On Publicly Verifiable Tokens in Group Signatures with Message-Dependent Opening

Takuma Watanabe[1] and Keita Emura*[1,2]

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

September 10, 2026

Abstract

Group signatures (GSs; Chaum and van Heyst, EUROCRYPT 1991) are digital signatures that allow a signer to anonymously prove group membership, while still enabling a special authority, called the opener, to identify the signer when necessary. Group Signatures with Message-Dependent Opening (GS-MDO; Sakai et al., Pairing 2012) weaken the power of the opener by introducing another authority, the admitter, who issues a message-dependent token. In previous GS-MDO schemes, these tokens can be viewed as signatures. Therefore they can be publicly verified using the verification algorithm of the underlying signature scheme. However, no explicit notion of public verifiability for tokens, meaning the ability to publicly verify whether a token can be used for opening a group signature, has been defined so far. Clarifying this implicit security property is important for understanding the feasibility of GS-MDO. In this paper, we formally define public verifiability of tokens. We establish a proper relationship between verifying a token as a signature and verifying that the token can be used for opening, which typically requires the opener's secret key. We also show that the Ohara et al. pairing-based GS-MDO scheme (AsiaCCS 2013), the Libert et al. lattice-based GS-MDO scheme (ACNS 2016), and the Libert et al. pairing-based GS-MDO scheme (CT-RSA 2014) satisfy our definition, suggesting that our formalization is reasonable. Finally, we discuss how publicly verifiable tokens can be used to provide accountability for the admitter, enabling them to demonstrate that tokens have been honestly generated according to the token-generation algorithm.

1 Introduction

Group Signatures (GSs) [19] are a type of signature scheme that balances anonymity with traceability. While a verifier can verify a signature anonymously without identifying the signer, a group manager has the authority to identify the signer. It should be noted that because the group manager can identify the signer for any given group signature, their authority is excessively strong (or requires a trust assumption that they will not identify honest signers). Sakai et al. proposed Group Signatures with Message-Dependent Opening (GS-MDO) [45] to weaken the power of the opener by introducing another authority, the admitter. The opener holds an opening key ok, and the admitter holds an admitter secret key ask. For a group signature σ generated for a message M by a user with a signing key gsk_i, the admitter generates a message-dependent token t_{M} and

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

presents it to the opener. The opener can identify the signer by using both $\mathsf{t_M}$ and ok. To ensure that neither entity can identify the signer alone, the properties of opener anonymity and admitter anonymity are defined.

Generally, group signatures can be constructed from public-key encryption (PKE), digital signatures, and non-interactive zero-knowledge (NIZK) proofs [7]. Furthermore, because PKE can be constructed from group signatures [1], it has been shown that PKE (or cryptographic primitives similar to PKE) is essential for constructing GSs. Sakai et al. [45] demonstrated that identity-based encryption (IBE) can be constructed from GS-MDO, implying that IBE (or cryptographic primitives similar to IBE) is necessary for constructing GS-MDO. Additionally, Imura et al. [31] showed that timed-release encryption (TRE) can be directly constructed from GS-MDO. Although IBE can be constructed from cryptographic primitives similar to PKE via non-black-box techniques [23, 24], and although we do not address the feasibility of NIZKs, these feasibility results [1, 31, 45] suggest that GS-MDO is a stronger primitive than GS, given the impossibility of constructing IBE from PKE in a black-box manner [13]. To support this claim, clarifying the implicit security property is important for understanding the feasibility of GS-MDO.

Unforgeability of Tokens. Since GS-MDO implies IBE, the following observations suggest that tokens in GS-MDO naturally inherit the role of signatures and tokens are unforgeable in a certain sense: (1) secret keys in IBE can be regarded as signatures (via the Naor transformation), and (2) the IBE secret key in an IBE scheme derived from GS-MDO corresponds to a token in the GS-MDO setting. Imura et al. [32] formalized this unforgeability of tokens as a natural extension of existential unforgeability under chosen-message attacks (EUF-CMA).[1] As another motivation for introducing token unforgeability, Imura et al. discussed the security of an ID-management system based on GS-MDO [30], which employs the GS-MDO scheme of Ohara et al. [44]. They pointed out that tokens in this system are BLS (Boneh-Lynn-Shacham) signatures [12], allowing users to verify tokens as service-provision certificates. From this perspective, token unforgeability was introduced to ensure that users in the system cannot forge certificates, thereby providing provable security. We remark that token unforgeability is implicitly achieved by all GS-MDO schemes, since it is implied by opener anonymity, which is an essential security requirement for GS-MDO. Imura et al. also noted that tokens correspond to GPV (Gentry-Peikert-Vaikuntanathan) signatures [27] in the lattice-based GS-MDO scheme of Libert et al. [39]. Because both BLS and GPV signature schemes are strongly existentially unforgeable, Imura et al. suggested that these GS-MDO schemes may satisfy a stronger (or qualitatively different) security property than token unforgeability, one that does not fully capture strong EUF-CMA security.

Public Verifiability of Tokens as Signatures. In addition to the pairing-based GS-MDO scheme of Ohara et al. [44] and the lattice-based GS-MDO scheme of Libert et al. [39], we clarify that tokens are signatures in all other previously proposed GS-MDO schemes; that is, they are publicly verifiable *as signatures*. Concretely, in the pairing-based GS-MDO scheme of Libert et al. [38], a token consists of two Waters signatures [29, 48]. In the lattice-based GS-MDO scheme of Chen et al. [20], tokens correspond to GPV signatures [27]. Anada et al. [3] proposed a generic construction for GS-MDO in which the secret key of an ID-KEM (Identity-based Key Encapsulation Mechanism) scheme is used as the token. As an instantiation, they employed the Heng-Kurosawa (k-resilient) ID-KEM scheme [28]. Because a secret key in the Heng-Kurosawa ID-KEM scheme is defined as a point $f(ID)$ on a k-th degree polynomial f, and the values $(g^{d_0}, \ldots, g^{d_k})$ corresponding to the polynomial coefficients $(d_0, \ldots, d_k)$ are included in the public parameters, it is publicly verifiable.

Our Motivation: Public Verifiability of Tokens in GS-MDO. The fact that tokens are sig-

[1] Unforgeability of tokens was introduced in [32] which is the full version of the paper [31].

natures and are publicly verifiable as signatures in previously proposed GS-MDO schemes suggests that public verifiability of tokens in GS-MDO, meaning the ability to publicly verify whether a token can be used for opening a group signature, is a natural security property for GS-MDO. Clarifying this implicitly achieved security property is expected to provide insights into the feasibility of GS-MDO. The formalization of this public verifiability of tokens in the context of GS-MDO serves as the primary motivation of this paper.

Here, we clarify that unforgeability of tokens does not guarantee that tokens are publicly verifiable in GS-MDO. The formal definition of unforgeability of tokens is given in Section **??** for the sake of clarity. To explain the gap between these notions, we first review the relationship among GS-MDO, IBE, and signatures. We begin by reviewing the signature verification algorithm obtained via the Naor transformation. In this transformation, a message M to be signed is treated as an identity in the underlying IBE scheme, and an IBE secret key for M is regarded as a signature on M. The signature verification algorithm operates as follows:

- A random plaintext R is encrypted under the identity M.
- If decrypting the ciphertext with the IBE secret key yields R, then the IBE secret key is regarded as a valid signature on M.

Since IBE is a public-key encryption, no secret value is required for running the encryption algorithm in the signature verification procedure.

As mentioned above, Sakai et al. [45] showed that IBE can be constructed from GS-MDO. In the resulting GS-MDO-based IBE scheme, an IBE secret key corresponds to a GS-MDO token. Therefore, a GS-MDO token can naturally be viewed as a signature. When constructing a signature scheme from a GS-MDO-based IBE scheme via the Naor transformation, that is, when verifying that a token t_M for a message M is a valid IBE secret key (i.e., it can be used for opening in the GS-MDO setting), a straightforward adaptation of the Naor transformation proceeds as follows:

- A user i is chosen at random, and a group signature on the message M is generated using gsk_i.
- If opening the group signature using ok and t_M yields i, then t_M is regarded as a valid signature on M.

In the above verification procedure, the signing key gsk_i is required for generating a group signature, and the opening key ok is required for opening the group signature. Since these are secret values in the GS-MDO context, the token is not publicly verifiable in GS-MDO. We emphasize that, in the IBE construction from GS-MDO by Sakai et al., all signing keys and the opening key ok are included in the public parameters of the GS-MDO-based IBE scheme.

We note that a signature scheme directly derived from an IBE scheme is not necessarily identical to the signature scheme obtained via the Naor transformation. For example, in the Boneh-Franklin IBE scheme [10], a secret key can be interpreted as a BLS signature [12] whose signing algorithm is deterministic, whereas the signing algorithm of the signature scheme obtained via the Naor transformation is probabilistic due to the use of a random plaintext. This observation suggests that the signature scheme naturally associated with a GS-MDO-based IBE scheme need not coincide with the one obtained via the Naor transformation. Indeed, as mentioned above, in all previously proposed GS-MDO schemes, tokens are publicly verifiable as signatures. However, these observations alone do not guarantee public verifiability of tokens in the GS-MDO setting. We clarify this point in the next paragraph.

