assume that $\mathcal{A}$ outputs conflicting witnesses w_0^{escrw} and w_1^{escrw} where for a public key pk', $\mathsf{pk}' = \mathsf{KOE.KeyRand}(\mathsf{pk}_0; w_0^{\mathsf{escrw}})$ and $\mathsf{pk}' = \mathsf{KOE.KeyRand}(\mathsf{pk}_1; w_1^{\mathsf{escrw}})$ simultaneously hold. This implies that $\mathsf{pk}' = (A_0^{w_0^{\mathsf{escrw}}}, B_0^{w_0^{\mathsf{escrw}}}) = (A_1^{w_1^{\mathsf{escrw}}}, B_1^{w_1^{\mathsf{escrw}}})$, $r_0 w_0^{\mathsf{escrw}} = r_1 w_1^{\mathsf{escrw}}$, and $r_0 \gamma_0 w_0^{\mathsf{escrw}} = r_1 \gamma_1 w_1^{\mathsf{escrw}}$ hold. This implies $\mathsf{sk}_0 = \mathsf{sk}_1$ since $\gamma_0 = r_1 \gamma_1 w_1^{\mathsf{escrw}} / r_0 w_0^{\mathsf{escrw}} = \gamma_1$ holds. To sum up, if $\mathcal{A}$, that produces gpk containing pk_1, produces conflicting witnesses that break accountability soundness, $\mathsf{sk}_0 = \mathsf{sk}_1$ holds.

By utilizing this fact, we propose a patched version of the Kohlweiss-Miers ATS scheme. We modify the Account algorithm to verify that the knowledge of sk_1 in a zero-knowledge manner.

Then, using the knowledge extractor, sk_1 is recovered and then sk_0 is also recovered. Then, we can break plaintext indistinguishability under key randomization security of the underlying KOE scheme. To sum up, if the underlying KOE scheme provides plaintext indistinguishability under key randomization, then no adversary can produce conflicting witnesses.

$\mathsf{ATS.Setup}(1^\lambda)$**:** Run $\mathsf{KOE.pp} \leftarrow \mathsf{KOE.Setup}(1^\lambda)$ and $(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{KOE.KeyGen}(1^\lambda)$, erase sk_0, and output $\mathsf{ATS.pp} = (\mathsf{KOE.pp}, \mathsf{pk}_0, H)$ where $H : \{0,1\}^* \to \mathbb{Z}_p$ is a hash function modeled as the random oracle.

$\mathsf{GKeyGen}(\mathsf{ATS.pp})$**:** Parse $\mathsf{ATS.pp} = (\mathsf{pk}_0, \mathsf{KOE.pp})$. Run $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$ and a non-interactive zero-knowledge proof of knowledge π that proves knowledge of sk_1. Output $\mathsf{gpk} = (\mathsf{ATS.pp}, \mathsf{pk}_1, \pi)$ and $\mathsf{ok} = \mathsf{sk}_1$ (other values are omitted here).

$\mathsf{Account}(\mathsf{gpk}, \mathsf{cert}, w^{\mathsf{escrw}}, \mathsf{tr})$**:** Output 1 if π contained in gpk is valid and for pk' contained in cert, $\mathsf{pk}' = \mathsf{KOE.KeyRand}(\mathsf{pk}_{\mathsf{tr}}; w^{\mathsf{escrw}})$ holds. Otherwise, output 0.

For the ElGamal-type KOE scheme, π can be constructed in a standard manner: Let $\mathsf{pk} = (A, B)$ and $\mathsf{sk} = \gamma$ where $A^\gamma = B$ holds. Choose $r_\gamma \xleftarrow{\$} \mathbb{Z}_p$ and compute $R = A^{r_\gamma}$. Let $H : \{0,1\}^* \to \mathbb{Z}_p$ be a hash function modeled as a random oracle. Compute $c \leftarrow H(R, A, B)$ and $z = r_\gamma + c\gamma$. Output $\pi = (c, z)$. π is valid if $c = H(A^z/B^c, A, B)$ holds. By rewinding $\mathcal{A}$, we obtain $z = r_\gamma + c\gamma$ and $z' = r_\gamma + c'\gamma$ where $c \neq c'$, and $\gamma = (z - z')/(c - c')$ can be extracted.

