

Threshold Structure-Preserving Signatures

Elizabeth Crites[1(✉)], Markulf Kohlweiss[1,2], Bart Preneel[3], Mahdi Sedaghat[3], and Daniel Slamanig[4]

[1] University of Edinburgh, Edinburgh, UK
ecrites@ed.ac.uk, mkohlwei@inf.ed.ac.uk
[2] Input Output, Edinburgh, UK
[3] COSIC, KU Leuven, Leuven, Belgium
bart.preneel@esat.kuleuven.be, ssedagha@esat.kuleuven.be
[4] AIT Austrian Institute of Technology, Vienna, Austria
daniel.slamanig@ait.ac.at

Abstract. Structure-preserving signatures (SPS) are an important building block for privacy-preserving cryptographic primitives, such as electronic cash, anonymous credentials, and delegatable anonymous credentials. In this work, we introduce the first *threshold* structure-preserving signature scheme (TSPS). This enables multiple parties to jointly sign a message, resulting in a standard, single-party SPS signature, and can thus be used as a replacement for applications based on SPS.

We begin by defining and constructing SPS for indexed messages, which are messages defined relative to a unique index. We prove its security in the random oracle model under a variant of the generalized Pointcheval-Sanders assumption (PS). Moreover, we generalize this scheme to an indexed multi-message SPS for signing vectors of indexed messages, which we prove secure under the same assumption. We then formally define the notion of a TSPS and propose a construction based on our indexed multi-message SPS. Our TSPS construction is *fully non-interactive*, meaning that signers simply output partial signatures without communicating with the other signers. Additionally, signatures are short: they consist of 2 group elements and require 2 pairing product equations to verify. We prove the security of our TSPS under the security of our indexed multi-message SPS scheme. Finally, we show that our TSPS may be used as a drop-in replacement for UC-secure Threshold-Issuance Anonymous Credential (TIAC) schemes, such as Coconut, without the overhead of the Fischlin transform.

Keywords: Threshold Signatures · Structure-Preserving Signatures · Indexed Message Structure-Preserving Signatures

1 Introduction

Threshold cryptography [37–39] was designed to reduce the trust in single entities and improve the availability of keying material. It allows a secret key to

© International Association for Cryptologic Research 2023
J. Guo and R. Steinfeld (Eds.): ASIACRYPT 2023, LNCS 14439, pp. 348–382, 2023.
https://doi.org/10.1007/978-981-99-8724-5_11

be shared among a set of parties [16,79] such that the task involving the key can only be performed if some threshold of them collaborates. Threshold signatures [35,80], threshold encryption [26,81], and threshold verifiable unpredictable functions [61] enable distributed protocols, such as e-voting systems [30,31] and multi-party computation [29,36].

Threshold signatures in particular have attracted significant interest recently, in part because of advances in distributed ledger technologies, cryptocurrencies, and decentralized identity management [25,33,40,68,69]. They are also the subject of current standardization efforts by NIST [20,21]. Signatures used by certification authorities to issue credentials or to secure digital wallets make attractive targets for misuse or forgery. To mitigate these risks, an (n, t)-threshold signature scheme distributes the signing key among n parties such that any quorum of at least t signers can jointly generate a signature, but the scheme remains secure as long as fewer than t key shares are known to the adversary.

A threshold signature that is *fully non-interactive* consists of a single round of communication. On input the message, each signer computes its partial signature independently of other signers, and aggregation of at least t partial signatures results in a single signature representing the group. Interactive signing protocols involving two or more rounds add complexity and are error prone [41,84]. Thus, fully non-interactive schemes are preferable, the canonical example being threshold BLS [18,19].

Structure-Preserving Signatures. Structure-preserving signatures (SPS) [4] are pairing-based signatures where the message, signature, and verification key consist of source group elements only (in one or both source groups), and signature verification consists of group membership checks and pairing product equations only. SPS have been studied extensively, with a main focus on short signatures [5,7,52,54], lower bounds [1,5,6], and (tight) security under well-known assumptions [2,8,51,62,63,65,72].

SPS are compatible with Groth-Sahai non-interactive zero-knowledge proofs (NIZKs) [60] and, more generally, help to avoid the expensive extraction of exponents in security proofs. This makes them attractive for the modular design of protocols relying on signatures and NIZKs. Indeed, SPS have seen widespread adoption in privacy-preserving applications, such as group signatures [4,72], traceable signatures [3], blind signatures [4,47], attribute-based signatures [42], malleable signatures [9], anonymous credentials [23,46,48], delegatable anonymous credentials [13,34], and anonymous e-cash [17].

For such signature-based applications, compromise of the signing key represents a single point of attack and failure. Replacing the use of SPS with TSPS together with distributed key generation (DKG) would help to reduce the trust in a single authority and increase the availability of the respective signing service. While many of the aforementioned applications of SPS would benefit from thresholdization, until now there was no known threshold construction of SPS that could serve as their basis. We provide the first candidate TSPS scheme as the main contribution of this work.

Towards Constructing a Threshold SPS. Our goal is to construct threshold SPS that are fully non-interactive, i.e., there is no coordination among signers. This puts some requirements on the used SPS and in particular prevents the use of nonlinear operations of the signing randomness and secret keys (cf. Sect. 2), which existing SPS fail to satisfy. Thus, as a starting point for our TSPS, we consider the pairing-based Pointcheval-Sanders signature scheme (PS) [76] (cf. Sect. 3.2), as its randomness is simply a random base group element and it avoids hashing during verification. We recall that the PS scheme is defined over an asymmetric bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g})$ with signing key $\mathsf{sk} = (x, y) \in (\mathbb{Z}_p^*)^2$ and corresponding verification key $\mathsf{vk} = (\hat{g}^x, \hat{g}^y) \in \mathbb{G}_2^2$. The signing algorithm takes as input a scalar message $m \in \mathbb{Z}_p$ and outputs a signature

$$\sigma = (h, s) = (g^r, h^{x+my}) \in \mathbb{G}_1^2 \ .$$

Importantly, the nonce r (or equivalently the base h) is sampled fresh for each signature. This scheme fails to be an SPS because the message is not a group element (or elements). Ghadafi [52] made the observation that a PS-like SPS scheme can be constructed for a group element message $(M_1, M_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ for which there exists a scalar message $m \in \mathbb{Z}_p$ such that $M_1 = g^m$ and $M_2 = \hat{g}^m$. This is referred to as a Diffie-Hellman (DH) message. (cf. Sect. 1 for more on this message space.) A Ghadafi SPS signature (cf. Sect. 3.2) has the form:

$$\sigma = (h, s, t) = (g^r, M_1^r, h^x s^y) \in \mathbb{G}_1^3 \ .$$

Let us see how one might construct a threshold version of this scheme. Suppose each signer possesses a share $\mathsf{sk}_i = (x_i, y_i)$ of the secret key $\mathsf{sk} = (x, y)$. A first (non-interactive) attempt might have each signer output a partial signature of the form:

$$\sigma_i = (h_i, s_i, t_i) = (g^{r_i}, M_1^{r_i}, h_i^{x_i} s_i^{y_i}) \ ,$$

with aggregation of the third term having the form:

$$t = \prod_{i \in \mathcal{T}} t_i^{\lambda_i} = \prod_{i \in \mathcal{T}} g^{r_i x_i \lambda_i} M_1^{r_i y_i \lambda_i} \ ,$$

where λ_i is the Lagrange coefficient for party i in the signing set $\mathcal{T}$ of size at least t (the threshold). As with other existing SPS, this however does not allow reconstruction via Lagrange interpolation because each term in the exponent is multiplied by a distinct random integer r_i. To overcome this, due to the specific form of the signatures, the signers would have to agree on a common random element $h = g^r$. Indeed, this will be our approach to solve this issue.

A second (interactive) attempt might have each signer output randomness shares $h_i = g^{r_i}$ and corresponding $s_i = M_1^{r_i}$ in a first round of signing, followed by a second round in which each signer computes aggregate values $h = g^r = \prod_{i \in \mathcal{T}} h_i = g^{\sum_{i \in \mathcal{T}} r_i}$ and $s = \prod_{i \in \mathcal{T}} s_i$ and outputs a partial signature of the form:

$$\sigma_i = (h, s, t_i = h^{x_i} s^{y_i}) \ , \tag{1}$$

with aggregation of the third term having the form:

$$t = \prod_{i \in \mathcal{T}} t_i^{\lambda_i} = \prod_{i \in \mathcal{T}} g^{r x_i \lambda_i} M_1^{r y_i \lambda_i} \quad .$$

This allows reconstruction via Lagrange interpolation. In terms of security, the unforgeability of this threshold scheme may be reduced to the unforgeability of single-party Ghadafi SPS signatures. However, the reduction needs to obtain the corrupt h_j, s_j values before revealing honest values h_i, s_i. The addition of a third signing round could achieve this, whereby all values h_i, s_i are committed to in the first round as $\mathsf{H}(h_i), \mathsf{H}'(s_i)$, for H and H' modeled as random oracles, and then revealed in the second round. However, the reduction needs to obtain the *nonces* r_j of the corrupt parties, which may be extracted from zero-knowledge proofs appending the outputs h_i, s_i in round two. These additional rounds and zero-knowledge proofs add significant overhead.

Our approach is clean and straightforward: we instead have signers obtain shared randomness $h = g^r$ via a random oracle, yielding a fully non-interactive scheme. But observe that if partial signatures have the form of Eq. (1), then $s = M_1^r$ cannot be computed without knowledge of the discrete logarithm $\mathsf{dlog}_h(M_1)$. Thus, we borrow techniques from Sonnino et al. [82] and Camenisch et al. [22], which implicitly sign *indexed* Diffie-Hellman messages (id, M_1, M_2), a concept we define and formalize rigorously in this work. Indexing can be understood as requiring the existence of an injective function f that maps each scalar message $m \in \mathbb{Z}_p$ to an index $id = f(m)$. We then have $h = \mathsf{H}(id)$, where H is modeled as a random oracle, and $M_1 = \mathsf{H}(id)^m$. Then each partial signature has the form:

$$\sigma_i = (h, s_i) = (\mathsf{H}(id), h^{x_i} M_1^{y_i}),$$

and the aggregated signature has the form:

$$\sigma = (h, s) = (\mathsf{H}(id), h^x M_1^y). \tag{2}$$

This is exactly our TSPS construction, with the underlying SPS signature defined by Eq. (2). We extend these techniques to vectors of indexed Diffie-Hellman messages $(id, \vec{M}_1, \vec{M}_2)$, which allows additional elements to be signed, e.g., attributes when used within anonymous credential systems [76,82]. It is important to note that the index is not needed for verification (and therefore $\mathsf{H}(id)$ is not computed), so our schemes are indeed structure preserving.