Difficulty of Formalization. First, we clarify that the following two situations are completely distinct.

- Verify that a token $\mathsf{t_M}$ is a valid signature on M.
- Verify that a token $\mathsf{t_M}$ can be used for opening a group signature σ on M.

This distinction explains why token unforgeability, which is naturally defined based on the EUF-CMA security of the signature scheme corresponding to the token, does not guarantee public verifiability of tokens. As mentioned earlier, directly verifying whether a token can be used for opening requires secret values (gsk and ok), which contradicts the notion of public verifiability. On the other hand, in actual GS-MDO schemes, tokens are indeed signatures, and thus tokens are publicly verifiable as signatures. To sum up, what we need to formalize is how to suitably relate *the verification that a token is valid as a signature* to *the verification that a token can be used for opening a group signature*. Since the former is publicly verifiable whereas the latter is not directly publicly verifiable, it is non-trivial to formalize public verifiability of tokens in GS-MDO.

Our Contributions. In this paper, we formally define public verifiability of tokens in GS-MDO and prove that the pairing-based GS-MDO scheme of Ohara et al. [44], the lattice-based GS-MDO scheme of Libert et al. [39], and the pairing-based GS-MDO scheme of Libert et al. [38] satisfy our definition, suggesting that our formalization is reasonable.

We also discuss how publicly verifiable tokens can be used to provide accountability for the admitter, enabling the admitter to demonstrate that tokens have been honestly generated according to the token-generation algorithm. In brief, the pairing-based GS-MDO scheme of Ohara et al. provides accountability of the admitter because tokens remain publicly verifiable even against an adversary that possesses the admitter's secret key, whereas the Libert et al. GS-MDO schemes do not provide such accountability (we emphasize that this observation does not constitute an attack on the Libert et al. schemes).

Technical Overview. In our formalization, we focus on correctness of GS-MDO and utilize the fact that any token generated by the trapdoor generation algorithm Td can be used for opening a group signature. Intuitively, we introduce a set of all tokens for a message M generated by the Td algorithm as:

$$\mathsf{TSet_M} := \{(\mathsf{t_M}, \mathsf{M}) : \mathsf{t_M} \leftarrow \mathsf{Td}(\mathsf{ask}, \mathsf{M})\}$$

where ask is the admitter's secret key. We define that a token is publicly verifiable if there exists a publicly executable token verification algorithm TVf such that:

$$\mathsf{TVf}(\mathsf{apk}, \mathsf{t_M}, \mathsf{M}) = 1 \text{ iff } (\mathsf{t_M}, \mathsf{M}) \in \mathsf{TSet_M}$$

where apk is the admitter's public key. Later, we define public verifiability of token in a computational security manner where an PPT (probabilistic polynomial-time) adversary $\mathcal{A}$ wins if $\mathcal{A}$ produces $(\mathsf{t_{M^*}}, \mathsf{M}^*)$ satisfying $\mathsf{TVf}(\mathsf{apk}, \mathsf{t_{M^*}}, \mathsf{M}^*) = 1 \wedge (\mathsf{t_{M^*}}, \mathsf{M}^*) \notin \mathsf{TSet_{M^*}}$. Here, correctness of GS-MDO guarantees that any token included in $\mathsf{TSet_M}$ can be used for identifying the actual signer of M via the Open algorithm (with ok). Thus, our formalization adequately relates signature verification via the public key to verifying whether a token can be used for opening. Our formalization is inspired by the definition of public verifiability for predicate encryption [49] that also consider correctness.[2]

Differences from the proceedings version. An extended abstract appeared at the 28th International Conference on Information and Communications Security (ICICS 2026) that considered Full

[2]Briefly, for a ciphertext $CT \in \{0,1\}^*$ and a ciphertext attribute Y, if the verification algorithm outputs 1, then there exists a plaintext such that CT belongs to the set of all possible ciphertexts generated by the encryption algorithm for that plaintext and Y. See Definition 3 in [49] for details.

Domain Hash (FDH)-type tokens in the Ohara et al. pairing-based GS-MDO scheme [44] and the Libert et al. lattice-based GS-MDO scheme [39]. We employed the fact that once $\mathsf{pp}_{\mathsf{GS\text{-}MDO}}$ and apk are fixed, the token t_{M} for a certain message M is uniquely determined. We also employ string unforgeability of the GPV signature scheme. In this full version, we additionally prove that the Libert et al. pairing-based GS-MDO scheme [38] provides public verifiability of tokens. A token consists of two Waters signatures [29, 48], and each $\mathsf{TSet}_{\mathsf{M}}$ contains more than one (exponentially-many) signatures since the signing algorithm of the Waters signature scheme is probabilistic. Moreover, the Waters signature scheme is not strongly unforgeable since it can be re-randomized. Due to this situation, we need to consider another methodology for proving that the scheme provides public verifiability of tokens. We also explore distributed group signatures with signature-dependent opening [9], which appear to provide public verifiability of tokens, and discuss why this observation does not affect the contributions of this paper.

Related Work. Blazy and Mallordy [9] proposed distributed group signatures with signature-dependent opening, which can be viewed as an extension of GS-MDO. In their scheme, tokens are generated for a signature, and once more than a threshold number of tokens, together with the opener's secret key, are collected, the opener can identify the signer of the signature. In their generic construction, they employ dynamic threshold public-key encryption (dynamic TPKE) [22, 46] as a building block. A signature corresponds to a ciphertext of the underlying dynamic TPKE scheme along with NIZK proofs, and each token corresponds to a decryption share. Although Blazy and Mallordy do not explicitly discuss the public verifiability of decryption shares, and their lattice-based instantiation does not provide a share verification algorithm, a dynamic TPKE scheme inherently provides such a mechanism. That is, each decryption share is publicly verifiable. Consequently, if other dynamic TPKE schemes such as the pairing-based dynamic TPKE scheme by Delerablée and Pointcheval [22], a generic construction of dynamic TPKE [46] from public-key encryption with non-interactive opening (PKENO) [21, 26], or the pairing-based dynamic TPKE scheme by Sakai et al. [46] are used as building blocks, the resulting scheme seems to provide public verifiability of tokens. We note that this observation applies to signature-dependent opening rather than message-dependent opening, and heavily depends on threshold decryption. Therefore, this observation does not affect the contributions of this paper.

2 GS-MDO

In this section, we define GS-MDO. In this paper, we adopt the syntax that separates the key generation for the opener and the admitter [31]. A GS-MDO scheme $\mathsf{GS\text{-}MDO}$ consists of the following seven algorithms: $(\mathsf{GS\text{-}MDO.Setup}, \mathsf{GKg}, \mathsf{AKg}, \mathsf{GSig}, \mathsf{Td}, \mathsf{GVf}, \mathsf{Open})$.

Definition 1 (Syntax of GS-MDO)**.**

$\mathsf{GS\text{-}MDO.Setup}$**:** *The setup algorithm takes the security parameter $\lambda \in \mathbb{N}$ as input and outputs the common parameter $\mathsf{pp}_{\mathsf{GS\text{-}MDO}}$. Hereafter, it is assumed that all algorithms take $\mathsf{pp}_{\mathsf{GS\text{-}MDO}}$ as input, even if not explicitly stated.*

GKg**:** *The group key generation algorithm takes $\mathsf{pp}_{\mathsf{GS\text{-}MDO}}$ and the number of group members $N \in \mathbb{N}$ as input, and outputs the group public key gpk, the opening key ok, and the set of group signing keys $\{\mathsf{gsk}_i\}_{i\in[N]}$.*

AKg**:** *The admitter key generation algorithm takes $\mathsf{pp}_{\mathsf{GS\text{-}MDO}}$ as input and outputs the admitter public key apk and the admitter secret key ask.*

GSig: *The signing algorithm takes* gpk, apk, gsk_i, *and a message* M *as input, and outputs a group signature* σ.

Td: *The message-dependent token generation algorithm takes* ask *and* M *as input, and outputs a token* t_M.

GVf: *The signature verification algorithm takes* gpk, apk, M, *and* σ *as input, and outputs 0 or 1.*

Open: *The signer tracing algorithm takes* gpk, apk, ok, M, σ, *and* t_M *as input, and outputs a user index* $i \in [N]$ *or* $\perp$.

Next, we define correctness. This property guarantees that if all values are generated according to the algorithms, the group signature passes verification and the signer is correctly identified. We define that GS-MDO is correct if, for all $\mathsf{pp}_{\mathsf{GS\text{-}MDO}} \leftarrow \mathsf{GS\text{-}MDO.Setup}(1^\lambda)$, $(\mathsf{gpk}, \mathsf{ok}, \{\mathsf{gsk}_i\}_{i\in[N]}) \leftarrow \mathsf{GKg}(\mathsf{pp}_{\mathsf{GS\text{-}MDO}}, 1^N)$, $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{AKg}(\mathsf{pp}_{\mathsf{GS\text{-}MDO}})$, $\mathsf{M} \in \{0,1\}^*$, and $i \in [N]$, it holds that $\mathsf{GVf}(\mathsf{gpk}, \mathsf{apk}, \mathsf{M}, \mathsf{GSig}(\mathsf{gpk}, \mathsf{apk}, \mathsf{gsk}_i, \mathsf{M})) = 1$ and $\mathsf{Open}(\mathsf{gpk}, \mathsf{apk}, \mathsf{ok}, \mathsf{M}, \mathsf{GSig}(\mathsf{gpk}, \mathsf{apk}, \mathsf{gsk}_i, \mathsf{M}), \mathsf{Td}(\mathsf{ask}, \mathsf{M})) = i$.