Theorem 2. *The patched scheme instantiated by the ElGamal-type KOE scheme provides accountability soundness in the random oracle model.*

Proof. Let $\mathcal{C}$ be the plaintext indistinguishability under key randomization challenger of the KOE scheme and $\mathcal{A}$ be an adversary that breaks accountability soundness. We construct an algorithm $\mathcal{B}$ that breaks plaintext indistinguishability under key randomization using $\mathcal{A}$ as follows.

$\mathcal{C}$ generates $\mathsf{KOE.pp} = (g, \mathbb{G}, p)$ and $(\mathsf{pk}, \mathsf{sk})$ and sends $\mathsf{KOE.pp}$ and pk to $\mathcal{B}$. $\mathcal{B}$ sets $\mathsf{pk} = \mathsf{pk}_0$ and sends $\mathsf{ATS.pp} = (\mathsf{KOE.pp}, \mathsf{pk}_0, H)$ to $\mathcal{A}$. $\mathcal{A}$ produces gpk containing pk_1 and π, and cert and conflicting witnesses that breaks accountability soundness. By rewinding $\mathcal{A}$, $\mathcal{B}$ obtains sk_1. As mentioned before $\mathsf{sk}_1 = \mathsf{sk}_0 = \mathsf{sk}$ holds. $\mathcal{B}$ chooses any randomness r^*, computes $\mathsf{pk}^* \leftarrow \mathsf{KOE.KeyRand}(\mathsf{pk}; r^*)$, and sends (pk^*, r^*) and two challenge plaintexts (M_0^*, M_1^*) to $\mathcal{C}$. $\mathcal{C}$ flips a coin $coin \xleftarrow{\$} \{0,1\}$ and computes the challenge ciphertext for M^*_{coin}, and sends the challenge ciphertext to $\mathcal{B}$. $\mathcal{B}$ decrypts the challenge ciphertext using sk_1 and outputs coin from the decryption result M^*_{coin}, that breaks plaintext indistinguishability under key randomization. □

6 Security Analysis on Lattice-based ATS

The lattice-based KOE scheme [29], which is used in their lattice-based ATS scheme, is briefly described as follows. Let $n = \mathcal{O}(\lambda)$ be a power of 2, $q = \tilde{\mathcal{O}}(n^4)$, and $\ell = \lfloor \log \frac{q-1}{2} \rfloor + 1$. Define the rings $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = R/qR$. Let the integer bound B be of order $\tilde{\mathcal{O}}(\sqrt{n})$ and χ be a B-bounded distribution over R. The public parameters are $\mathsf{KOE.pp} = (n, q, \ell, R, R_q, B, \chi)$.

$\mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$**:** Sample $s \leftarrow \chi$, $\boldsymbol{e} \leftarrow \chi^\ell$, and $\boldsymbol{a} \xleftarrow{\$} R_q^\ell$. Output $\mathsf{pk} = (\boldsymbol{a}, \boldsymbol{b}) = (\boldsymbol{a}, \boldsymbol{a} \cdot s + \boldsymbol{e}) \in R_q^\ell \times R_q^\ell$.

$\mathsf{KOE.KeyRand}(\mathsf{pk})$**:** Sample $g \leftarrow \chi$ and $\boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow \chi^\ell$. Output $\mathsf{pk}' = (\boldsymbol{a}', \boldsymbol{b}') = (\boldsymbol{a} \cdot g + \boldsymbol{e}_1, \boldsymbol{b} \cdot g + \boldsymbol{e}_2) \in R_q^\ell \times R_q^\ell$.

We say that the condition holds for pk and pk' with $(g, \boldsymbol{e}_1, \boldsymbol{e}_2)$ if $\mathsf{pk}' = \mathsf{KOE.KeyRand}(\mathsf{pk}; (g, \boldsymbol{e}_1, \boldsymbol{e}_2))$ holds.

If we further randomize pk', then the randomized public key is described as follows.

$$\mathsf{pk}'' = (\boldsymbol{a}'', \boldsymbol{b}'') = (\boldsymbol{a}' \cdot g' + \boldsymbol{e}'_1, \boldsymbol{b}' \cdot g' + \boldsymbol{e}'_2)$$

where $g' \leftarrow \chi$ and $\boldsymbol{e}'_1, \boldsymbol{e}'_2 \leftarrow \chi^\ell$. Here, if pk' is recovered from pk'' with $g'', \boldsymbol{e}''_1, \boldsymbol{e}''_2$, then it requires that