We define an appropriate notion of unforgeability for indexed messages: existential unforgeability under chosen indexed message attack (EUF-CiMA) and prove the security of our constructions under this notion. We discuss various ways of defining the index function, depending on the application. For example, if privacy is not required and the message and public key are known, the index function may simply be the identity function: $id = f(m) = m$, capturing the intuitive notion that each nonce r is associated with a single scalar message m.

Why Diffie-Hellman Messages? Diffie-Hellman messages can be traced back to the introduction of automorphic signatures [45] and SPS [4], and have since appeared in various other SPS constructions [52–55]. Their use is largely motivated by an impossibility result by Abe et al. [5], which proves that any SPS in the Type-III setting must have at least 3 group elements and 2 pairing product equations in the verification. Furthermore, the result rules out unilateral signatures (those containing elements from only one source group) meeting this lower bound. However, if messages are in *both* source groups, it is possible to construct a unilateral SPS meeting this lower bound. This is what Diffie-Hellman messages and the Ghadafi construction [52] achieve. We follow the same approach to construct efficient TSPS.

Constructing a TSPS over standard, unilateral message spaces remains an interesting open problem. However, such a scheme would necessarily contain more group elements in the signature and more pairing product equations to verify, due to this impossibility result. This is an important consideration when combining with Groth-Sahai NIZK proofs in applications, as the number of pairings required for verification scales linearly with the number of source pairings.

1.1 Our Contributions

Our contributions can be summarized as follows:

- We formalize the concept of indexed message spaces and formally define the notion of structure-preserving signatures (SPS) over indexed message spaces and corresponding notion of security: existential unforgeability under chosen indexed message attack (EUF-CiMA).
- We propose a concrete SPS construction over indexed Diffie-Hellman messages, called IM-SPS, and prove its EUF-CiMA security under a new variant of the generalized Pointcheval-Sanders assumption. We reduce this assumption to the hardness of the $(2, 1)$-discrete logarithm problem in the algebraic group model (AGM).
- We provide an indexed multi-message SPS construction, called IMM-SPS, which allows vectors of indexed Diffie-Hellman messages to be signed, and prove its EUF-CiMA security under the same assumption.
- We introduce the notion of a threshold structure-preserving signature (TSPS) scheme and propose a fully non-interactive TSPS based on our EUF-CiMA secure SPS scheme. Signatures contain only 2 group elements and verification consists of 2 pairing product equations. We prove the security of our TSPS under the EUF-CiMA security of IMM-SPS.
- We discuss applications of our TSPS construction and, in particular, blind signing of messages. This represents a core functionality in Threshold-Issuance Anonymous Credential (TIAC) systems. We outline how our TSPS can be used in TIAC systems as a drop-in replacement that avoids rewinding extractors for the required non-interactive zero-knowledge (NIZK) proofs.

2 Related Work

We provide an overview of pairing-based non-interactive threshold signature schemes in Table 1 and structure-preserving signature schemes (SPS) in Table 2 and discuss how these schemes fail to meet our requirements.

Table 1. Table of pairing-based non-interactive threshold signature schemes. iDH refers to indexed Diffie-Hellman messages (Definition 7). ✓: Satisfied. ✗: Not satisfied.

Scheme	Message Space	Signature Size	Structure Preserving
BLS [10,18]	$\{0,1\}^*$	$1\mathbb{G}_1$	✗
LJY ‡1 [70]	$\{0,1\}^*$	$2\mathbb{G}_1$	✗
LJY ‡2 [70]	$\{0,1\}^*$	$4\mathbb{G}_1 + 2\mathbb{G}_2$	✗
GJMMST [61]	$\{0,1\}^*$	$4\mathbb{G}_1 + 2\mathbb{G}_2$	✗
PS [82,83]	$\mathbb{Z}_p$	$2\mathbb{G}_1$	✗
Our TSPS	iDH	$2\mathbb{G}_1$	✓

Threshold Signatures. BLS [19] and its threshold version [10,18] are not structure preserving, as they map bitstring messages $\{0,1\}^*$ to the group using a random oracle. Libert et al. [70] propose a secure non-interactive threshold signature scheme based on linearly-homomorphic SPS (LHSPS) [71]. While this construction meets many of our requirements, the resulting threshold signature is not structure preserving. It either relies on random oracles to hash bitstring messages to group elements (‡1 [70]) or, when avoiding random oracles, a bit-wise encoding of the message is required (‡2 [70]). Gurkan et al. [61] propose a pairing-based threshold Verifiable Unpredictable Function (VUF), which is essentially a unique threshold signature [73]. However, their construction is not structure preserving: it hashes bitstring messages to the group using a random oracle. Sonnino et al. [82] and Tomescu et al. [83] present non-interactive threshold versions of Pointcheval-Sanders (PS) signatures; however, verification takes place over scalar vectors, and is thus not structure preserving. We note that signatures for scalar vectors are intuitively closer to SPS than ones for bitstring messages, as evidenced, for example, by Ghadafi's scheme [52]. We do not know of a general conversion technique, however.

Structure-Preserving Signatures. Most structure-preserving signatures in the literature fail to be good candidates for thresholdization due to nonlinear operations of signer-specific randomness and secret key elements, which are not amenable to Lagrange interpolation (e.g., [4,5,7,11,54,58]). However, there are two promising approaches: linearly-homomorphic SPS (LHSPS) [71] and the SPS by Ghadafi [52]. The former is a one-time signature, meaning that a key pair

can only sign a single message[1]. The SPS by Ghadafi [52] lends itself to thresholdization, but it requires multiple communication rounds and incurs significant overhead. (See Sect. 1 for a discussion of this approach.)

Table 2. Table of structure-preserving signature schemes (SPS). DH refers to Diffie-Hellman messages (Definition 2), and iDH refers to indexed Diffie-Hellman messages (Definition 7). Avoids Nonlinearity refers to operations of the signing randomness and secret keys. ✓: Satisfied. ✗: Not satisfied.

Scheme	Message Space	Signature Size	Avoids Nonlinearity
AFGHO [4]	$\mathbb{G}_1$	$5\mathbb{G}_1 + 2\mathbb{G}_2$	✗
AGHO [5]	$\mathbb{G}_1 \times \mathbb{G}_2$ / $\mathbb{G}_2$	$2\mathbb{G}_1 + 1\mathbb{G}_2$	✗
AGOT [7]	$\mathbb{G}_1$	$2\mathbb{G}_1 + 1\mathbb{G}_2$	✗
BFFSST [11]	$\mathbb{G}_2$	$1\mathbb{G}_1 + 2\mathbb{G}_2$	✗
Ghadafi [54]	DH	$2\mathbb{G}_1$	✗
Ghadafi [52]	DH	$3\mathbb{G}_1$	✗
Groth [58]	$\mathbb{G}_2$	$1\mathbb{G}_1 + 2\mathbb{G}_2$	✗
LPJY [71]*	$\mathbb{G}_1$	$2\mathbb{G}_1$	✓
Our SPS	iDH	$2\mathbb{G}_1$	✓

* One-time: a key pair can only sign a single message.

3 Preliminaries and Definitions

3.1 General

Let $\kappa \in \mathbb{N}$ denote the security parameter and 1^κ its unary representation. Let p be a κ-bit prime. For all positive polynomials $f(\kappa)$, a function $\nu : \mathbb{N} \to \mathbb{R}^+$ is called *negligible* if $\exists\ \kappa_0 \in \mathbb{N}$ such that $\forall\ \kappa > \kappa_0$ it holds that $\nu(\kappa) < 1/f(\kappa)$. We denote by $\mathbb{G}^*$ the set $\mathbb{G} \setminus 1_{\mathbb{G}}$, where $1_{\mathbb{G}}$ is the identity element of the group $\mathbb{G}$. We denote the group of integers mod p by $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$, its multiplicative group of units by $\mathbb{Z}_p^*$, and the polynomial ring over $\mathbb{Z}_p$ by $\mathbb{Z}_p[X]$. For a group $\mathbb{G}$ of order p with generator g, we denote the discrete logarithm $m \in \mathbb{Z}_p$ of $M \in \mathbb{G}$ base g by $\mathsf{dlog}_g(M)$ (i.e., $M = g^m$). We denote the set of integers $\{1, \ldots, n\}$ by $[1, n]$ and the vector A by $\vec{A}$. Let $Y \leftarrow\!\!\$\ F(X)$ denote running probabilistic algorithm F on input X and assigning its output to Y. Let $x \leftarrow\!\!\$\ \mathbb{Z}_p$ denote sampling an element of $\mathbb{Z}_p$ uniformly at random. All algorithms are randomized unless expressly stated otherwise. PPT refers to probabilistic polynomial time. We denote the output of a security game $\mathbf{G}^{\text{Game}}$ between a challenger and a PPT adversary $\mathcal{A}$ by $\mathbf{G}_{\mathcal{A}}^{\text{Game}}$, where $\mathcal{A}$ wins the game if $\mathbf{G}_{\mathcal{A}}^{\text{Game}} = 1$.

[1] Note that the LHSPS in [71] is designed over symmetric bilinear groups with signatures consisting of 3 group elements. The authors in [70] extend this LHSPS over asymmetric bilinear groups with signatures of size 2.