Next, we review the notion of unforgeability of tokens (token unforgeability) and clarify its relationship to public verifiability of tokens. Token unforgeability guarantees that an adversary who does not possess the admitter's secret key ask cannot forge a token that can be used for opening a group signature. This notion can be formalized without introducing a token verification algorithm because the challenger generates a group signature and executes the Open algorithm in the security experiment. As noted in the introduction, this procedure requires gsk_i and ok and is therefore not publicly executable.

Definition 2 (Token Unforgeability [32])**.** *Let* $\mathcal{A}$ *be an adversary and* $\mathcal{C}$ *be the challenger.* $X \xleftarrow{R} S$ *denotes that an element* X *is chosen uniformly at random from a set* S.

Setup: $\mathcal{C}$ *computes* $\mathsf{pp}_{\mathsf{GS\text{-}MDO}} \leftarrow \mathsf{GS\text{-}MDO.Setup}(1^\lambda)$, $(\mathsf{gpk}, \mathsf{ok}, \{\mathsf{gsk}_i\}_{i\in[N]}) \leftarrow \mathsf{GKg}(\mathsf{pp}_{\mathsf{GS\text{-}MDO}}, 1^N)$, *and* $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{AKg}(\mathsf{pp}_{\mathsf{GS\text{-}MDO}})$, *and gives* $(\mathsf{gpk}, \mathsf{ok}, \mathsf{apk}, \{\mathsf{gsk}_i\}_{i\in[N]})$ *to* $\mathcal{A}$.

Token Query: $\mathcal{A}$ *sends* M *to* $\mathcal{C}$. *If* M *is not previously queried, then* $\mathcal{C}$ *computes* $\mathsf{t}_\mathsf{M} \leftarrow \mathsf{Td}(\mathsf{ask}, \mathsf{M})$ *and stores* $(\mathsf{M}, \mathsf{t}_\mathsf{M})$. *Otherwise,* $\mathcal{C}$ *retrieves* t_M. $\mathcal{C}$ *returns* t_M *to* $\mathcal{A}$.

Forge: $\mathcal{A}$ *outputs* $(\mathsf{M}^*, \mathsf{t}_{\mathsf{M}^*})$. $\mathcal{C}$ *chooses* $i \xleftarrow{R} [N]$ *and computes* $\sigma^* \leftarrow \mathsf{GSig}(\mathsf{gpk}, \mathsf{apk}, \mathsf{gsk}_i, \mathsf{M}^*)$. $\mathcal{A}$ *wins if* $\mathsf{Open}(\mathsf{gpk}, \mathsf{apk}, \mathsf{ok}, \mathsf{M}^*, \sigma^*, \mathsf{t}_{\mathsf{M}^*}) = i$ *holds and* $\mathcal{A}$ *did not send* M^* *as a token query.*

We say that GS-MDO *provides token unforgeability if, for all PPT adversaries* $\mathcal{A}$*, the advantage*

$$\mathsf{Adv}^{\mathsf{token\text{-}unforge}}_{\mathsf{GS\text{-}MDO},\mathcal{A}}(\lambda, N) = \Pr[\mathcal{A} \textit{ wins}]$$

is negligible in the security parameter.

In GS-MDO, the following security properties are defined. Opener anonymity guarantees that even the opener cannot identify the signer without the corresponding token. Note that unforgeability of tokens is implied by opener anonymity as shown in [32]. Indeed, if an adversary could forge a token that can be used for opening a group signature, it could open the challenge group signature and thereby identify the signer, which would violate opener anonymity. Admitter anonymity guarantees that even the admitter cannot identify the signer without the opening key. Traceability guarantees that, for any valid group signature (i.e., one for which the GVf algorithm outputs 1), the signer can be identified using the corresponding token and the opening key.

3 Public Verifiability of Tokens

In this section, we define public verifiability of tokens. First, we define the token verification algorithm TVf as follows:

Definition 3 (Token Verification Algorithm)**.**

TVf**:** *The token verification algorithm takes the admitter's public key* apk*, a token* $\mathsf{t_M}$*, and a message* M *as input, and outputs 0 or 1.*

Note that since a token is generated by ask, it is expected that gpk is not required for its verification. Therefore, we have adopted a definition in which the algorithm does not take gpk as input.

Naturally, we can extend the correctness: it guarantees that if all values are generated according to the algorithms, the group signature passes verification, the signer is correctly identified, and the token passes verification. We define that $\mathsf{GS\text{-}MDO}$ is correct if, for all $\mathsf{pp_{GS\text{-}MDO}} \leftarrow \mathsf{GS\text{-}MDO.Setup}(1^\lambda)$, $(\mathsf{gpk}, \mathsf{ok}, \{\mathsf{gsk}_i\}_{i\in[N]}) \leftarrow \mathsf{GKg}(\mathsf{pp_{GS\text{-}MDO}}, 1^N)$, $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{AKg}(\mathsf{pp_{GS\text{-}MDO}})$, $\mathsf{M} \in \{0,1\}^*$, and $i \in [N]$, it holds that $\mathsf{GVf}(\mathsf{gpk}, \mathsf{apk}, \mathsf{M}, \mathsf{GSig}(\mathsf{gpk}, \mathsf{apk}, \mathsf{gsk}_i, \mathsf{M})) = 1$, $\mathsf{Open}(\mathsf{gpk}, \mathsf{apk}, \mathsf{ok}, \mathsf{M}, \mathsf{GSig}(\mathsf{gpk}, \mathsf{apk}, \mathsf{gsk}_i, \mathsf{M}), \mathsf{Td}(\mathsf{ask}, \mathsf{M})) = i$, and $\mathsf{TVf}(\mathsf{apk}, \mathsf{Td}(\mathsf{ask}, \mathsf{M}), \mathsf{M}) = 1$.

Next, we define public verifiability of tokens. Since tokens correspond to signatures in actual GS-MDO schemes, the primary motivation of this paper is to formalize the public verifiability inherently provided by these schemes. Therefore, we first consider a security-oriented definition, from which one may expect a reduction from public verifiability of tokens to the security of the corresponding signature scheme. In this interpretation, ask can be regarded as a signing key and the Td algorithm can be regarded as a signing algorithm. Since the security of a signature scheme is typically defined against an adversary that does not obtain the secret signing key, it is natural to consider an adversary $\mathcal{A}$ that is given all keys except ask, is allowed to access the token oracle, and outputs a token-message pair $(\mathsf{t_{M^*}}, \mathsf{M}^*)$. We say that $\mathcal{A}$ wins if $\mathsf{TVf}(\mathsf{apk}, \mathsf{t_{M^*}}, \mathsf{M}^*) = 1$ and $(\mathsf{t_{M^*}}, \mathsf{M}^*) \notin \mathsf{TSet_{M^*}}$. Note that the token oracle returns tokens generated by the Td algorithm. That is, for a token query M and its response $\mathsf{t_M}$, $(\mathsf{t_M}, \mathsf{M}) \in \mathsf{TSet_M}$ holds. Thus, the adversary's output is automatically different from the inputs and outputs of the token oracle. Specifically, for the set of all token oracle inputs M and their outputs $\mathsf{t_M}$, denoted as $\{(\mathsf{t_M}, \mathsf{M})\}$, it holds that $(\mathsf{t_{M^*}}, \mathsf{M}^*) \notin \{(\mathsf{t_M}, \mathsf{M})\}$.

We provide the formal definition below. Considering that BLS and GPV signatures are Full Domain Hash (FDH) signatures, i.e., the signing algorithm is deterministic and does not use randomness, we define the token oracle to return the previously responded token for a token query on the same message. If the token generation algorithm is probabilistic, it can simply be modified that the oracle runs $\mathsf{t_M} \leftarrow \mathsf{Td}(\mathsf{ask}, \mathsf{M})$ and returns $\mathsf{t_M}$ to $\mathcal{A}$.