$$\begin{aligned}
\boldsymbol{a}' &= \boldsymbol{a}'' \cdot g'' + \boldsymbol{e}''_1 = (\boldsymbol{a}' \cdot g' + \boldsymbol{e}'_1) \cdot g'' + \boldsymbol{e}''_1 = \boldsymbol{a}' \cdot g' \cdot g'' + \boldsymbol{e}'_1 \cdot g'' + \boldsymbol{e}''_1, \\
\boldsymbol{b}' &= \boldsymbol{b}'' \cdot g'' + \boldsymbol{e}''_2 = (\boldsymbol{b}' \cdot g' + \boldsymbol{e}'_2) \cdot g'' + \boldsymbol{e}''_2 = \boldsymbol{b}' \cdot g' \cdot g'' + \boldsymbol{e}'_2 \cdot g'' + \boldsymbol{e}''_2.
\end{aligned}$$

To construct conflicting witnesses, it suffices to require

$$g' \cdot g'' = 1, \ \boldsymbol{e}'_1 \cdot g'' + \boldsymbol{e}''_1 = 0, \ \text{and } \boldsymbol{e}'_2 \cdot g'' + \boldsymbol{e}''_2 = 0.$$

Assume that g' is invertible in R_q. We can then set $g'' = (g')^{-1}$, $\boldsymbol{e}''_1 = -\boldsymbol{e}'_1 \cdot g''$, and $\boldsymbol{e}''_2 = -\boldsymbol{e}'_2 \cdot g''$. Set $\mathsf{pk}_0 = \mathsf{pk}$ and $\mathsf{pk}_1 = \mathsf{pk}''$. We thus obtain two conflicting algebraic witnesses for pk' as $w_0^{\mathsf{escrw}} = (g, \boldsymbol{e}_1, \boldsymbol{e}_2)$ and $w_1^{\mathsf{escrw}} = (g'', \boldsymbol{e}''_1, \boldsymbol{e}''_2)$. In fact, most elements of R_q are invertible under the parameters of [29]. For the ATS modulus $q = 3^k$, an element $g \in R_q$ is invertible if and only if $g \mod 3$ and $X^n + 1$ are coprime over $\mathbb{F}_3[X]$. Since $X^n + 1$ has exactly two irreducible factors, each of degree $n/2$, a uniformly sampled element of R_q is invertible with overwhelming probability. However, the probability that $g \leftarrow \chi$ is invertible depends on the distribution χ, which is specified only to be B-bounded in [29].

We note that in the lattice-based ATS scheme, two KOE public keys are used for each tracing choice. For simplicity, we consider a single KOE public key for each choice, denoted by pk_0 and pk_1, respectively. We explore the $\mathsf{Account}$ algorithm in [29]. It checks whether the Ducas–Micciancio signature contained in cert is valid, and whether the condition for pk_0 and pk' with w_0^{escrw} holds (or that for pk_1 and pk' with w_1^{escrw} holds). However, it does not explicitly check the norms of the disclosed randomness. One possible countermeasure against the above attack is to additionally check whether $\|g\|_\infty, \|\boldsymbol{e}_1\|_\infty, \|\boldsymbol{e}_2\|_\infty \le B$. However, such norm checking does not necessarily rule out conflicting witnesses. As a simple example, consider $g' = 1 + X^{n/4} + (X^{n/4})^2$ and $g'' = 1 - (X^{n/4})^2 + (X^{n/4})^3$. These elements satisfy $g'g'' = 1$ with $\|g'\|_\infty = \|g''\|_\infty = 1$. Furthermore, by choosing each component of $\boldsymbol{e}'_i$ as $\pm X^t g'$ for some t, the corresponding component of $\boldsymbol{e}''_i$ becomes $\mp X^t$. Thus, both $(g', \boldsymbol{e}'_1, \boldsymbol{e}'_2)$ and $(g'', \boldsymbol{e}''_1, \boldsymbol{e}''_2)$ satisfy the required norm bounds. This is just one example, and many other constructions are possible. For instance, taking suitable powers and signed monomial shifts yields further bounded inverse pairs, and the error components can also be chosen in various ways. Thus, excluding this particular construction alone would not eliminate the possibility of conflicting witnesses.

Another possible countermeasure is to reject invertible elements in $\mathsf{KeyRand}$ and witnesses containing such elements in $\mathsf{Account}$. Specifically, whenever the sampled element g is invertible, it is rejected and resampled until a non-invertible element is obtained. In addition, any witness containing an invertible element is rejected as invalid. However, as discussed above, there are far fewer non-invertible elements than invertible ones in R_q. Thus, although invertibility can be checked efficiently, rejection sampling may be highly inefficient, depending on the distribution χ. Moreover, restricting the randomness to non-invertible elements changes its distribution. Consequently, the original security proofs may no longer apply, and the security of the modified scheme is not guaranteed.