Definition 1 (Bilinear Group). *A bilinear group generator $\mathcal{BG}(1^\kappa)$ returns a tuple $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g})$ such that $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are finite groups of the same prime order p, $g \in \mathbb{G}_1$ and $\hat{g} \in \mathbb{G}_2$ are generators, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable bilinear pairing, which satisfies the following properties:*

1. *$e(g, \hat{g}) \neq 1_{\mathbb{G}_T}$ (non-degeneracy).*
2. *$\forall\ a, b \in \mathbb{Z}_p$, $e(g^a, \hat{g}^b) = e(g, \hat{g})^{ab} = e(g^b, \hat{g}^a)$ (bilinearity).*

We rely on bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with no efficiently computable isomorphism between them [50], also called Type-III or asymmetric bilinear groups. To date, they are the most efficient choice for relevant security levels.

Definition 2 (Diffie-Hellman Message Space [4,45]). *Over an asymmetric bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g})$, a pair $(M_1, M_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ belongs to the Diffie-Hellman (DH) message space $\mathcal{M}_{\mathsf{DH}}$ if there exists $m \in \mathbb{Z}_p$ such that $M_1 = g^m$ and $M_2 = \hat{g}^m$.*

One can efficiently verify whether $(M_1, M_2) \in \mathcal{M}_{\mathsf{DH}}$ by checking $e(M_1, \hat{g}) = e(g, M_2)$.

Definition 3 (Algebraic Group Model [49]). *An adversary is algebraic if for every group element $h \in \mathbb{G} = \langle g \rangle$ that it outputs, it is required to output a representation $\vec{h} = (\eta_0, \eta_1, \eta_2, \dots)$ such that $h = g^{\eta_0} \prod g_i^{\eta_i}$, where $g, g_1, g_2, \dots \in \mathbb{G}$ are group elements that the adversary has seen thus far.*

The original definition of the algebraic group model (AGM) [49] only captured regular cyclic groups $\mathbb{G} = \langle g \rangle$. Mizuide et al. [74] extended this definition to include symmetric pairing groups ($\mathbb{G}_1 = \mathbb{G}_2$), such that the adversary is also allowed to output target group elements (in $\mathbb{G}_T$) and their representations. Recently, Couteau and Hartmann [28] defined the Algebraic Asymmetric Bilinear Group Model, which extends the AGM definition for asymmetric pairings by allowing the adversary to output multiple elements from all three groups. The definition can be found in the full version [32].

3.2 Schemes

Pointcheval-Sanders Signatures [76]. The PS signature scheme is defined over the message space $\mathcal{M}$ of scalar messages $m \in \mathbb{Z}_p$ and consists of the following PPT algorithms:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$: Output $\mathsf{pp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$.
- $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$: Sample $x, y \leftarrow_\$ \mathbb{Z}_p^*$ and set $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2) = (x, y)$ and $\mathsf{vk} = (\mathsf{vk}_1, \mathsf{vk}_2) = (\hat{g}^x, \hat{g}^y)$. Output $(\mathsf{sk}, \mathsf{vk})$.
- $\sigma \leftarrow \mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, m)$: Sample $r \leftarrow_\$ \mathbb{Z}_p^*$ and compute $\sigma = (h, s) = (g^r, h^{x+my})$. Output σ.
- $0/1 \leftarrow \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, m, \sigma)$: If $h \in \mathbb{G}_1, h \neq 1_{\mathbb{G}_1}$, and the pairing product equation $e(h, \mathsf{vk}_1 \mathsf{vk}_2^m) = e(s, \hat{g})$ holds, output 1 (accept); else, output 0 (reject).

Pointcheval-Sanders signatures are EUF-CMA secure under the PS assumption (Definition 5) [76].

Ghadafi SPS [52]**.** The Ghadafi structure-preserving signature scheme is defined over the message space $\mathcal{M}_{\mathsf{DH}}$ of Diffie-Hellman pairs $(M_1, M_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ such that $e(M_1, \hat{g}) = e(g, M_2)$ and consists of the following PPT algorithms:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$: Output $\mathsf{pp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$.
- $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$: Sample $x, y \leftarrow_\$ \mathbb{Z}_p^*$ and set $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2) = (x, y)$ and $\mathsf{vk} = (\mathsf{vk}_1, \mathsf{vk}_2) = (\hat{g}^x, \hat{g}^y)$. Output $(\mathsf{sk}, \mathsf{vk})$.
- $\sigma \leftarrow \mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, M_1, M_2)$: Sample $r \leftarrow_\$ \mathbb{Z}_p^*$ and compute $\sigma = (h, s, t) = (g^r, M_1^r, h^x s^y)$. Output σ.
- $0/1 \leftarrow \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \sigma, M_1, M_2)$: If $h, s, t \in \mathbb{G}_1, h \neq 1_{\mathbb{G}_1}$, and both pairing product equations $e(h, M_2) = e(s, \hat{g})$ and $e(t, \hat{g}) = e(h, \mathsf{vk}_1)e(s, \mathsf{vk}_2)$ hold, output 1 (accept); else, output 0 (reject).

The Ghadafi SPS is weakly EUF-CMA secure in the generic group model (GGM) [52].

Shamir Secret Sharing [79]**.** An (n, t)-Shamir secret sharing divides a secret s among n shareholders such that each subset of at least t shareholders can reconstruct s, but fewer than t cannot (and s remains information-theoretically hidden). A dealer who knows the secret s forms a polynomial $f(x)$ of degree t with randomly chosen coefficients from $\mathbb{Z}_p$ such that $f(0) = s$. The dealer then securely provides each shareholder with $s_i = f(i), i \in [1, n]$. Let $\vec{s} \leftarrow_\$ \mathsf{Share}(s, p, n, t)$ denote the process of computing shares $\vec{s} = (s_1, \ldots, s_n)$ of a secret s. Each subset $\mathcal{T} \subset [1, n]$ of size at least t can pool their shares to reconstruct the secret s using Lagrange interpolation, as $s = f(0) = \sum_{i \in \mathcal{T}} s_i \lambda_i$, where $\lambda_i = \prod_{j \in \mathcal{T}, j \neq i} \frac{j}{j-i}$.

3.3 Assumptions

Definition 4 ($(2,1)$-Discrete Logarithm Assumption [12]**).** *Let* $\mathsf{pp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$ *be an asymmetric bilinear group. The* $(2,1)$*-discrete logarithm assumption holds with respect to* $\mathcal{BG}$ *if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* ν *such that*

$$\Pr\left[z \leftarrow_\$ \mathbb{Z}_p^*;\ (Z, Z', \hat{Z}) \leftarrow (g^z, g^{z^2}, \hat{g}^z);\ z' \leftarrow_\$ \mathcal{A}(\mathsf{pp}, Z, Z', \hat{Z}) :\ z' = z\right] < \nu(\kappa) \ .$$

Definition 5 (PS Assumption [76]**).** *Let the advantage of an adversary* $\mathcal{A}$ *against the PS game* $\mathbf{G}^{PS}$*, as defined in Fig. 1, be as follows:*

$$Adv_{\mathcal{A}}^{PS}(\kappa) = \Pr\left[\mathbf{G}_{\mathcal{A}}^{PS} = 1\right] \ .$$

The PS assumption holds if for all PPT adversaries $\mathcal{A}$*, there exists a negligible function* ν *such that* $Adv_{\mathcal{A}}^{PS}(\kappa) < \nu(\kappa)$.

$\mathbf{G}^{\mathrm{PS}}(1^\kappa)$

1 : $\mathsf{pp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$
2 : $x, y \leftarrow\$\, \mathbb{Z}_p^*$
3 : $(m^*, h^*, s^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathrm{PS}}}(\mathsf{pp}, \hat{g}^x, \hat{g}^y)$
4 : **return** $\big((1)\ h^* \neq 1_{\mathbb{G}_1} \wedge\ m^* \neq 0 \wedge$
5 : $\quad (2)\ s^* = h^{*^{x+m^*y}} \wedge$
6 : $\quad (3)\ m^* \notin \mathcal{Q}\big)$

$\mathcal{O}^{\mathrm{PS}}(m)$ // $m \in \mathbb{Z}_p$

1 : $h \leftarrow\$\, \mathbb{G}_1$
2 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{m\}$
3 : **return** (h, h^{x+my})

Fig. 1. Game defining the PS assumption.

The validity of the tuple (m^*, h^*, s^*) is decidable by checking $e(s^*, \hat{g}) = e(h^*, \hat{g}^x(\hat{g}^y)^{m^*})$. The PS assumption is an interactive assumption defined by Pointcheval and Sanders [76] to construct an efficient randomizable signature and has been shown to hold in the GGM.

Kim et al. [66] introduced a generalized version of the PS assumption (GPS) that splits the PS oracle $\mathcal{O}^{\mathrm{PS}}(\cdot)$ into two oracles $\mathcal{O}_0^{\mathrm{GPS}}(), \mathcal{O}_1^{\mathrm{GPS}}(\cdot)$: the first samples $h \leftarrow\$\, \mathbb{G}_1$, and the second takes h and m as input and generates the PS value h^{x+my}. Recently, Kim et al. [67] extended the GPS assumption (GPS_2), replacing field element inputs, such as m, with group element inputs. The GPS_2 assumption holds under the $(2,1)$-DL assumption (Definition 4) in the AGM. Both the GPS and GPS_2 assumptions can be found in the full version [32].

Owing to the fact that our SPS and TSPS constructions rely on a different message space, we introduce an analogous generalized PS assumption (GPS_3), defined as follows.

Definition 6 (GPS_3 Assumption). *Let the advantage of an adversary $\mathcal{A}$ against the GPS_3 game $\mathbf{G}^{GPS_3}$, as defined in Fig. 2, be as follows:*

$$Adv_{\mathcal{A}}^{GPS_3}(\kappa) = \Pr\left[\mathbf{G}_{\mathcal{A}}^{GPS_3} = 1\right] \ .$$

The GPS_3 assumption holds if for all PPT adversaries $\mathcal{A}$, there exists a negligible function ν such that $Adv_{\mathcal{A}}^{GPS_3}(\kappa) < \nu(\kappa)$.

We prove that this assumption holds in the AGM if the $(2,1)$-DL problem is hard (Theorem 1).