Definition 4 (Public Verifiability of Tokens)**.** *Let* $\mathcal{A}$ *be an adversary and* $\mathcal{C}$ *be the challenger.*

Setup: $\mathcal{C}$ *runs* $\mathsf{pp_{GS\text{-}MDO}} \leftarrow \mathsf{GS\text{-}MDO.Setup}(1^\lambda)$, $(\mathsf{gpk}, \mathsf{ok}, \{\mathsf{gsk}_i\}_{i\in[N]}) \leftarrow \mathsf{GKg}(\mathsf{pp_{GS\text{-}MDO}}, 1^N)$*, and* $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{AKg}(\mathsf{pp_{GS\text{-}MDO}})$*, and sends* $(\mathsf{gpk}, \mathsf{ok}, \mathsf{apk}, \{\mathsf{gsk}_i\}_{i\in[N]})$ *to* $\mathcal{A}$*.*

Token Query: $\mathcal{A}$ *sends* M*.* $\mathcal{C}$ *checks whether* $(\mathsf{t_M}, \mathsf{M})$ *has been stored. If it is stored,* $\mathcal{C}$ *returns* $\mathsf{t_M}$ *to* $\mathcal{A}$*. If not,* $\mathcal{C}$ *runs* $\mathsf{t_M} \leftarrow \mathsf{Td}(\mathsf{ask}, \mathsf{M})$*, stores the pair* $(\mathsf{t_M}, \mathsf{M})$*, and returns* $\mathsf{t_M}$ *to* $\mathcal{A}$*.*

Forge: *Finally,* $\mathcal{A}$ *outputs* $(\mathsf{t_{M^*}}, \mathsf{M}^*)$*. We say that* $\mathcal{A}$ *wins if* $\mathsf{TVf}(\mathsf{apk}, \mathsf{t_{M^*}}, \mathsf{M}^*) = 1$ *and* $(\mathsf{t_{M^*}}, \mathsf{M}^*) \notin \mathsf{TSet_{M^*}}$*.*

We define that GS-MDO *provides public verifiability of tokens if, for all PPT adversaries* $\mathcal{A}$*, the advantage*

$$\mathsf{Adv}^{\text{Public-Verifiability}}_{\text{GS-MDO},\mathcal{A}}(\lambda, N) = \Pr[\mathcal{A} \textit{ wins}]$$

is negligible in the security parameter λ*.*

We clarify that $\mathcal{A}$ is allowed to send M^* as a token query. Let the response be $\mathsf{t}'_{\mathsf{M}^*}$. As mentioned before, $(\mathsf{t}'_{\mathsf{M}^*}, \mathsf{M}^*) \in \mathsf{TSet}_{\mathsf{M}^*}$. Due to the wining condition $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) \notin \mathsf{TSet}_{\mathsf{M}^*}$, $\mathsf{t}'_{\mathsf{M}^*} \neq \mathsf{t}_{\mathsf{M}^*}$ holds.

Next, we examine the kinds of real-world situations that this definition captures. First, we assume the admitter to be an honest-but-curious entity that does not deviate from the prescribed protocol. That is, since the admitter generates tokens using the Td algorithm, any token generated by the admitter always satisfies $(\mathsf{t}_{\mathsf{M}}, \mathsf{M}) \in \mathsf{TSet}_{\mathsf{M}}$. The adversary, while being able to view the tokens generated by the admitter (through oracle queries), outputs a pair $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*)$ that satisfies $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) \notin \mathsf{TSet}_{\mathsf{M}^*}$. That is, the public verifiability of tokens is defined as

> *a security property that guarantees that, even if entities other than the admitter collude, they cannot generate a valid token (i.e., one for which the* TVf *algorithm outputs 1) that is unusable for opening,*

and it is regarded as a security property that protects the admitter.

Note that the adversary $\mathcal{A}$ is assumed not to possess the admitter's secret key ask. Therefore, public verifiability of a token does not guarantee that the admitter can prove that the token was honestly generated according to the Td algorithm. More precisely, if $\mathcal{A}$ is allowed to have ask, then $\mathcal{A}$ might be able to produce a token that breaks public verifiability of tokens. If public verifiability of tokens holds even against an adversary having ask, then the admitter's claim, namely, accountability of the admitter, is guaranteed. Later, we discuss accountability of the admitter in Section 5.

4 Formal Analysis of Existing GS-MDO Schemes

In this section, we prove that the pairing-based GS-MDO scheme by Ohara et al. [44], the lattice-based GS-MDO scheme by Libert et al. [39], and the pairing-based GS-MDO scheme by Libert et al. [38] provide public verifiability of tokens.

4.1 Ohara et al. Pairing-based GS-MDO Scheme

In the following, we describe the Td algorithm (which coincides with the signing algorithm of the BLS signature) and the naturally derived TVf algorithm (which corresponds to the BLS signature-verification algorithm) in the GS-MDO scheme of Ohara et al. Note that the original GS-MDO scheme of Ohara et al. was constructed using symmetric pairings. For implementation efficiency, Arai et al. [4] converted it into a construction using asymmetric pairings via the Abe-Hoshino-Ohkubo transformation [2]. Below, we present the Td and TVf algorithms for the asymmetric-pairing version of the GS-MDO scheme of Ohara et al.

Let p be a λ-bit prime, and let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be groups of order p, where a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is defined. Also, let g_1 and g_2 be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. In this setting, the following properties are satisfied:

Bilinearity: For any $h_1 \in \mathbb{G}_1$, $h_2 \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$, it holds that $e(h_1^a, h_2^b) = e(h_1, h_2)^{ab}$.

Non-degeneracy: $e(g_1, g_2) \neq 1$ holds (where 1 is the identity element of $\mathbb{G}_T$).

Next, we describe the definitions of ask and apk. Both ask and apk are generated by the AKg algorithm. Given the common parameters $\mathsf{pp}_{\mathsf{GS\text{-}MDO}} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e, H)$ as input, it selects ξ uniformly at random from $\mathbb{Z}_p$ and computes $y = g_2^{\xi}$. It then outputs $\mathsf{ask} = \xi$ as the admitter secret key and $\mathsf{apk} = y$ as the admitter public key. Note that $H : \{0,1\}^* \to \mathbb{G}_1$ is a hash function (modeled as a random oracle).

$\mathsf{Td}(\mathsf{ask}, \mathsf{M})$**:** The token generation algorithm takes the admitter secret key $\mathsf{ask} = \xi$ and a message M as input, it outputs a token $\mathsf{t}_{\mathsf{M}} = H(\mathsf{M})^{\xi}$.

$\mathsf{TVf}(\mathsf{apk}, \mathsf{t}_{\mathsf{M}}, \mathsf{M})$**:** The token verification algorithm takes the admitter public key apk, a token t_{M}, and a message M as input, it outputs 1 if $e(\mathsf{t}_{\mathsf{M}}, g_2) = e(H(\mathsf{M}), \mathsf{apk})$ holds, and 0 otherwise.

It holds that $e(\mathsf{t}_{\mathsf{M}}, g_2) = e(H(\mathsf{M})^{\xi}, g_2) = e(H(\mathsf{M}), g_2^{\xi}) = e(H(\mathsf{M}), \mathsf{apk})$. Note that it is also possible to set $\mathsf{apk} = g_1^{\xi} \in \mathbb{G}_1$, and $H : \{0,1\}^* \to \mathbb{G}_2$, with $\mathsf{t}_{\mathsf{M}} = H(\mathsf{M})^{\xi} \in \mathbb{G}_2$. In this case, the TVf algorithm verifies whether $e(g_1, \mathsf{t}_{\mathsf{M}}) = e(\mathsf{apk}, H(\mathsf{M}))$ holds. Note that for $\mathbb{G}_2$, an elliptic curve defined over an extension field is typically used (e.g., BN (Barreto-Naehrig) curves [6] and BLS (Barreto-Lynn-Scott) curves [5]). Elements of $\mathbb{G}_1$ are smaller in size than those of $\mathbb{G}_2$, and group operations in $\mathbb{G}_1$ are more efficient than in $\mathbb{G}_2$. Moreover, while apk is generated only once, a token must be generated for multiple messages. Taking these considerations into account, we set tokens to be elements of $\mathbb{G}_1$.

Theorem 1. *The GS-MDO scheme by Ohara et al. provides public verifiability of tokens.*

Proof. Tokens in the GS-MDO scheme by Ohara et al. correspond to BLS signatures. Since tokens included in $\mathsf{TSet}_{\mathsf{M}}$ are signatures generated by the BLS signature algorithm, it is clear from the correctness of the BLS signature scheme that $\mathsf{TVf}(\mathsf{apk}, \mathsf{t}_{\mathsf{M}}, \mathsf{M}) = 1$ always holds. Therefore, we prove that if $\mathsf{TVf}(\mathsf{apk}, \mathsf{t}_{\mathsf{M}}, \mathsf{M}) = 1$, then $(\mathsf{t}_{\mathsf{M}}, \mathsf{M}) \in \mathsf{TSet}_{\mathsf{M}}$. The BLS signature scheme is regarded as an FDH signature scheme and does not use randomness for signature generation. That is, once $\mathsf{pp}_{\mathsf{GS\text{-}MDO}} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g_1, g_2, e, H)$ and $\mathsf{apk} = g_2^{\xi}$ are fixed, the token $\mathsf{t}_{\mathsf{M}} = H(\mathsf{M})^{\xi}$ for a certain message M is uniquely determined. Consequently, for $(\mathsf{t}_{\mathsf{M}}, \mathsf{M}) \notin \mathsf{TSet}_{\mathsf{M}}$, no valid BLS signature exists. Thus, the GS-MDO scheme by Ohara et al. satisfies the public verifiability of tokens. □

As mentioned in the proof, no valid BLS signature exists where $(\mathsf{t}_{\mathsf{M}}, \mathsf{M}) \notin \mathsf{TSet}_{\mathsf{M}}$. This implies that even if $\mathcal{A}$ has $\mathsf{ask} = \xi$, $\mathcal{A}$ has no way to produce $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*)$ satisfying both $\mathsf{TVf}(\mathsf{apk}, \mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) = 1$ and $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) \notin \mathsf{TSet}_{\mathsf{M}^*}$. This property will be revisited later when we discuss the accountability of the admitter.