In summary, the lattice-based ATS construction of [29] admits conflicting algebraic witnesses when the randomization element is invertible. Although such witnesses can satisfy the required

norm bounds, it remains unclear whether they can be generated according to the original distribution χ. Moreover, whether these witnesses constitute an attack on accountability soundness depends on the admissibility of the reference public keys, particularly $\mathsf{pk}_1 = \mathsf{pk}''$. Since accountability soundness is a new security notion considered in this paper, these observations do not necessarily contradict the original security guarantees of [29]. Establishing accountability soundness while preserving those guarantees remains an interesting direction for future work.

7 Security Analysis on Isogeny-based ATS

The isogeny-based KOE scheme of [34] is based on a hash-based PKE scheme constructed from CSIDH [14]. We omit some mathematical details and refer the reader to [34] for a complete description. Let $\mathsf{Cl}(\mathcal{O})$ denote an ideal class group, and let $\mathsf{Ell}_p(\mathcal{O})$ denote the set of $\mathbb{F}_p$-isomorphism classes of supersingular elliptic curves. The public parameter $\mathsf{KOE.pp}$ consists of small distinct odd primes $\ell_1, \ldots, \ell_n$, a large prime $p = 4\ell_1\ell_2 \cdots \ell_n - 1$, a base elliptic curve $E_0 \in \mathsf{Ell}_p(\mathcal{O})$ over $\mathbb{F}_p$ with endomorphism ring $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$, and an integer m satisfying $2m + 1 \geq \sqrt[n]{\sharp\mathsf{Cl}(\mathcal{O})}$, where $\sharp\mathsf{Cl}(\mathcal{O})$ denotes the cardinality of $\mathsf{Cl}(\mathcal{O})$. We also omit the function that maps an isomorphism class of elliptic curves to its Montgomery coefficient, as well as the hash functions used in the construction. The algorithm $\mathsf{KOE.KeyGen}$ takes $\mathsf{KOE.pp}$ and samples a vector $\mathsf{u} = (u_1, \ldots, u_n)$ uniformly at random, where $u_i \in [-m, m]$ for all $i \in \{1, \ldots, n\}$. This vector defines an ideal class $[\mathfrak{u}] = [\mathfrak{l}_1^{u_1}\mathfrak{l}_2^{u_2} \cdots \mathfrak{l}_n^{u_n}] \in \mathsf{Cl}(\mathcal{O})$, where $\mathfrak{l}_i = \langle \ell_i, \pi - 1 \rangle$ and π denotes the Frobenius endomorphism. Following the notation of [21], we write $[\mathfrak{u}]E$ instead of $[\mathfrak{u}]E$, since $\mathfrak{u}$ is uniquely determined by the vector $\mathsf{u} = (u_1, \ldots, u_n)$. Likewise, let $[\mathfrak{v}] = [\mathfrak{l}_1^{v_1}\mathfrak{l}_2^{v_2} \cdots \mathfrak{l}_n^{v_n}]$ with $\mathsf{v} = (v_1, \ldots, v_n)$. Then $[\mathfrak{u}][\mathfrak{v}]E = [\mathsf{u} + \mathsf{v}]E$, since $[\mathfrak{u}][\mathfrak{v}] = [\mathfrak{l}_1^{u_1+v_1}\mathfrak{l}_2^{u_2+v_2} \cdots \mathfrak{l}_n^{u_n+v_n}]$. The $\mathsf{KOE.KeyRand}$ algorithm of [34] exploits this linearity.

$\mathsf{KOE.KeyGen}(\mathsf{KOE.pp})$**:** Randomly sample $\mathsf{a} = (a_1, \ldots, a_n)$ and $\mathsf{r} = (r_1, \ldots, r_n)$ where $a_i, r_i \in [-m, m]$ for $i = 1, 2, \ldots, n$. $\mathsf{pk} = (E_1, E_2) = ([\mathsf{a}]E_0, [\mathsf{a} + \mathsf{r}]E_0)$ and $\mathsf{sk} = \mathsf{r}$.