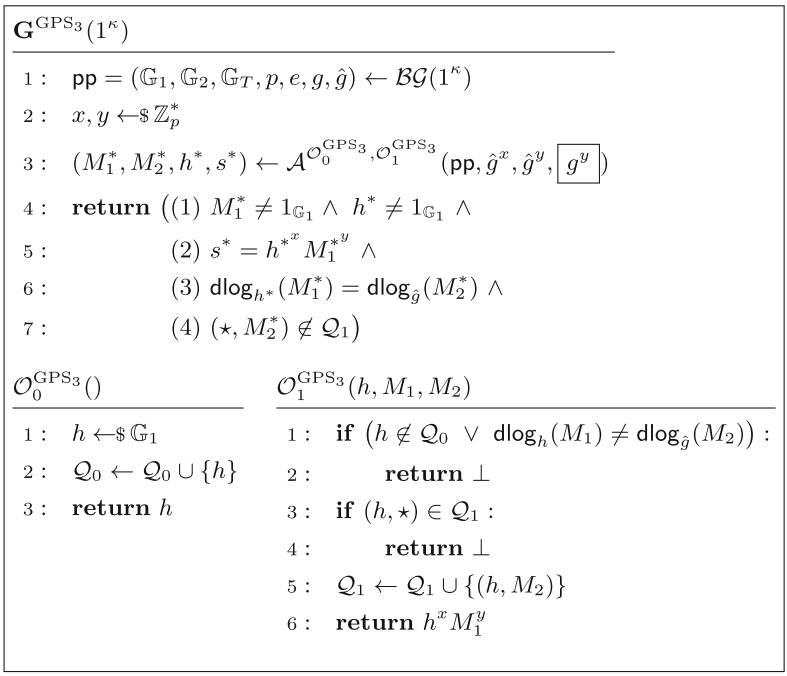


Fig. 2. Game defining our GPS_3 assumption. The additional element in the solid box is required for blind signing only (cf. Sect. 6.1).

4 Indexed Message Structure-Preserving Signatures

In this section, we introduce the notion of structure-preserving signatures (SPS) on indexed messages as well as a corresponding notion of security: unforgeability against chosen indexed message attack (EUF-CiMA). We provide an indexed message SPS construction, called IM-SPS, and prove its EUF-CiMA security under the GPS_3 assumption (Definition 6) in the random oracle model (ROM) (Theorem 2). We also propose an indexed *multi*-message SPS construction, called IMM-SPS, which allows vectors of indexed messages to be signed, and prove its EUF-CiMA security under the same assumptions (Theorem 3). IMM-SPS are useful for applications where additional elements, such as attributes, are signed.

Indexing can be understood as requiring the existence of an injective function f that maps each message to an index. We model this by requiring that for all index/message pairs in an indexed message space $\mathcal{M}$, the following uniqueness property holds: $(id, \tilde{M}) \in \mathcal{M}$, $(id', \tilde{M'}) \in \mathcal{M}$, $id = id' \Rightarrow \tilde{M} = \tilde{M'}$. That is, no two messages use the same index. We refer to index/message pairs as $M = (id, \tilde{M})$.

Indexing is useful, as signatures can depend on the index; for example, in our schemes, signing involves evaluating a hash-to-curve function H on the index to obtain a base element $h \leftarrow \mathsf{H}(id)$. Verifying a message/signature pair does not

require availability of the index, making it structure preserving. Consequently, the verification message space $\tilde{\mathcal{M}}$ is obtained from $\mathcal{M}$ by omitting the index.

For our schemes, we need to consider that in verification one can provide a base element h^r obtained by randomizing the original base element h. This is due to the partial randomizability of the signatures. Thus, different messages $\tilde{M}, \tilde{M}'$ may be valid representations for the same scalar message m. Consequently, similar to SPS on equivalence classes (SPS-EQ) [48], the verification message space $\tilde{\mathcal{M}}$ is expanded to consider equivalent (randomized) messages: $\tilde{\mathcal{M}} = \{\tilde{M} \mid \exists\ (\cdot, \tilde{M}') \in \mathcal{M}\ ,\ \tilde{M} \in \mathsf{EQ}(\tilde{M}')\}$. The function EQ depends on the concrete message space and determines the respective set of equivalent messages.

Next, we define the indexed Diffie-Hellman message space used by our IM-SPS scheme (cf. Fig. 3 for its encoding function).

Definition 7 (Indexed Diffie-Hellman Message Space). *Given an asymmetric bilinear group* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$*, an index set* $\mathcal{I}$*, and a random oracle* $\mathsf{H} : \mathcal{I} \to \mathbb{G}_1$*,* $\mathcal{M}^{\mathsf{H}}_{\mathsf{iDH}}$ *is an indexed Diffie-Hellman (DH) message space if* $\mathcal{M}^{\mathsf{H}}_{\mathsf{iDH}} \subset \{(id, \tilde{M}) \mid id \in \mathcal{I}, m \in \mathbb{Z}_p, \tilde{M} = (\mathsf{H}(id)^m, \hat{g}^m) \in \mathbb{G}_1 \times \mathbb{G}_2\}$ *and the following index uniqueness property holds: for all* $(id, \tilde{M}) \in \mathcal{M}^{\mathsf{H}}_{\mathsf{iDH}}$*,* $(id', \tilde{M}') \in \mathcal{M}^{\mathsf{H}}_{\mathsf{iDH}}$*,* $id = id' \Rightarrow \tilde{M} = \tilde{M}'$*.*

We define the equivalence class for each message $\tilde{M} = (M_1, M_2) \in \tilde{\mathcal{M}}^{\mathsf{H}}_{\mathsf{iDH}}$ *as* $\mathsf{EQ}_{\mathsf{iDH}}(M_1, M_2) = \{(M_1^r, M_2) \mid \exists\ r \in \mathbb{Z}_p\}$*.*

$\mathsf{iDH}^{\mathsf{H}}(id, m)$		$\mathsf{H}(id)$	
1 :	$h \leftarrow \mathsf{H}(id)$	1 :	**if** $\mathcal{Q}_{\mathsf{H}}[id] = \perp$:
2 :	$\tilde{M} \leftarrow (h^m, \hat{g}^m)$	2 :	$\mathcal{Q}_{\mathsf{H}}[id] \leftarrow\!\!\$\ \mathbb{G}_1$
3 :	**return** $(id, \tilde{M})$	3 :	**return** $\mathcal{Q}_{\mathsf{H}}[id]$

Fig. 3. Encoding function of indexed Diffie-Hellman message space in the ROM.

The subset membership is efficiently decidable by checking $e(M_1, \hat{g}) = e(h, M_2)$ for $h \leftarrow \mathsf{H}(id)$. Note that, in addition, one needs to guarantee that no two messages use the same index. This is the responsibility of the signer.[2] As mentioned above, messages $\tilde{M}$ lie in a different verification message space $\tilde{\mathcal{M}}^{\mathsf{H}}_{\mathsf{iDH}}$ that is uniquely determined by $\mathcal{M}^{\mathsf{H}}_{\mathsf{iDH}}$ and $\mathsf{EQ}_{\mathsf{iDH}}$. Note that most $\tilde{M} \in \tilde{\mathcal{M}}^{\mathsf{H}}_{\mathsf{iDH}}$ are not indexed Diffie-Hellman messages. In particular, when expanding the definition of $\mathsf{EQ}_{\mathsf{iDH}}$, the verification message space is $\tilde{\mathcal{M}}^{\mathsf{H}}_{\mathsf{iDH}} = \{(M_1^r, M_2) \mid \exists\ r \in \mathbb{Z}_p\ ,\ \exists\ (\cdot, M_1, M_2) \in \mathcal{M}^{\mathsf{H}}_{\mathsf{iDH}}\}$.

[2] To highlight this responsibility, we enforce uniqueness both in the message space and later on in Line 1 of the of $\mathcal{O}_{\mathsf{Sign}}(\cdot)$ oracle of Fig. 5.

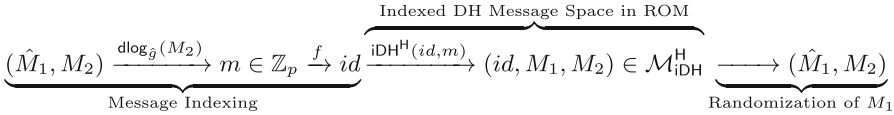


Fig. 4. From $\tilde{M}$ to M and back again: The first message component is randomizable; the second fixes the index.

Does $\tilde{M}$ depend on id or does id depend on $\tilde{M}$? One might observe the above apparent circularity with respect to the indexing technique. On the one hand, we require existence of an injective function f that maps (M_1, M_2) to id. On the other hand, M_1 is computed as $M_1 = \mathsf{H}(id)^{\mathsf{dlog}_{\hat{g}}(M_2)}$. This circularity is avoided by computing id from the partial message M_2, or more commonly its discrete logarithm m.

As illustrated in Fig. 4, the indexing function f assigns an index id to each scalar message $m \in \mathbb{Z}_p$. Then, a hash-to-curve function $\mathsf{H} : \{0,1\}^* \to \mathbb{G}_1$ (modeled as a random oracle) is used to generate a unique base element h. A source group message (M_1, M_2) can then be obtained using h. In an indexed message SPS, the signing algorithm takes as input the source group message together with an index and generates the underlying signature with access to H. Note that the index does not destroy the structure since the verifier does not need to know id to verify a signature on message $\tilde{M} = (M_1, M_2)$.

Indexing Function Instantiations. Depending on the application, the indexing function f can be instantiated in different ways. For example, if messages and signatures are allowed to be public, the indexing function can be instantiated by using the scalar message m itself as the index: $f(m) \mapsto m = id$.

If message and signatures must be hidden, as in the case of applications to anonymous credentials, one can take the approach of committing to the scalar message and providing a proof of well-formedness of the commitment, as done by Sonnino et al. [82]. As it is infeasible to open a well-formed commitment to two different messages, this guarantees uniqueness of the index. Camenisch et al. [22] take yet another approach for indexing messages: they assume the existence of a pre-defined and publicly available indexing function. That is, there is a unique index value for each message that is known to all signers. The corresponding base element can be obtained by evaluating the hash-to-curve function at the given index. As the authors note, if the size of the message space is polynomial and known in advance, then this approach is secure, since it is equivalent to including the base element in the public parameters. However, this is impractical for large message spaces.