4.2 Libert et al. Lattice-based GS-MDO Scheme

Tokens in the GS-MDO scheme of Libert et al. [39] correspond to GPV signatures. Similar to BLS signatures, GPV signatures can be viewed as FDH signatures and do not use randomness in the signature-generation process. More precisely, a derandomization step is performed using a pseudorandom function (PRF): by including a PRF key in the signing key and using the PRF output on the message as the randomness for signing, the signing process becomes deterministic. Since the GPV signature scheme is analyzed in the random oracle model, the random oracle can be used as the PRF. In the following, we omit the PRF for simplicity.

We describe the Td and TVf algorithms for the GS-MDO scheme by Libert et al. Let $n = O(\lambda)$ be a lattice parameter and $m \leq \mathsf{poly}(n)$. We define $\mathsf{apk} := \mathbf{C} \in \mathbb{Z}_q^{n \times m}$ as the master public key of GPV-IBE, $\mathsf{ask} := \mathbf{T}_{\mathbf{C}} \in \mathbb{Z}^{m \times m}$ as the master secret key (GPV trapdoor) of GPV-IBE,

and $N_{\mathsf{max}} = 2^\ell = \mathsf{poly}(\lambda)$ as the maximum number of group members in the GS-MDO scheme. Furthermore, let $H : \{0,1\} \to \mathbb{Z}_q^{n \times \ell \lceil \log q \rceil}$ be a hash function (modeled as a random oracle). We assume that H is included in $\mathsf{pp}_{\mathsf{GS\text{-}MDO}}$ and is implicitly included in the input of each algorithm. Note that in the GS-MDO scheme by Libert et al., the token is described as the decryption key of GPV-IBE, which indicates that the token is a GPV signature. In this context, $\mathbf{C}$ and $\mathbf{T_C}$ correspond to the verification key and the signing key of the GPV signature, respectively.

$\mathsf{Td}(\mathsf{ask}, \mathsf{M})$**:** The token generation algorithm takes the admitter secret key $\mathsf{ask} = \mathbf{T_C}$ and a message M as input, computes $\hat{\mathbf{G}} = H(\mathsf{M})$, and using $\mathbf{T_C}$, computes a small-norm matrix $\mathbf{E}_\mathsf{M}$ such that $\mathbf{C} \cdot \mathbf{E}_M = \hat{\mathbf{G}}$ holds, and it outputs a token $\mathsf{t}_\mathsf{M} = \mathbf{E}_\mathsf{M}$.

$\mathsf{TVf}(\mathsf{apk}, \mathsf{t}_\mathsf{M}, \mathsf{M})$**:** The token verification algorithm takes the admitter public key $\mathsf{apk} = \mathbf{C}$, a token $\mathsf{t}_\mathsf{M} = \mathbf{E}_\mathsf{M}$, and a message M as input, it outputs $\mathbf{C} \cdot \mathbf{E}_M = \hat{\mathbf{G}}$ holds and the norm of $\mathbf{E}_\mathsf{M}$ is small, and outputs 0 otherwise.

Theorem 2. *If the GPV signature scheme provides strong existential unforgeability, the Libert et al. GS-MDO scheme provides public verifiability of tokens.*

Outline of the Security Proof. In the GPV paper [27], a Preimage Sampleable (trapdoor) Function (PSF) is first defined, and then it is shown that the signature scheme using PSF is strongly existentially unforgeable[3]. After providing the proof for this, they propose a construction of PSF from lattices. In the proof of strong existential unforgeability (Proposition 6.1 in [27]), consider the message M^* queried by the adversary to the signing oracle, the oracle's response σ, and the forged signature σ^*, output by the adversary. It is shown that the probability that $\sigma \neq \sigma^*$ is negligible, due to the Preimage Min-entropy Property of the PSF. That is, although the probability that an adversary outputs such a signature is negligible, it can be said that there exist valid signatures on M not included in TSet_M. Here, it should be noted that once the verification key $\mathbf{C} \in \mathbb{Z}_q^{n \times m}$, the signing key $\mathbf{T_C} \in \mathbb{Z}^{m \times m}$ (and the PRF key for the derandomization process) of the GPV signature are fixed, $\mathsf{TSet}_{\mathsf{M}^*}$ is uniquely determined. This does not depend on whether the adversary has already sent M^* as a token query for the pair $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*)$ output by the adversary in the game of public verifiability of tokens. If the adversary has sent M^* as a token query, the token included in $\mathsf{TSet}_{\mathsf{M}^*}$ is returned. By definition, since $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) \notin \mathsf{TSet}_{\mathsf{M}^*}$, it is possible to construct a reduction algorithm that breaks the strong existential unforgeability of the GPV signature scheme.

Proof. Let $\mathcal{A}$ be an adversary against the public verifiability of tokens, and let $\mathcal{C}$ be a challenger for strong existential unforgeability. We construct a reduction algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to break the strong existential unforgeability of the GPV signature scheme.

First, $\mathcal{C}$ generates a verification key $\mathbf{C} \in \mathbb{Z}_q^{n \times m}$ and a signing key $\mathbf{T_C} \in \mathbb{Z}^{m \times m}$, and sends $\mathbf{C}$ to $\mathcal{B}$. Then, $\mathcal{B}$ sets $\mathsf{apk} := \mathbf{C}$. Additionally, $\mathcal{B}$ generates all values of the GS-MDO scheme other than apk, namely $(\mathsf{gpk}, \mathsf{ok}, \{\mathsf{gsk}_i\}_{i \in [N]})$. It should be noted that $\mathcal{B}$ can generate these values by itself because they are independent of $(\mathsf{apk}, \mathsf{ask})$. $\mathcal{B}$ then sends $(\mathsf{gpk}, \mathsf{ok}, \mathsf{apk}, \{\mathsf{gsk}_i\}_{i \in [N]})$ to $\mathcal{A}$.

Upon receiving a token query M from $\mathcal{A}$, $\mathcal{B}$ sends M as a signing query to $\mathcal{C}$. $\mathcal{C}$ generates a GPV signature $\mathbf{E}_\mathsf{M}$ and sends it to $\mathcal{B}$. $\mathcal{B}$ sets $\mathsf{t}_\mathsf{M} = \mathbf{E}_\mathsf{M}$ and returns t_M to $\mathcal{A}$. Note that if $\mathcal{A}$ sends a message M that has been sent previously, $\mathcal{B}$ simply returns the previous response.

[3]The challenger generates a pair consisting of a verification key and a signing key, and provides the verification key to the adversary. The adversary sends a message M to the signing oracle and obtains a signature σ. Let $\{(\sigma, \mathsf{M})\}$ denote the set of these message-signature pairs. The adversary outputs a forgery (σ^*, M^*), and is defined to win if σ^* is a valid signature for M^* and $(\sigma^*, \mathsf{M}^*) \notin \{(\sigma, \mathsf{M})\}$. If the probability that all PPT adversaries win is negligible with respect to the security parameter, the signature scheme is defined to be strongly existentially unforgeable.

Finally, $\mathcal{A}$ outputs $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*)$. Let $\mathsf{t}_{\mathsf{M}^*} = \mathbf{E}^*_{\mathsf{M}^*}$. Since $\mathsf{t}_{\mathsf{M}^*}$ is a valid signature on M^* and $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) \notin \mathsf{TSet}_{\mathsf{M}^*}$ by definition, $\mathcal{B}$ outputs $(\mathbf{E}^*_{\mathsf{M}^*}, \mathsf{M}^*)$, thereby breaking the strong existential unforgeability.

Due to the difference between the BLS signature scheme and the GPV signature scheme, there exists valid tokens where $(\mathsf{t}_{\mathsf{M}}, \mathsf{M}) \notin \mathsf{TSet}_{\mathsf{M}}$, unlike to the Ohara et al. GS-MDO scheme. This implies that if $\mathcal{A}$ has $\mathsf{ask} = \mathbf{T_C}$, $\mathcal{A}$ may be able to produce $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*)$ satisfying both $\mathsf{TVf}(\mathsf{apk}, \mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) = 1$ and $(\mathsf{t}_{\mathsf{M}^*}, \mathsf{M}^*) \notin \mathsf{TSet}_{\mathsf{M}^*}$. We emphasize that this does not contradict the strong existential unforgeability of the GPV signature scheme, since the adversary in the strong existential unforgeability game does not possess the signing key. This property will be revisited later when we discuss the accountability of the admitter.

4.3 Libert et al. Pairing-based GS-MDO Scheme

Tokens of the two GS-MDO schemes investigated in previous sections are regarded as FDH signatures. Here, we explore another type of tokens, concretely, a token consists of two Waters signatures [29,48] in the pairing-based GS-MDO scheme by Libert et al. [38]. Each $\mathsf{TSet}_{\mathsf{M}}$ contains more than one (exponentially-many) signatures since the signing algorithm of the Waters signature scheme is probabilistic. Moreover, the Waters signature scheme is not strongly unforgeable since it can be re-randomized. Due to this situation, we need to consider another methodology for proving that the scheme provides public verifiability of tokens.