$\mathsf{KOE.KeyRand}(\mathsf{pk})$**:** Randomly sample $\mathsf{r}' = (r'_1, \ldots, r'_n)$ where $r'_i \in [-m, m]$ for $i = 1, 2, \ldots, n$. Output $\mathsf{pk}' = (E'_1, E'_2) = ([\mathsf{r}']E_1, [\mathsf{r}']E_2)$.

Here, $[\mathsf{r}']E_1 = [\mathsf{r}'][\mathsf{a}]E_0 = [\mathsf{r}' + \mathsf{a}]E_0$ and $[\mathsf{r}']E_2 = [\mathsf{r}'][\mathsf{a} + \mathsf{r}]E_0 = [\mathsf{r}' + \mathsf{a} + \mathsf{r}]E_0$ hold. Because of the randomization procedure, a public key can be randomized more than once. Moreover, if the randomization vector $\mathsf{r}' = (r'_1, \ldots, r'_n)$ is known, then the source public key pk can be recovered from the randomized public key pk' by applying $-\mathsf{r}' = (-r'_1, \ldots, -r'_n)$ such that

$$[-\mathsf{r}']E'_1 = [-\mathsf{r}'][\mathsf{r}']E_1 = E_1 \text{ and } [-\mathsf{r}']E'_2 = [-\mathsf{r}'][\mathsf{r}']E_2 = E_2$$

hold. Therefore, our attack is effective against the isogeny-based ATS scheme, showing that it does not provide accountability soundness. We stress that accountability soundness is a new security notion introduced in this paper. Thus, the attacks discussed in this work fall outside the security models considered in [34] and do not contradict their claimed security guarantees. We expect that an isogeny-based proof-of-knowledge technique, such as those in [5, 16–19], could be applicable to prevent our attack as in the patched scheme.

8 Conclusion

In this paper, we formalized accountability soundness and showed that the Kohlweiss-Miers ATS scheme does not satisfy this property. We further proposed a patched version of the scheme and

argued that it achieves accountability soundness under the security of the underlying KOE scheme. We also discussed the applicability of our attack to lattice-based and isogeny-based ATS schemes [29, 34], together with possible countermeasures. Determining the effectiveness of these countermeasures and rigorously analyzing modified constructions that preserve the intended security guarantees are left for future work.

Acknowledgment: This work was supported by JSPS KAKENHI Grant Number JP26K00966. This paper was edited for grammar using Copilot.

References

[1] Joël Alwen, Sandro Coretti, Yevgeniy Dodis, and Yiannis Tselekounis. Security analysis and improvements for the IETF MLS standard for group messaging. In *CRYPTO, Part I*, pages 248–277, 2020.

[2] Joël Alwen, Georg Fuchsbauer, and Marta Mularczyk. Updatable public-key encryption, revisited. In *EUROCRYPT, Part VI*, pages 346–376, 2024.

[3] Kyoichi Asano and Yohei Watanabe. Updatable public key encryption with strong CCA security: Security analysis and efficient generic construction. In *CT-RSA*, pages 223–246, 2025.

[4] James Bartusek, Sanjam Garg, Abhishek Jain, and Guru-Vamsi Policharla. End-to-end secure messaging with traceability only for illegal content. In *EUROCRYPT, Part V*, pages 35–66, 2023.

[5] Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust. In *EUROCRYPT, Part II*, pages 405–437, 2023.

[6] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT*, pages 614–629, 2003.

[7] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA*, pages 136–153, 2005.

[8] Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: generic, simple, and efficient. *Designs, Codes and Cryptography*, 91(6):2141–2200, 2023.

[9] Dan Boneh and Chelsea Komlo. Threshold signatures with private accountability. In *CRYPTO, Part IV*, pages 551–581, 2022.

[10] Dan Boneh, Aditi Partap, and Lior Rotem. Proactive refresh for accountable threshold signatures. In *Financial Cryptography and Data Security, Part II*, pages 140–159, 2024.

[11] Dan Boneh, Aditi Partap, and Brent Waters. Accountable multi-signatures with constant size public keys. In *Public-Key Cryptography, Part II*, pages 32–65, 2025.

[12] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, and Jens Groth. Foundations of fully dynamic group signatures. In *Applied Cryptography and Network Security*, pages 117–136, 2016.

[13] Ran Canetti, Abhishek Jain, and Alessandra Scafuro. Practical UC security with a global random oracle. In *ACM CCS*, pages 597–608. ACM, 2014.

[14] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In *ASIACRYPT, Part III*, pages 395–427, 2018.