4.1 Definition of Unforgeability for Indexed Message SPS

We adapt the notion of EUF-CMA security for digital signatures (Definition 16) to existential unforgeability against chosen *indexed* message attack (EUF-CiMA).

There are two adjustments: (1) the adversary makes queries to the signing oracle by providing *index*/message pairs, and (2) we expand the set of signed messages $\mathcal{Q}_{\mathsf{S}} = \{(id_i, \tilde{M}_i)\}_i$ to the set of trivially forgeable messages $\mathcal{Q}_{\mathsf{EQ}} = \{\mathsf{EQ}(\tilde{M}_i)\}_i$, i.e., all equivalent messages in the verification message space, and use it in the winning condition of the adversary.

Definition 8 (Existential Unforgeability under Chosen Indexed Message Attack (EUF-CiMA)). *A digital signature scheme over indexed message space $\mathcal{M}$ is EUF-CiMA secure if for all PPT adversaries $\mathcal{A}$ playing game $\mathbf{G}^{EUF\text{-}CiMA}$ (Fig. 5), there exists a negligible function ν such that*

$$Adv^{EUF\text{-}CiMA}_{\mathcal{A}}(\kappa) = \Pr\left[\mathbf{G}^{EUF\text{-}CiMA}_{\mathcal{A}}(1^\kappa) = 1\right] \leq \nu(\kappa).$$

$\mathbf{G}^{\text{EUF-CiMA}}_{\mathcal{A}}(1^\kappa)$

1 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$
2 : $(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\$\ \mathsf{KGen}(\mathsf{pp})$
3 : $(\tilde{M}^*, \sigma^*) \leftarrow\!\$\ \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(\mathsf{pp}, \mathsf{vk})$
4 : **return** $\big(\tilde{M}^* \notin \mathcal{Q}_{\mathsf{EQ}} \wedge$
5 : $\qquad \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \tilde{M}^*, \sigma^*)\big)$

$\mathcal{O}_{\mathsf{Sign}}(id, \tilde{M})$

1 : **if** $(id, \star) \in \mathcal{Q}_{\mathsf{S}}$:
2 : $\quad$ **return** $\perp$
3 : **else** : $\sigma \leftarrow \mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, (id, \tilde{M}))$
4 : $\quad \mathcal{Q}_{\mathsf{S}} \leftarrow \mathcal{Q}_{\mathsf{S}} \cup \{(id, \tilde{M})\}$
5 : $\quad \mathcal{Q}_{\mathsf{EQ}} \leftarrow \mathcal{Q}_{\mathsf{EQ}} \cup \{\mathsf{EQ}(\tilde{M})\}$
6 : $\quad$ **return** σ

Fig. 5. Game $\mathbf{G}^{\text{EUF-CiMA}}_{\mathcal{A}}(1^\kappa)$.

4.2 Our Indexed Message SPS

In Fig. 6, we present our indexed message SPS construction IM-SPS over the indexed Diffie-Hellman message space $\mathcal{M}^{\mathsf{H}}_{\mathsf{iDH}}$.

4.3 Security of IM-SPS

We prove that our proposed IM-SPS construction (Fig. 6) is EUF-CiMA secure under the GPS_3 assumption (Definition 6) in the random oracle model.

The GPS_3 assumption underpins both the security of IM-SPS as well as our indexed *multi*-message SPS construction IMM-SPS (Sect. 4.4). Our security reductions from IM-SPS and IMM-SPS to GPS_3 are tight. Furthermore, we show the tight security of our TSPS (Sect. 5) under the security of IMM-SPS. Figure 7 defines a roadmap for our IM-SPS, IMM-SPS, and TSPS constructions and their underlying assumptions. Thus, as a starting point, we reduce the GPS_3 assumption to the hardness of the $(2,1)$-DL problem (Definition 4) in the algebraic group model.

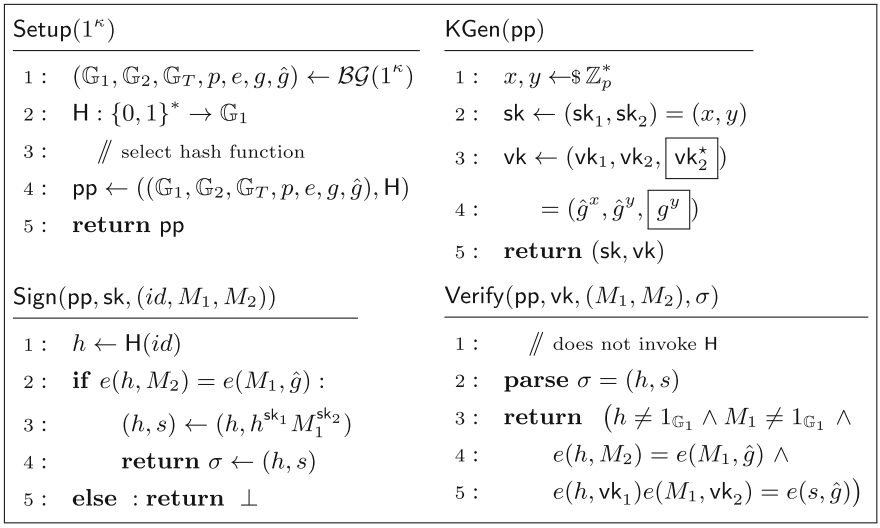


Fig. 6. Our Indexed Message SPS Construction IM-SPS. The additional elements in solid boxes are required for blind signing only (cf. Sect. 6.1).

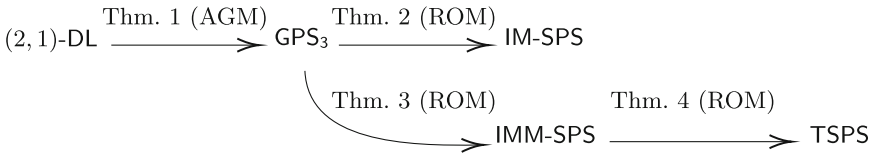


Fig. 7. The proposed constructions and underlying assumptions.

Theorem 1. *The* GPS_3 *assumption (Definition 6) holds in the asymmetric algebraic bilinear group model under the hardness of the* $(2,1)$*-DL problem (Definition 4).*

The proof is provided in the full version [32].

Theorem 2. *The indexed message SPS scheme* IM-SPS *(Fig. 6) is correct and* EUF-CiMA *secure (Definition 8) under the* GPS_3 *assumption (Definition 6) in the random oracle model.*

We first present an attack to motivate the need for uniqueness in the indexed message space. Assume there were no uniqueness requirement, and suppose the redundant check in Line 1 of the of $\mathcal{O}_{\mathsf{Sign}}(\cdot)$ oracle of Fig. 5 were not present. Then, a forger could obtain two signatures $s = h^x M_1^y$, $s' = h^x M_1'^y$ and compute a forgery $s^* = s^2/s' = h^x (M_1^2/M_1')^y$.

Proof Outline. Let $\mathcal{A}$ be a PPT adversary against the EUF-CiMA security of IM-SPS. We construct a PPT reduction $\mathcal{B}$ against the GPS_3 assumption as fol-

lows. When $\mathcal{A}$ queries the random oracle H on a fresh id, $\mathcal{B}$ queries its oracle $\mathcal{O}_0^{\mathrm{GPS}_3}()$ to obtain a random base element h, which it stores and returns to $\mathcal{A}$. When $\mathcal{A}$ queries its signing oracle $\mathcal{O}_{\mathsf{Sign}}(\cdot)$ on (id, M_1, M_2), $\mathcal{B}$ looks up $h = \mathsf{H}(id)$ and queries its oracle $\mathcal{O}_1^{\mathrm{GPS}_3}(\cdot)$ on (h, M_1, M_2) to receive $h^x M_1^y$. Finally, $\mathcal{B}$ returns the signature $\sigma = (h, h^x M_1^y)$ to $\mathcal{A}$. $\mathcal{B}$ correctly simulates the $\mathsf{EUF\text{-}CiMA}$ game, and the success probability of $\mathcal{A}$ and $\mathcal{B}$ is the same.

The attack above would violate the condition $(h, \star) \notin \mathcal{Q}_1$ in Line 3 of the $\mathcal{O}_1^{\mathrm{GPS}_3}(\cdot)$ oracle in Fig. 2. The full proof is provided in the full version [32].

4.4 Our Indexed Multi-message SPS

We extend our $\mathsf{IM\text{-}SPS}$ construction to an indexed *multi*-message SPS construction $\mathsf{IMM\text{-}SPS}$, which allows vectors of indexed messages to be signed, and prove its $\mathsf{EUF\text{-}CiMA}$ security. Extending the message space to allow vectors of any length is desirable for applications in which several attributes may be signed. The number of pairings required for verification scales linearly with the length of the message vectors, but signatures remain constant sized (2 group elements).

$\mathsf{MiDH}^{\mathsf{H}}(id, \vec{m})$

1 : $h \leftarrow \mathsf{H}(id)$
2 : **for** $j \in [1, \ell]$:
3 : $\quad M_{1j} \leftarrow h^{m_j};\ M_{2j} \leftarrow \hat{g}^{m_j}$
4 : **return** $\left(id, (\vec{M}_1, \vec{M}_2)\right)$

$\mathsf{H}(id)$

1 : **if** $\mathcal{Q}_{\mathsf{H}}[id] = \perp$:
2 : $\quad \mathcal{Q}_{\mathsf{H}}[id] \leftarrow\!\$\ \mathbb{G}_1$
3 : **return** $\mathcal{Q}_{\mathsf{H}}[id]$

Fig. 8. Encoding function of iDH multi-message space in the ROM.

We first generalize the notion of an indexed message space to the multi-message setting. In Fig. 8, we present the encoding function $\mathsf{MiDH}^{\mathsf{H}}(id, \vec{m})$ of a multi-message variant of the indexed Diffie-Hellman message space that maps, for any $\ell > 1$, ℓ-scalar message vectors $\vec{m} = (m_1, \ldots, m_\ell) \in \mathbb{Z}_p^\ell$ to 2ℓ-source group message vectors $(\vec{M}_1, \vec{M}_2) = ((M_{11}, \ldots, M_{1\ell}), (M_{21}, \ldots, M_{2\ell})) \in \mathbb{G}_1^\ell \times \mathbb{G}_2^\ell$ based on a given index id.