Libert et al. employed symmetric pairings defined as follows. Let p be a λ-bit prime, and let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of order p, where a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is defined, and $g \in \mathbb{G}$ be a generator. $X \xleftarrow{R} S$ denotes that an element X is chosen uniformly at random from a set S. We assume that the Computational Diffie-Hellman (CDH) assumption over $\mathbb{G}$ holds, for $a, b \xleftarrow{R} \mathbb{Z}_p$, i.e., it is infeasible to compute g^{ab} from $(\mathbb{G}, \mathbb{G}_T, p, g, e, g^a, g^b)$. As in the asymmetric pairing defined in Section 4.1, bilineality and non-degeneracy are assumed to be provided. We describe the definitions of ask and apk. Both ask and apk are generated by the AKg algorithm. Let $L \in \mathrm{poly}(\lambda)$ be the bit-length of a message to be signed M. Given the common parameters $\mathsf{pp}_{\mathsf{GS\text{-}MDO}} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$ as input, the AKg algorithm chooses $\alpha_0, \alpha_1 \xleftarrow{R} \mathbb{Z}_p$, $g_2 \xleftarrow{R} \mathbb{G}$, and $u_0, u_1, \ldots, u_L \xleftarrow{R} \mathbb{G}$, and sets $\mathsf{ask} = (g_2^{\alpha_0}, g_2^{\alpha_1})$, $\mathsf{apk} = (g_0 = g^{\alpha_0}, g_1 = g^{\alpha_1}, g_2, \{u_i\}_{i=0}^L)$. For a message $\mathsf{M} \in \{0,1\}^L$, we define the function $H_U(M) \in \mathbb{G}$ as $H_U(M) = u_0 \cdot \prod_{i=1}^L u_i^{M[i]}$, where $M[i] \in \{0,1\}$ denotes the i-th bit of M. We describe the Td and TVf algorithms for the pairing-based GS-MDO scheme by Libert et al. as follows.

$\mathsf{Td}(\mathsf{ask}, \mathsf{M})$**:** The token generation algorithm takes the admitter secret key $\mathsf{ask} = (g_2^{\alpha_0}, g_2^{\alpha_1})$ and a message $M \in \{0,1\}^L$ as input. It chooses $r_0, r_1 \xleftarrow{R} \mathbb{Z}_p$, where $r_0 \neq r_1$, to compute and return a token :

$$\mathsf{t}_{\mathsf{M}} = (t_{0,1}, t_{0,2}, t_{1,1}, t_{1,2}) = (g_2^{\alpha_0} \cdot H_U(\mathsf{M})^{r_0}, g^{r_0}, g_2^{\alpha_1} \cdot H_U(\mathsf{M})^{r_1}, g^{r_1}).$$

$\mathsf{TVf}(\mathsf{apk}, \mathsf{t}_{\mathsf{M}}, \mathsf{M})$**:** The token verification algorithm takes the admitter public key $\mathsf{apk} = (g_0, g_1, g_2, \{u_i\}_{i=0}^L)$, a token t_M, and a message M as input. It outputs 1 if

$$e(t_{0,1}, g) = e(g_2, g_0)e(H_U(\mathsf{M}), t_{0,2}) \quad \text{and} \quad e(t_{1,1}, g) = e(g_2, g_1)e(H_U(\mathsf{M}), t_{1,2})$$

holds, and 0 otherwise.

t_{M} consists of two Waters signatures, $(t_{0,1}, t_{0,2})$ and $(t_{1,1}, t_{1,2})$. If they are honestly generated, then

$$e(t_{0,1}, g) = e(g_2^{\alpha_0}, g)e(H_U(\mathsf{M})^{r_0}, g) = e(g_2, g_0)e(H_U(\mathsf{M}), t_{0,2})$$
$$e(t_{1,1}, g) = e(g_2^{\alpha_1}, g)e(H_U(\mathsf{M})^{r_1}, g) = e(g_2, g_1)e(H_U(\mathsf{M}), t_{1,2})$$

hold. That is, for honestly generated $\mathsf{t_M} \leftarrow \mathsf{Td}(\mathsf{ask}, \mathsf{M})$, i.e., for $(\mathsf{t_M}, \mathsf{M}) \in \mathsf{TSet_M}$, $\mathsf{TVf}(\mathsf{apk}, \mathsf{t_M}, \mathsf{M}) = 1$ holds.

We remark that Libert et al. described $r_0, r_1 \xleftarrow{R} \mathbb{Z}_p$. That is, there is a (negligible) probability that $r_0 = r_1$ holds. We clarify that the condition $r_0 \neq r_1$ is implicitly assumed in their security proof (Lemma 1 in [38]). We explicitly describe the condition $r_0 \neq r_1$ in the Td algorithm whereas the TVf algorithm intentionally does not check $r_0 \neq r_1$ (which can be checked by $t_{0,2} \neq t_{1,2}$). More precisely, for $(\mathsf{t_M}, \mathsf{M}) \in \mathsf{TSet_M}$ where $\mathsf{t_M} = (t_{0,1}, t_{0,2}, t_{1,1}, t_{1,2})$, $t_{0,2} \neq t_{1,2}$ holds. We will explain the reason that the TVf algorithm intentionally does not check $t_{0,2} \neq t_{1,2}$ later.

Theorem 3. *The Libert et al. pairing-based GS-MDO scheme provides public verifiability of tokens under the CDH assumption in* $\mathbb{G}$.

Outline of the Security Proof. Let $(\mathsf{M}^*, \mathsf{t_{M^*}})$ be the output of $\mathcal{A}$ where $\mathsf{t_{M^*}} = (t^*_{0,1}, t^*_{0,2}, t^*_{1,1}, t^*_{1,2})$. Due to the wining condition, $\mathsf{TVf}(\mathsf{apk}, \mathsf{t_{M^*}}, \mathsf{M}^*) = 1$ and $(\mathsf{M}^*, \mathsf{t_{M^*}}) \notin \mathsf{TSet_{M^*}}$ holds. Here, $\mathsf{TVf}(\mathsf{apk}, \mathsf{t_{M^*}}, \mathsf{M}^*) = 1$ if $(t^*_{0,1}, t^*_{0,2})$ and $(t^*_{1,1}, t^*_{1,2})$ are valid Waters signatures on M regardless of $t^*_{0,2} = t^*_{1,2}$ or not. Moreover, due to the pairing properties in the Waters signature verification algorithm, values satisfying the verification equation are restricted to the form of Waters signatures.[4] Consequently, a token $\mathsf{t_M}$ that passes the TVf algorithm but is not contained in $\mathsf{TSet_M}$ occurs only when $r_0 = r_1$ because the TVf algorithm does not check $t_{0,2} \neq t_{1,2}$. If $r_0 = r_1$, then anyone can extract the values $g_2^{\alpha_0 - \alpha_1}$ by computing $t_{0,1}/t_{1,1}$. From this, we can extract the CDH solution.

Proof. Let $\mathcal{A}$ be an adversary against the public verifiability of tokens. We construct an algorithm $\mathcal{B}$ that solves the CDH problem using $\mathcal{A}$. First, $\mathcal{B}$ takes in an instance (g, g^a, g^b) of the CDH problem and prepares the admitter public key apk by randomly choosing $\gamma_0, \gamma_1 \xleftarrow{R} \mathbb{Z}_p$, $\nu \xleftarrow{R} \{0, \ldots, L\}$, $\rho_0, \rho_1, \ldots, \rho_L \xleftarrow{R} \{0, \ldots, \zeta - 1\}$, and $\delta_0, \delta_1, \ldots, \delta_L \xleftarrow{R} \mathbb{Z}_p$, where $\zeta = 2q$ and q is the maximal number of token queries throughout the game. These are used to define $u_0 = g^{\delta_0} \cdot (g^b)^{\nu \cdot \zeta - \rho_0}$ and $u_i = g^{\delta_i} \cdot (g^b)^{-\rho_i}$ for all $i \in \{1, \ldots, L\}$, and set $g_0 = (g^a)^{\gamma_0}, g_1 = (g^a)^{\gamma_1}$ as well as $g_2 = g^b$. Note that this implicitly defines $\alpha_0 = a \cdot \gamma_0$ and $\alpha_1 = a \cdot \gamma_1$. $\mathcal{B}$ sets $\mathsf{apk} = ((g^a)^{\gamma_0}, (g^a)^{\gamma_1}, g^b, \{u_i\}_{i=0}^L)$. Additionally, $\mathcal{B}$ generates all values of the GS-MDO scheme other than apk, namely $(\mathsf{gpk}, \mathsf{ok}, \{\mathsf{gsk}_i\}_{i \in [N]})$. We note that these values can be computed without using the CDH instance, except the generator g. $\mathcal{B}$ then sends $(\mathsf{gpk}, \mathsf{ok}, \mathsf{apk}, \{\mathsf{gsk}_i\}_{i \in [N]})$ to $\mathcal{A}$.