[15] David Chaum and Eugène van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.

[16] Kelong Cong, Yi-Fu Lai, and Shai Levin. Efficient isogeny proofs using generic techniques. In *Applied Cryptography and Network Security, Part II*, pages 248–275, 2023.

[17] Thomas den Hollander, Sören Kleine, Marzio Mula, Daniel Slamanig, and Sebastian A. Spindler. More efficient isogeny proofs of knowledge via canonical modular polynomials. In *CRYPTO, Part I*, pages 131–166, 2025.

[18] Thomas den Hollander, Shai Levin, Marzio Mula, Robi Pedersen, Daniel Slamanig, and Sebastian A. Spindler. SPRINT: New isogeny proofs of knowledge and isogeny-based signatures. In *ASIACRYPT*, 2026, to appear. Available at `https://eprint.iacr.org/2026/364`.

[19] Thomas den Hollander, Marzio Mula, Daniel Slamanig, and Sebastian A. Spindler. On the use of atkin and weber modular polynomials in isogeny proofs of knowledge. In *Post-Quantum Cryptography, Part II*, pages 39–73, 2026.

[20] Yevgeniy Dodis, Harish Karthikeyan, and Daniel Wichs. Updatable public key encryption in the standard model. In *Theory of Cryptography, Part III*, pages 254–285, 2021.

[21] Luca De Feo and Michael Meyer. Threshold schemes from isogeny assumptions. In *Public-Key Cryptography, Part II*, pages 187–212, 2020.

[22] Mike Graf, Ralf Küsters, and Daniel Rausch. AUC: accountable universal composability. In *IEEE Symposium on Security and Privacy*, pages 1148–1167. IEEE, 2023.

[23] Ai Ishida, Keita Emura, Goichiro Hanaoka, Yusuke Sakai, and Keisuke Tanaka. Group signature with deniability: How to disavow a signature. In *CANS*, pages 228–244, 2016.

[24] Daniel Jost, Ueli Maurer, and Marta Mularczyk. Efficient ratcheting: Almost-optimal guarantees for secure messaging. In *EUROCRYPT, Part I*, pages 159–188, 2019.

[25] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable signatures. In *EUROCRYPT*, pages 571–589, 2004.

[26] Markulf Kohlweiss and Ian Miers. Accountable metadata-hiding escrow: A group signature case study. *Privacy Enhancing Technologies*, 2015(2):206–221, 2015.

[27] Benoît Libert, Fabrice Mouhartem, and Khoa Nguyen. A lattice-based group signature scheme with message-dependent opening. In *Applied Cryptography and Network Security*, pages 137–155, 2016.

[28] Benoît Libert and Moti Yung. Efficient traceable signatures in the standard model. *Theoretical Computer Science*, 412(12-14):1220–1242, 2011.

[29] San Ling, Khoa Nguyen, Huaxiong Wang, and Yanhong Xu. Accountable tracing signatures from lattices. In *CT-RSA*, pages 556–576, 2019.

[30] San Ling, Khoa Nguyen, Huaxiong Wang, and Yanhong Xu. Lattice-based group signatures: Achieving full dynamicity (and deniability) with ease. *Theoretical Computer Science*, 783:71–94, 2019.

[31] Kazuma Ohara, Yusuke Sakai, Keita Emura, and Goichiro Hanaoka. A group signature scheme with unbounded message-dependent opening. In *ACM ASIACCS*, pages 517–522. ACM, 2013.

[32] Sunoo Park and Adam Sealfon. It wasn't me! - repudiability and claimability of ring signatures. In *CRYPTO, Part III*, pages 159–190, 2019.

[33] Yusuke Sakai, Keita Emura, Goichiro Hanaoka, Yutaka Kawai, Takahiro Matsuda, and Kazumasa Omote. Group signatures with message-dependent opening. In *Pairing-Based Cryptography*, pages 270–294, 2012.

[34] Surbhi Shaw and Ratna Dutta. Key-oblivious encryption from isogenies with application to accountable tracing signatures. In *INDOCRYPT*, pages 362–386, 2021.

[35] Nam Tran, Khoa Nguyen, Dongxi Liu, Josef Pieprzyk, and Willy Susilo. Traceable signatures from lattices. *Computer Standards & Interfaces*, 96:104092, 2026.

[36] Takuma Watanabe and Keita Emura. On publicly verifiable tokens in group signatures with message-dependent opening. In *ICICS, Part V*, pages 127–138.