Definition 9 (Indexed Diffie-Hellman Multi-Message Space). *Given an asymmetric bilinear group* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$*, an index set* $\mathcal{I}$*, and a random oracle* $\mathsf{H} : \mathcal{I} \to \mathbb{G}_1$*,* $\mathcal{M}_{\mathsf{MiDH}}^{\mathsf{H}}$ *is an indexed Diffie-Hellman (DH) message space if* $\mathcal{M}_{\mathsf{MiDH}}^{\mathsf{H}} \subset \{(id, \tilde{M}) \mid id \in \mathcal{I}, \vec{m} \in \mathbb{Z}_p^\ell, \tilde{M} = \mathsf{MiDH}^{\mathsf{H}}(id, \vec{m})\}$ *and the following index uniqueness property holds: for all* $(id, \tilde{M}) \in \mathcal{M}_{\mathsf{MiDH}}^{\mathsf{H}}$*,* $(id', \tilde{M}') \in \mathcal{M}_{\mathsf{MiDH}}^{\mathsf{H}}$*,* $id = id' \Rightarrow \tilde{M} = \tilde{M}'$*.*

We define the equivalence class for each multi-message $\tilde{M} = (\vec{M}_1, \vec{M}_2) \in \tilde{\mathcal{M}}_{\mathsf{MiDH}}^{\mathsf{H}}$ *as* $\mathsf{EQ}_{\mathsf{MiDH}}(\vec{M}_1, \vec{M}_2) = \{(\vec{M}_1^r, \vec{M}_2) \mid \exists\ r \in \mathbb{Z}_p\}$*.*

$\mathsf{Setup}(1^\kappa)$

1 : $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$
2 : $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{G}_1$
3 : // select hash function
4 : $\mathsf{pp} \leftarrow ((\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}), \mathsf{H})$
5 : **return** pp

$\mathsf{KGen}(\mathsf{pp}, \ell)$

1 : $x, y_1, \ldots, y_\ell \leftarrow\$ \mathbb{Z}_p^*$
2 : $\vec{\mathsf{sk}} \leftarrow (\mathsf{sk}_0, \ldots, \mathsf{sk}_\ell) = (x, y_1, \ldots, y_\ell)$
3 : $\vec{\mathsf{vk}} \leftarrow (\mathsf{vk}_0, \mathsf{vk}_1, \boxed{\mathsf{vk}_1^\star}, \ldots, \mathsf{vk}_\ell, \boxed{\mathsf{vk}_\ell^\star})$
4 : $= (\hat{g}^x, \hat{g}^{y_1}, \boxed{g^{y_1}}, \ldots, \hat{g}^{y_\ell}, \boxed{g^{y_\ell}})$
5 : **return** $(\vec{\mathsf{sk}}, \vec{\mathsf{vk}})$

$\mathsf{Sign}(\mathsf{pp}, \vec{\mathsf{sk}}, (id, \vec{M}_1, \vec{M}_2))$

1 : $h \leftarrow \mathsf{H}(id)$
2 : **if** $\exists\, j \in [1, \ell] \;\mid\; e(h, M_{2j}) \neq e(M_{1j}, \hat{g})$:
3 : **return** $\perp$
4 : **else** : **return** $\sigma \leftarrow (h, s) = (h, h^{\mathsf{sk}_0} \prod_{j=1}^{\ell} M_{1j}^{\mathsf{sk}_j})$

$\mathsf{Verify}(\mathsf{pp}, \vec{\mathsf{vk}}, (\vec{M}_1, \vec{M}_2), \sigma)$

1 : // does not invoke H
2 : **parse** $\sigma = (h, s)$
3 : **return** $\big(h \neq 1_{\mathbb{G}_1} \;\wedge\; \{M_{1j}\}_{j\in[1,\ell]} \neq 1_{\mathbb{G}_1} \;\wedge\; \big\{e(h, M_{2j}) = e(M_{1j}, \hat{g})\big\}_{j\in[1,\ell]} \wedge$
4 : $e(h, \mathsf{vk}_0) \prod_{j=1}^{\ell} e(M_{1j}, \mathsf{vk}_j) = e(s, \hat{g})\big)$

Fig. 9. Our Indexed Multi-Message SPS Construction IMM-SPS. The additional elements in solid boxes are required for blind signing only (cf. Section 6.1).

This generalization of the indexed Diffie-Hellman message space leads us to an indexed multi-message SPS, described in Fig. 9.

Theorem 3. *The indexed multi-message SPS scheme* IMM-SPS *(Fig. 9) is correct and EUF-CiMA secure (Definition 8) under the GPS_3 assumption (Definition 6) in the random oracle model.*

The proof is provided in the full version [32].

5 Threshold Structure-Preserving Signatures

We now define the syntax and security notions for non-interactive (n, t)-Threshold Structure-Preserving Signatures (TSPS) for indexed message spaces. We then propose an efficient instantiation for an indexed Diffie-Hellman multi-message space. In an (n, t)-TSPS, the signing key is distributed among n parties, and the generation of any signature requires the cooperation of a subset of at

least t parties. We assume a centralized key generation algorithm for distributing the signing key, but a decentralized key generation protocol (DKG), such as Pedersen's DKG [75], may be used instead.

Definition 10 (Threshold Structure-Preserving Signature). *For a given security parameter κ and bilinear group $\mathcal{BG}$, an (n,t)-TSPS over indexed message space $\mathcal{M}$ consists of a tuple* $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{ParSign}, \mathsf{ParVerify}, \mathsf{Reconst}, \mathsf{Verify})$ *of PPT algorithms defined as follows:*

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$*: The setup algorithm takes the security parameter 1^κ as input and returns the public parameters* pp*.*
- $(\vec{\mathsf{sk}}, \vec{\mathsf{vk}}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp}, \ell, n, t)$*: The key generation algorithm takes the public parameters* pp *and length ℓ along with two integers $t, n \in \mathsf{poly}(1^\kappa)$ such that $1 \leq t \leq n$ as inputs. It returns two vectors of size n of signing/verification keys $\vec{\mathsf{sk}} = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$ and $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$ such that each party P_i for $i \in [n]$ receives a pair $(\mathsf{sk}_i, \mathsf{vk}_i)$ along with the global verification key* vk*.*
- $\sigma_i \leftarrow \mathsf{ParSign}(\mathsf{pp}, \mathsf{sk}_i, M)$*: The partial signing algorithm takes the public parameters* pp*, a secret signing key* sk_i*, and a message $M \in \mathcal{M}$ as inputs and returns a partial signature σ_i.*
- $0/1 \leftarrow \mathsf{ParVerify}(\mathsf{pp}, \mathsf{vk}_i, \tilde{M}, \sigma_i)$*: The partial verification algorithm is a deterministic algorithm that takes the public parameters* pp*, a verification key* vk_i*, message $\tilde{M} \in \tilde{\mathcal{M}}$, and a purported partial signature σ_i as inputs. If σ_i is a valid partial signature, it returns* 1 *(accept); else, it returns* 0 *(reject).*
- $(\sigma, \perp) \leftarrow \mathsf{Reconst}(\mathsf{pp}, \{i, \sigma_i\}_{i \in \mathcal{T}})$*: The reconstruction algorithm is a deterministic algorithm that takes public parameters* pp *and a set $\mathcal{T}$ of t partial signatures $\{i, \sigma_i\}$ with corresponding indices as inputs and returns an aggregated signature σ or $\perp$.*
- $0/1 \leftarrow \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \tilde{M}, \sigma)$*: The verification algorithm is a deterministic algorithm that takes the public parameters* pp*, the global verification key* vk*, a message $\tilde{M} \in \tilde{\mathcal{M}}$, and a purported signature σ as inputs. If σ is a valid signature, it returns* 1 *(accept); else, it returns* 0 *(reject).*

Three main security properties for TSPS defined over indexed message spaces are *partial verification correctness*, *evaluation correctness*, and *threshold existential unforgeability against chosen indexed message attack* (Threshold $\mathsf{EUF\text{-}CiMA}$). Intuitively, partial verification correctness means that any correctly generated partial signature via the $\mathsf{ParSign}$ algorithm passes the $\mathsf{ParVerify}$ verification checks, and evaluation correctness means that the $\mathsf{Reconst}$ algorithm for a set of well-formed partial signatures $\{i, \sigma_i\}_{i \in \mathcal{T}}$ (meaning all with the same index, on a message M) results in a valid aggregated signature σ.

Definition 11 (Partial Verification Correctness). *An (n,t)-TSPS scheme satisfies partial verification correctness if for all correctly indexed messages $M \in \mathcal{M}$,* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa), (\vec{\mathsf{sk}}, \vec{\mathsf{vk}}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp}, \ell, n, t)$ *and $i \in [1, n]$ that*

$$\Pr\left[\mathsf{ParVerify}(\mathsf{pp}, \mathsf{vk}_i, \tilde{M}, \mathsf{ParSign}(\mathsf{pp}, \mathsf{sk}_i, M)) = 1\right] = 1\ .$$

Definition 12 (Evaluation Correctness). *An (n,t)-TSPS scheme satisfies evaluation correctness if for all correctly indexed messages $M \in \mathcal{M}$, $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa), (\vec{\mathsf{sk}}, \vec{\mathsf{vk}}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp}, \ell, n, t)$ and $\mathcal{T} \subseteq [1,n], |\mathcal{T}| = t$ that*

$$\Pr\left[\sigma \leftarrow \mathsf{Reconst}(\mathsf{pp}, \{i, \mathsf{ParSign}(\mathsf{pp}, \mathsf{sk}_i, M)\}_{i \in \mathcal{T}}) : \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \tilde{M}, \sigma) = 1\right] = 1.$$

Threshold Unforgeability. We next define the notion of threshold unforgeability for non-interactive (n,t)-TSPS schemes. The Threshold EUF-CiMA game is defined formally in Fig. 10. Given a set of party indices $\mathcal{P} = \{1, \ldots, n\}$, we assume that the adversary can corrupt up to $t-1$ parties and that there is at least one honest party. We denote the set of corrupt parties by $\mathcal{C}$ and the set of honest parties by $\mathcal{H} = \mathcal{P} \setminus \mathcal{C}$.