Upon receiving a token query M from $\mathcal{A}$. Before handling queries, we define the hash function $H_U(\mathsf{M}) = u_0 \cdot \prod_{i=1}^{L} u_i^{\mathsf{M}[i]}$. Note that $H_U(\mathsf{M})$ can be written as $g^{J_2(\mathsf{M})} \cdot (g^b)^{J_1(\mathsf{M})}$, where $J_1(\mathsf{M}) = \nu \cdot \zeta - \rho_0 - \sum_{i=1}^{L} \rho_i \cdot \mathsf{M}[i]$ and $J_2(\mathsf{M}) = \delta_0 - \sum_{i=1}^{L} \delta_i \cdot \mathsf{M}[i]$. Here, the construction of $\mathsf{t_M}$ by $\mathcal{B}$ depends on the value of $J_1(\mathsf{M})$. If $J_1(\mathsf{M}) \neq 0$, $\mathcal{B}$ can calculate $\mathsf{t_M}$ as follows. $\mathcal{B}$ chooses $\tilde{r}_0, \tilde{r}_1 \xleftarrow{R} \mathbb{Z}_p$ and implicitly sets $r_0 = \tilde{r}_0 - \gamma_0 \cdot a / J_1(\mathsf{M})$ and $r_1 = \tilde{r}_1 - \gamma_1 \cdot a / J_1(\mathsf{M})$. Here,

$$
\begin{aligned}
& g_2^{\alpha_0} \cdot H_U(\mathsf{M})^{r_0} \\
=& (g^b)^{\gamma_0 \cdot a} (g^{J_2(\mathsf{M})} (g^b)^{J_1(\mathsf{M})})^{\tilde{r}_0 - \gamma_0 \cdot a / J_1(\mathsf{M})} \\
=& (g^{J_2(\mathsf{M})})^{\tilde{r}_0} (g^{J_2(\mathsf{M})})^{-\gamma_0 \cdot a / J_1(\mathsf{M})} (g^b)^{J_1(\mathsf{M}) \cdot \tilde{r}_0} \\
=& (g^{J_2(\mathsf{M})} (g^b)^{J_1(\mathsf{M})})^{\tilde{r}_0} (g^a)^{-\gamma_0 \cdot J_2(\mathsf{M}) / J_1(\mathsf{M})} \\
=& H_U(\mathsf{M})^{\tilde{r}_0} (g^a)^{-\gamma_0 \cdot J_2(\mathsf{M}) / J_1(\mathsf{M})}
\end{aligned}
$$

[4] Let $\mathsf{t_M} = (t_{0,1}, t_{0,2}, t_{1,1}, t_{1,2})$ satisfy $\mathsf{TVf}(\mathsf{apk}, \mathsf{t_M}, \mathsf{M}) = 1$, i.e., $e(t_{0,1}, g) = e(g_2, g_0) e(H_U(\mathsf{M}), t_{0,2})$ and $e(t_{1,1}, g) = e(g_2, g_1) e(H_U(\mathsf{M}), t_{1,2})$ hold. Let $r_0 := \log_g t_{0,2}$ and $r_1 := \log_g t_{1,2}$. Then, $e(g_2, g_0) e(H_U(\mathsf{M}), t_{0,2}) = e(g_2^{\alpha_0}, g) e(H_U(\mathsf{M})^{r_0}, g) = e(g_2^{\alpha_0} H_U(\mathsf{M})^{r_0}, g)$ and $e(g_2, g_1) e(H_U(\mathsf{M}), t_{1,2}) = e(g_2^{\alpha_1}, g) e(H_U(\mathsf{M})^{r_1}, g) = e(g_2^{\alpha_1} H_U(\mathsf{M})^{r_1}, g)$ hold. Thus, $t_{0,1} = g_2^{\alpha_0} H_U(\mathsf{M})^{r_0}$ and $t_{1,1} = g_2^{\alpha_1} H_U(\mathsf{M})^{r_1}$ hold.

and

$$g^{r_0} = g^{\tilde{r}_0 - \gamma_0 \cdot a/J_1(\mathsf{M})} = g^{\tilde{r}_0} \cdot (g^a)^{-\gamma_0/J_1(\mathsf{M})}$$

hold. Similarly, $g_2^{\alpha_1} \cdot H_U(\mathsf{M})^{r_1} = H_U(\mathsf{M})^{\tilde{r}_1}(g^a)^{-\gamma_1 \cdot J_2(\mathsf{M})/J_1(\mathsf{M})}$ and $g^{r_1} = g^{\tilde{r}_1} \cdot (g^a)^{-\gamma_1/J_1(\mathsf{M})}$ hold. $\mathcal{B}$ computes $\mathsf{t_M} = (H_U(\mathsf{M})^{\tilde{r}_0}(g^a)^{-\gamma_0 \cdot J_2(\mathsf{M})/J_1(\mathsf{M})}, g^{\tilde{r}_0} \cdot (g^a)^{-\gamma_0/J_1(\mathsf{M})}, H_U(\mathsf{M})^{\tilde{r}_1}(g^a)^{-\gamma_1 \cdot J_2(\mathsf{M})/J_1(\mathsf{M})}, g^{\tilde{r}_1} \cdot (g^a)^{-\gamma_1/J_1(\mathsf{M})})$ and sends $\mathsf{t_M}$ to $\mathcal{A}$. As mentioned in [48], the probability that $J_1(\mathsf{M}) \neq 0$ holds is non-negligible.

Finally, $\mathcal{A}$ outputs $(\mathsf{t_{M^*}}, \mathsf{M}^*)$ where $\mathsf{t_{M^*}} = (t^*_{0,1}, t^*_{0,2}, t^*_{1,1}, t^*_{1,2})$ such that $t^*_{0,2} = t^*_{1,2}$. We remark that we do not have to care whether $J_1(\mathsf{M}^*) = 0$ or not. More precisely, to prove the security of an IBE scheme, it needs to guarantee that $J_1(ID) \neq 0$ for all queried ID and $J_1(ID^*) = 0$ for generating the challenge ciphertext for the challenge identity ID^* (this also means that the simulator is prevented from generating a key that allows it to decrypt the challenge ciphertext itself). In our proof, we only need to consider whether $\mathsf{t_M}$ can be constructed (that is, whether a decryption key can be constructed in the IBE context), and no challenge ciphertext needs to be generated.

By calculating $(t^*_{0,1}/t^*_{1,1})^{1/(\gamma_0 - \gamma_1)}$, we obtain

$$(g_2^{\alpha_0 - \alpha_1})^{1/(\gamma_0 - \gamma_1)} = g^{ab}$$

which allows $\mathcal{B}$ to break the CDH problem.

□

5 Discussion

We conclude that the public verifiability of tokens introduced in this paper is a security property ensuring that even if all entities other than the admitter collude, they cannot generate a valid token (i.e,, the TVf algorithm outputs 1) that cannot be used for opening. This property therefore serves as a security guarantee that protects the admitter. In this section, we examine whether alternative perspectives or interpretations of this property are possible.

5.1 Admitter Accountability

We consider the situation in which the admitter wishes to convince a third party that a token they generated was produced honestly according to the token generation algorithm Td. We regard this as a subsequent step beyond the basic notion considered in Definition 4. If such a claim can be validated through public verifiability of tokens, then public verifiability can be regarded as being related to accountability of the admitter. Here, it should be noted that the adversary in the public verifiability game does not possess the admitter's secret key ask. In other words, an admitter who holds ask may have the ability to break public verifiability of tokens. In such a case, the admitter cannot be said to satisfy accountability. In the case of BLS signatures, however, there exists only one valid signature for each message. Thus, even if one possesses the signing key ξ, it is impossible to produce any valid signature for a message M other than $\sigma = H(\mathsf{M})^{\xi}$. Consequently, since the Ohara et al. GS-MDO scheme satisfies public verifiability of tokens even against an adversary holding ask, it can be said to achieve the accountability of the admitter. Formally, we say that $\mathsf{GS\text{-}MDO}$ provides admitter accountability if the public verifiability of tokens holds against any adversary $\mathcal{A}$ in the corresponding security game, where $\mathcal{A}$ is additionally given ask along with $(\mathsf{gpk}, \mathsf{ok}, \mathsf{apk}, \{\mathsf{gsk}_i\}_{i \in [N]})$. In the case of GPV signatures, although there exists only one valid

signature for each message due to the derandomization process, an adversary who possesses the signing key may be able to produce other valid signature, as mentioned in Section 4.2. Thus, the Libert et al. lattice-based GS-MDO scheme does not provide admitter accountability. In the case of the Waters signatures, there are exponentially many valid signatures on a message. Moreover, if an adversary has the signing key (ask in the GS-MDO context), the adversary can produce a valid token $\mathsf{t_M}$ but $(\mathsf{t_M}, \mathsf{M}) \notin \mathsf{TSet_M}$ by using the same randomness for generating two Waters signatures. Thus, the Libert et al. pairing-based GS-MDO scheme does not provide admitter accountability.

Regarding accountability of the opener in group signatures, a publicly verifiable Judge algorithm [8] is typically introduced to provide a proof of the opening result, and such an algorithm is also incorporated into GS-MDO in the Chen et al. GS-MDO scheme [20]. A typical approach is to encrypt the user ID using a public-key encryption scheme, and the opener, who holds the corresponding decryption key ok, proves that decrypting the ciphertext indeed yields the correct user ID without revealing ok. This functionality is related to PKENO [21,26], verifiable decryption [18], and accountable decryption [37]. As a direct connection between GS and PKENO, if a GS scheme secure in the Bellare-Shi-Zhang (BSZ) model [8] also provides opening soundness [47], then it implies PKENO [25]. Furthermore, various studies have investigated accountability in anonymous signature settings, e.g., [11, 17, 33–36, 40–43], or other contexts, e.g., [14–16]. However, to the best of our knowledge, accountability of the admitter has not previously been discussed. The fact that the Ohara et al. GS-MDO scheme implicitly satisfies admitter accountability is, in our view, an interesting observation.