$\mathbf{G}_{\mathcal{A}}^{\text{T-EUF-CiMA}}(1^\kappa)$

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$
$\mathcal{C} \leftarrow\$\ \mathcal{A}(\mathsf{pp})$ // set of corrupt signers
if $\mathcal{C} \notin [1,n] \ \vee\ |\mathcal{C}| > t-1$:
  return $\bot$
else : $\mathcal{H} \leftarrow [1,n] \setminus \mathcal{C}$ // set of honest signers
  $(\vec{\mathsf{sk}}, \vec{\mathsf{vk}}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp}, \ell, n, t)$
  $(\tilde{M}^*, \sigma^*) \leftarrow\$\ \mathcal{A}^{\mathcal{O}_{\mathsf{PSign}}}\left(\{\mathsf{sk}_i\}_{i \in \mathcal{C}}, \vec{\mathsf{vk}}, \mathsf{vk}\right)$
  return $(\tilde{M}^* \notin \mathcal{Q}_{\mathsf{EQ}} \ \wedge\ \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \tilde{M}^*, \sigma^*))$

$\mathcal{O}_{\mathsf{PSign}}(k, id, \tilde{M})$ // $M = (id, \tilde{M})$

if $(k \notin \mathcal{H} \ \vee\ (k, id, \star) \in \mathcal{Q}_{\mathsf{S}} \ \vee\ (\star, id, \tilde{M}') \in \mathcal{Q}_{\mathsf{S}}, \tilde{M}' \neq \tilde{M})$:
  return $\bot$
else : $\sigma_k \leftarrow \mathsf{ParSign}(\mathsf{pp}, \mathsf{sk}_k, (id, \tilde{M}))$
  $\mathcal{Q}_{\mathsf{S}} \leftarrow \mathcal{Q}_{\mathsf{S}} \cup \{(k, id, \tilde{M})\}$
  $\mathcal{Q}_{\mathsf{EQ}} \leftarrow \mathcal{Q}_{\mathsf{EQ}} \cup \{\mathsf{EQ}(\tilde{M})\}$
  return σ_k

Fig. 10. Game $\mathbf{G}_{\mathcal{A}}^{\text{T-EUF-CiMA}}(1^\kappa)$.

In the unforgeability game, the challenger generates public parameters pp and returns them to the adversary. The adversary chooses the set of corrupted participants $\mathcal{C}$. The challenger then runs KGen to derive the global verification key vk, the individual verification keys $\{\mathsf{vk}_i\}_{i=1}^n$, and the secret signing shares $\{\mathsf{sk}_i\}_{i=1}^n$. It returns vk, $\{\mathsf{vk}_i\}_{i=1}^n$, and the set of corrupt signing shares $\{\mathsf{sk}_j\}_{j \in \mathcal{C}}$ to the adversary. We assume the adversary maintains state before and after KGen.

After key generation, the adversary can request partial signatures on messages of its choosing from honest signers by querying oracle $\mathcal{O}_{\mathsf{PSign}}(\cdot)$.

$\mathsf{Setup}(1^\kappa)$

1 : $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$; $\mathsf{H} : \{0,1\}^* \to \mathbb{G}_1$ // select hash function

2 : **return** $\mathsf{pp} \leftarrow ((\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}), \mathsf{H})$

$\mathsf{KGen}(\mathsf{pp}, \ell, n, t)$

1 : $x, y_1, \ldots, y_\ell \leftarrow\$ \mathbb{Z}_p^*$

2 : $\vec{x} \leftarrow\$ \mathsf{Share}(x, p, n, t), \{\vec{y}_j \leftarrow\$ \mathsf{Share}(y_j, p, n, t)\}_{j\in[1,\ell]}$

3 : **for** $i \in [1, n]$:

4 : $\quad \mathsf{sk}_i \leftarrow (\mathsf{sk}_{i0}, \mathsf{sk}_{i1}, \ldots, \mathsf{sk}_{i\ell}) = (x_i, y_{i1}, \ldots, y_{i\ell})$

5 : $\quad \mathsf{vk}_i \leftarrow (\mathsf{vk}_{i0}, \mathsf{vk}_{i1}, \boxed{\mathsf{vk}^\star_{i1}}, \ldots, \mathsf{vk}_{i\ell}, \boxed{\mathsf{vk}^\star_{i\ell}}) = (\hat{g}^{x_i}, \hat{g}^{y_{i1}}, \boxed{g^{y_{i1}}}, \ldots, \hat{g}^{y_{i\ell}}, \boxed{g^{y_{i\ell}}})$

6 : $\vec{\mathsf{sk}} \leftarrow (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$

7 : $\vec{\mathsf{vk}} \leftarrow (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$

8 : $\mathsf{vk} \leftarrow (\mathsf{vk}_{00}, \mathsf{vk}_{01}, \boxed{\mathsf{vk}^\star_{01}}, \ldots, \mathsf{vk}_{0\ell}, \boxed{\mathsf{vk}^\star_{0\ell}}) = (\hat{g}^{x}, \hat{g}^{y_1}, \boxed{g^{y_1}}, \ldots, \hat{g}^{y_\ell}, \boxed{g^{y_\ell}})$

9 : **return** $(\vec{\mathsf{sk}}, \vec{\mathsf{vk}}, \mathsf{vk})$

$\mathsf{ParSign}\left(\mathsf{pp}, \mathsf{sk}_i, (id, \vec{M}_1, \vec{M}_2)\right)$

1 : $h \leftarrow \mathsf{H}(id)$

2 : **if** $\exists\, j \in [1, \ell] \mid$

3 : $e(h, M_{2j}) \neq e(M_{1j}, \hat{g})$:

4 : $\quad$ **return** $\perp$

5 : **else** : $s_i \leftarrow h^{\mathsf{sk}_{i0}} \prod_{j=1}^{\ell} {M_{1j}}^{\mathsf{sk}_{ij}}$

6 : $\quad$ **return** $\sigma_i \leftarrow (h, s_i)$

$\mathsf{ParVerify}\left(\mathsf{pp}, \mathsf{vk}_i, (\vec{M}_1, \vec{M}_2), \sigma_i\right)$

1 : // does not invoke H

2 : **parse** $\sigma_i = (h_i, s_i)$

3 : **return** $\big(h_i \neq 1_{\mathbb{G}_1} \wedge$

4 : $\quad \{M_{1j}\}_{j\in[1,\ell]} \neq 1_{\mathbb{G}_1} \wedge$

5 : $\quad \{e(h_i, M_{2j}) = e(M_{1j}, \hat{g})\}_{j\in[1,\ell]} \wedge$

6 : $\quad e(h_i, \mathsf{vk}_{i0}) \prod_{j=1}^{\ell} e(M_{1j}, \mathsf{vk}_{ij}) = e(s_i, \hat{g})$

$\mathsf{Reconst}\left(\mathsf{pp}, \vec{\mathsf{vk}}, (\vec{M}_1, \vec{M}_2), \{i, \sigma_i\}_{i\in\mathcal{T}}\right)$

1 : **parse** $\sigma_i = (h_i, s_i)$

2 : **if** $\exists\, i, j \in \mathcal{T}, i \neq j \mid h_i \neq h_j$

3 : $\vee\ \exists\, i \in \mathcal{T} \mid$

4 : $\mathsf{ParVerify}(\mathsf{pp}, \mathsf{vk}_i, (\vec{M}_1, \vec{M}_2), \sigma_i) = 0$

5 : $\quad$ **return** $\perp$

6 : **else** : $h \leftarrow h_i$

7 : $\quad$ **return** $\sigma \leftarrow (h, s) = (h, \prod_{i\in\mathcal{T}} s_i^{\lambda_i})$

$\mathsf{Verify}\left(\mathsf{pp}, \mathsf{vk}, (\vec{M}_1, \vec{M}_2), \sigma\right)$

1 : // does not invoke H

2 : **parse** $\sigma = (h, s)$

3 : **return** $\big(h \neq 1_{\mathbb{G}_1} \wedge$

4 : $\quad \{M_{1j}\}_{j\in[1,\ell]} \neq 1_{\mathbb{G}_1} \wedge$

5 : $\quad \{e(h, M_{2j}) = e(M_{1j}, \hat{g})\}_{j\in[1,\ell]} \wedge$

6 : $\quad e(h, \mathsf{vk}_{00}) \prod_{j=1}^{\ell} e(M_{1j}, \mathsf{vk}_{0j}) = e(s, \hat{g})$

Fig. 11. Our Threshold SPS Construction TSPS. The additional elements in solid boxes are required for blind signing only (cf. Sect. 6.1).

The adversary wins if it can produce a valid forgery $(\tilde{M}^*, \sigma^*)$ with respect to the global verification key vk representing the set of n signers, on a message $\tilde{M}^*$ for which no equivalent $\tilde{M}^{*'}$ has been previously queried to $\mathcal{O}_{\mathsf{PSign}}(\cdot)$.

Definition 13 (Threshold EUF-CiMA). *A non-interactive (n,t)-TSPS scheme over indexed message space $\mathcal{M}$ is Threshold EUF-CiMA secure if for all PPT adversaries $\mathcal{A}$ playing game $\mathbf{G}^{T\text{-}EUF\text{-}CiMA}$ (Fig. 10), there exists a negligible function ν such that*

$$Adv_{\mathcal{A}}^{T\text{-}EUF\text{-}CiMA}(\kappa) = \Pr\left[\mathbf{G}_{\mathcal{A}}^{T\text{-}EUF\text{-}CiMA}(1^\kappa) = 1\right] \leq \nu(\kappa) \ .$$

5.1 Our Indexed Multi-message TSPS

In Fig. 11, we present our (n,t)-TSPS scheme TSPS over an indexed Diffie-Hellman multi-message space $\mathcal{M}_{\mathsf{MiDH}}^{\mathsf{H}}$, as defined in Fig. 8.