5.2 Revisiting the Security of GS-MDO

Here, we focus again on the fact that in GS-MDO, the opener alone cannot perform the opening of a group signature. If a token is not generated legitimately, the opening will fail regardless of whether the opener performs the procedure correctly. In the conventional GS-MDO framework, the opener has no means to prove this fact. In the Judge algorithm of Chen et al.'s scheme, similar to that of standard group signatures, it is proven that the opener performed the opening correctly using ok, and the token is treated as a witness for a zero-knowledge proof. In other words, it is implicitly assumed that the token has been generated legitimately. By treating the token as a public statement and utilizing the public verifiability of tokens introduced in this paper, it would become possible for the opener to prove the fact when a token is invalid. However, it should be noted that in this scenario, since the admitter is assumed to be the adversary, we must consider a case where the adversary against public verifiability of tokens possesses ask. Fortunately, we have discussed that the Ohara et al. GS-MDO scheme satisfies public verifiability of tokens in the stronger sense.

Furthermore, there may be room for reconsidering traceability. In the definition of traceability, the adversary's output consists of a message and a group signature, and the adversary wins if they are untraceable using a token generated according to the Td algorithm and ok. That is, cases involving tokens not generated according to the Td algorithm are not covered. By employing public verifiability of tokens, it may be possible to capture scenarios where the adversary generates the token. Given that the adversary possesses ask in the definition of traceability, we conjecture that public verifiability of tokens would be required to hold even in such a case. We leave these detailed examinations as future work.

6 Conclusion

In this paper, we formalized public verifiability of tokens in GS-MDO and demonstrated that the schemes proposed by Ohara et al. [44] and Libert et al. [38, 39] satisfy our definition, suggesting

that our formalization is reasonable.

In the construction of other cryptographic primitives from GS-MDO, it has been observed that only opener anonymity is utilized for IBE, while both opener anonymity and admitter anonymity are required for TRE. By leveraging public verifiability of tokens in addition to traceability, it may be possible to construct even more advanced cryptographic primitives. Conversely, exploring whether the efficiency of GS-MDO can be enhanced when public verifiability of tokens is not required is another interesting direction. We leave these detailed investigations for future research.

Acknowledgment: This work was supported by the Telecommunications Advancement Foundation and JSPS KAKENHI Grant Number JP25H01106.

References

[1] Michel Abdalla and Bogdan Warinschi. On the minimal assumptions of group signature schemes. In *ICICS*, pages 1–13, 2004.

[2] Masayuki Abe, Fumitaka Hoshino, and Miyako Ohkubo. Design in type-I, run in type-III: Fast and scalable bilinear-type conversion using integer programming. In *CRYPTO, Part III*, pages 387–415, 2016.

[3] Hiroaki Anada, Masayuki Fukumitsu, and Shingo Hasegawa. Dynamic group signatures with message dependent opening and non-interactive signing. In *CANDAR*, pages 76–82. IEEE, 2022.

[4] Hiromi Arai, Keita Emura, and Takuya Hayashi. Privacy-preserving data analysis: Providing traceability without big brother. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 104-A(1):2–19, 2021.

[5] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In *Security in Communication Networks*, pages 257–267, 2002.

[6] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography*, pages 319–331, 2005.

[7] 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.

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

[9] Olivier Blazy and Lola-Baie Mallordy. Threshold reporting protocol for traceability in anonymous social networks. *IACR Cryptol. ePrint Arch.*, 2025:1873, 2025.

[10] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, pages 213–229, 2001.

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

[12] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *ASIACRYPT*, pages 514–532, 2001.

[13] Dan Boneh, Periklis A. Papakonstantinou, Charles Rackoff, Yevgeniy Vahlis, and Brent Waters. On the impossibility of basing identity based encryption on trapdoor permutations. In *FOCS*, pages 283–292. IEEE Computer Society, 2008.

[14] Dan Boneh, Aditi Partap, and Lior Rotem. Accountability for misbehavior in threshold decryption via threshold traitor tracing. In *CRYPTO, Part VII*, pages 317–351, 2024.

[15] Dan Boneh, Aditi Partap, and Lior Rotem. Proactive refresh for accountable threshold signatures. In *Financial Cryptography and Data Security - 28th International Conference, Part II*, pages 140–159, 2024.

[16] 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.

[17] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. Short accountable ring signatures based on DDH. In *ESORICS Part I*, pages 243–265, 2015.

[18] Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *CRYPTO*, pages 126–144, 2003.

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

[20] Simin Chen, Jiageng Chen, Atsuko Miyaji, and Kaiming Chen. Constant-size group signatures with message-dependent opening from lattices. In *ProvSec*, pages 166–185, 2023.

[21] Ivan Damgård, Dennis Hofheinz, Eike Kiltz, and Rune Thorbek. Public-key encryption with non-interactive opening. In *CT-RSA*, pages 239–255, 2008.

[22] Cécile Delerablée and David Pointcheval. Dynamic threshold public-key encryption. In David A. Wagner, editor, *CRYPTO*, 2008.

[23] Nico Döttling and Sanjam Garg. From selective IBE to full IBE and selective HIBE. In *TCC Part I*, pages 372–408, 2017.

[24] Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In *CRYPTO Part I*, pages 537–569, 2017.

[25] Keita Emura, Goichiro Hanaoka, Yusuke Sakai, and Jacob C. N. Schuldt. Group signature implies public-key encryption with non-interactive opening. *International Journal of Information Security*, 13(1):51–62, 2014.

[26] David Galindo, Benoît Libert, Marc Fischlin, Georg Fuchsbauer, Anja Lehmann, Mark Manulis, and Dominique Schröder. Public-key encryption with non-interactive opening: New constructions and stronger definitions. In *AFRICACRYPT*, pages 333–350, 2010.

[27] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *ACM STOC*, pages 197–206, 2008.

[28] Swee-Huay Heng and Kaoru Kurosawa. k-resilient identity-based encryption in the standard model. In *CT-RSA*, pages 67–80, 2004.

[29] Dennis Hofheinz, Tibor Jager, and Edward Knapp. Waters signatures with optimal security reduction. In *Public Key Cryptography*, pages 66–83, 2012.

[30] Yuto Imura and Keita Emura. An identity management system using group signatures with message-dependent opening. In *AsiaJCIS*, pages 40–47. IEEE, 2024.

[31] Yuto Imura and Keita Emura. Group signatures with message-dependent opening directly imply timed-release encryption. In *CANS*, pages 251–271, 2025.

[32] Yuto Imura and Keita Emura. Group signatures with message-dependent opening directly imply timed-release encryption. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E109-A(9):1446–1457, 2026.

[33] Thanh Xuan Khuc, Willy Susilo, Dung Hoang Duong, Fuchun Guo, Kazuhide Fukushima, and Shinsaku Kiyomoto. Threshold ring signatures with accountability. In *ACISP, Part I*, pages 368–388, 2024.

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

[35] Meng Li, Hanni Ding, Yifei Chen, Yan Qiao, Zijian Zhang, Liehuang Zhu, and Mauro Conti. Threshold signatures with verifiably timed combining and message-dependent tracing. *IEEE Transactions on Information Forensics and Security*, 20:9477–9491, 2025.

[36] Meng Li, Hanni Ding, Qing Wang, Zijian Zhang, and Mauro Conti. Threshold signatures with private accountability via secretly designated witnesses. In *ACISP, Part I*, pages 389–407, 2024.

[37] Rujia Li, Yuanzhao Li, Qin Wang, Sisi Duan, Qi Wang, and Mark Ryan. Accountable decryption made formal and practical. *IEEE Transactions on Information Forensics and Security*, 20:620–635, 2025.

[38] Benoît Libert and Marc Joye. Group signatures with message-dependent opening in the standard model. In *CT-RSA*, pages 286–306, 2014.

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

[40] Benoît Libert, Khoa Nguyen, Thomas Peters, and Moti Yung. Bifurcated signatures: Folding the accountability vs. anonymity dilemma into a single private signing scheme. In *EUROCRYPT*, pages 521–552, 2021.

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

[42] 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.

[43] Khoa Nguyen, Fuchun Guo, Willy Susilo, and Guomin Yang. Multimodal private signatures. In *CRYPTO*, pages 792–822, 2022.

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

[45] 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.

[46] Yusuke Sakai, Keita Emura, Jacob C. N. Schuldt, Goichiro Hanaoka, and Kazuo Ohta. Constructions of dynamic and non-dynamic threshold public-key encryption schemes with decryption consistency. *Theoretical Computer Science*, 630:95–116, 2016.

[47] Yusuke Sakai, Jacob C. N. Schuldt, Keita Emura, Goichiro Hanaoka, and Kazuo Ohta. On the security of dynamic group signatures: Preventing signature hijacking. In *Public Key Cryptography*, pages 715–732, 2012.

[48] Brent Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005.

[49] Shota Yamada, Nuttapong Attrapadung, Bagus Santoso, Jacob C. N. Schuldt, Goichiro Hanaoka, and Noboru Kunihiro. Verifiable predicate encryption and applications to CCA security and anonymous predicate authentication. In *Public Key Cryptography*, pages 243–261, 2012.