5.2 Security of TSPS

Theorem 4. *The indexed multi-message (n,t)-Threshold SPS scheme TSPS is correct and Threshold EUF-CiMA secure (Definition 13) in the random oracle model under the EUF-CiMA security of $\mathsf{IMM\text{-}SPS}$ (Theorem 3).*

Proof. **Correctness.** We first show that the proposed TSPS satisfies partial verification correctness (Definition 11), i.e., any correctly generated partial signature via the $\mathsf{ParSign}$ algorithm passes the $\mathsf{ParVerify}$ verification checks. Indeed, for all $i \in [1,n]$ and correctly indexed messages $M = (id, \vec{M}_1, \vec{M}_2) \in \mathcal{M}_{\mathsf{MiDH}}^{\mathsf{H}}$, we have:

$$e(h, \mathsf{vk}_{i0}) \prod_{j=1}^{\ell} e(M_{1j}, \mathsf{vk}_{ij}) = e(h, \hat{g}^{x_i}) \prod_{j=1}^{\ell} e(M_{1j}, \hat{g}^{y_{ij}}) e(h^{x_i} \prod_{j=1}^{\ell} M_{1j}^{y_{ij}}, \hat{g}) = e(s_i, \hat{g}) \ .$$

Next, we show that TSPS satisfies evaluation correctness (Definition 12); that is, the $\mathsf{Reconst}$ algorithm for a set of partial signatures $\{i, \sigma_i\}_{i \in \mathcal{T}}, \mathcal{T} \subseteq [1,n], |\mathcal{T}| = t$, on a message $M = (id, \vec{M}_1, \vec{M}_2)$ with the same $h \leftarrow \mathsf{H}(id)$ results in a valid aggregated signature $\sigma = (h, s)$. Indeed,

$$s = \prod_{i \in \mathcal{T}} s_i^{\lambda_i} = \prod_{i \in \mathcal{T}} (h^{\mathsf{sk}_{i0}} \prod_{j=1}^{\ell} M_{1j}^{\mathsf{sk}_{ij}})^{\lambda_i} = h^{\sum_{i \in \mathcal{T}} \mathsf{sk}_{i0} \lambda_i} \prod_{j=1}^{\ell} M_{1j}^{\sum_{i \in \mathcal{T}} \mathsf{sk}_{ij} \lambda_i} = h^{\mathsf{sk}_0} \prod_{j=1}^{\ell} M_{1j}^{\mathsf{sk}_j}$$

where λ_i is the Lagrange coefficient for party P_i with respect to the signing set $\mathcal{T}$. Next, we show that verification holds for the above aggregated signature σ on message $\tilde{M} = (\vec{M}_1, \vec{M}_2)$. Indeed, $\forall\ j \in [1,\ell]$ we have that $e(h, M_{2j}) = e(h, \hat{g}^{m_j}) = e(h^{m_j}, \hat{g}) = e(M_{1j}, \hat{g})$ and

$$e(h, \mathsf{vk}_0) \prod_{j=1}^{\ell} e(M_{1j}, \mathsf{vk}_j) = e(h, \hat{g}^{x}) \prod_{j=1}^{\ell} e(M_{1j}, \hat{g}^{y_j}) = e(h^{x} \prod_{j=1}^{\ell} M_{1j}^{y_j}, \hat{g}) = e(s, \hat{g}) \ .$$

Note that successful partial signature verification using ParVerify and consistency of h guarantee successful reconstruction.

Need for Uniqueness. The hypothetical attack described after Theorem 2 also works with a partial signing oracle $\mathcal{O}_{\mathsf{PSign}}(\cdot)$. Assume an (n,t)-TSPS with $n > 2t$, and suppose there were no uniqueness requirement for the message space and that the redundant check in Line 2 of the $\mathcal{O}_{\mathsf{PSign}}(\cdot)$ oracle of Fig. 10 were not present. Then, a forger could obtain $2t$ partial signatures to reconstruct signatures $s = h^x M_1^y$, $s' = h^x M_1'^y$ and compute a forgery $s^* = s^2/s' = h^x(M_1^2/M_1')^y$ that is a valid signature on fresh message M_1^2/M_1' .

Threshold EUF-CiMA. Our proof of security for TSPS resembles that of threshold BLS in [18]. We wish to show that if there exists a PPT adversary $\mathcal{A}$ that breaks the Threshold EUF-CiMA security (Fig. 10) of TSPS with non-negligible probability, then we can construct a PPT adversary $\mathcal{B}$ that breaks the EUF-CiMA security (Fig. 5) of the underlying IMM-SPS scheme (Fig. 6) with non-negligible probability.

Suppose there exists such a PPT adversary $\mathcal{A}$. Then, running $\mathcal{A}$ as a subroutine, we construct a reduction $\mathcal{B}$ breaking the EUF-CiMA security of IMM-SPS as follows.

The reduction $\mathcal{B}$ is responsible for simulating oracle responses for queries to $\mathcal{O}_{\mathsf{PSign}}(\cdot)$ and H. Let $\mathcal{Q}_{\mathsf{H}}$ be the set of H queries id and their responses. $\mathcal{B}$ may program the random oracle H. Let $\mathcal{Q}_{\mathsf{S}}$ be the set of $\mathcal{O}_{\mathsf{PSign}}(\cdot)$ queries $(k, id, \tilde{M})$ and $\mathcal{Q}_{\mathsf{EQ}}$ the set of equivalence classes of messages $\tilde{M}$. $\mathcal{B}$ initializes $\mathcal{Q}_{\mathsf{H}}, \mathcal{Q}_{\mathsf{S}}, \mathcal{Q}_{\mathsf{EQ}}$ to the empty set.

Initialization. $\mathcal{B}$ takes as input public parameters $\mathsf{pp} \leftarrow (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g})$ and an IMM-SPS verification key vk'. In the EUF-CiMA game, $\mathcal{B}$ has access to oracles $\mathcal{O}'_{\mathsf{Sign}}(\cdot)$ and H'. $\mathcal{B}$ uses $\mathsf{vk}' = (\mathsf{vk}'_{00}, \mathsf{vk}'_{01}, \mathsf{vk}'^{\star}_{01}, \ldots, \mathsf{vk}'_{0\ell}, \mathsf{vk}'^{\star}_{0\ell})$ as the TSPS verification key $\mathsf{vk} = (\mathsf{vk}_{00}, \mathsf{vk}_{01}, \mathsf{vk}^{\star}_{01}, \ldots, \mathsf{vk}_{0\ell}, \mathsf{vk}^{\star}_{0\ell})$.

Simulating Key Generation. $\mathcal{B}$ simulates the key generation algorithm as follows.

- $\mathcal{B}$ defines the pair of secret/verification keys of the corrupted parties $P_i, i \in \mathcal{C}$, as follows. Assume without loss of generality that $|\mathcal{C}| = t - 1$. For all $i \in \mathcal{C}$, $\mathcal{B}$ samples random values $x_{i0}, y_{i1}, \ldots, y_{i\ell} \leftarrow\!\!\$\, (\mathbb{Z}_p^*)^{\ell+1}$ and defines party P_i's secret key as $\mathsf{sk}_i \leftarrow (\mathsf{sk}_{i0}, \mathsf{sk}_{i1}, \ldots, \mathsf{sk}_{i\ell}) = (x_{i0}, y_{i1}, \ldots, y_{i\ell})$ and the corresponding verification key as $\mathsf{vk}_i \leftarrow (\mathsf{vk}_{i0}, \mathsf{vk}_{i1}, \mathsf{vk}^{\star}_{i1}, \ldots, \mathsf{vk}_{i\ell}, \mathsf{vk}^{\star}_{i\ell}) = (\hat{g}^{x_{i0}}, \hat{g}^{y_{i1}}, g^{y_{i1}}, \ldots, \hat{g}^{y_{i\ell}}, g^{y_{i\ell}})$.
- To generate the verification key of the honest parties $P_k, k \in \mathcal{H}, \mathcal{H} = [1,n] \setminus \mathcal{C}$, $\mathcal{B}$ proceeds as follows:
  1. For all $i \in \tilde{\mathcal{T}} := \mathcal{C} \cup \{0\}$, it computes the Lagrange polynomials evaluated at point k:
  $$\tilde{\lambda}_{ki} = L_i^{\tilde{\mathcal{T}}}(k) = \prod_{j \in \tilde{\mathcal{T}} j \neq i} \frac{(j-k)}{(j-i)} \quad . \tag{3}$$
  2. It takes the verification keys of corrupted parties $\{\mathsf{vk}_i\}_{i \in \mathcal{C}}$ and the global verification key vk and then computes

$$\begin{aligned}\mathsf{vk}_k &= (\mathsf{vk}_{k0}, \mathsf{vk}_{k1}, \mathsf{vk}^\star_{k1}, \ldots, \mathsf{vk}_{k\ell}, \mathsf{vk}^\star_{k\ell}) \\ &= \Big(\mathsf{vk}_{00}^{\tilde{\lambda}_{k0}} \prod_{i\in\mathcal{C}} \mathsf{vk}_{i0}^{\tilde{\lambda}_{ki}}, \mathsf{vk}_{01}^{\tilde{\lambda}_{k0}} \prod_{i\in\mathcal{C}} \mathsf{vk}_{i1}^{\tilde{\lambda}_{ki}}, {\mathsf{vk}^\star_{01}}^{\tilde{\lambda}_{k0}} \prod_{i\in\mathcal{C}} {\mathsf{vk}^\star_{i1}}^{\tilde{\lambda}_{ki}}, \ldots, \\ &\quad \mathsf{vk}_{0\ell}^{\tilde{\lambda}_{k0}} \prod_{i\in\mathcal{C}} \mathsf{vk}_{i\ell}^{\tilde{\lambda}_{ki}}, {\mathsf{vk}^\star_{0\ell}}^{\tilde{\lambda}_{k0}} \prod_{i\in\mathcal{C}} {\mathsf{vk}^\star_{i\ell}}^{\tilde{\lambda}_{ki}}\Big) .\end{aligned}$$

$\mathcal{B}$ returns the global verification key vk, $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$, and secret keys $\{\mathsf{sk}_j\}_{j\in\mathcal{C}}$ to $\mathcal{A}$.

Simulating Random Oracle $\mathsf{H}(id)$*:* When $\mathcal{A}$ queries H on index id, if $\mathcal{Q}_\mathsf{H}[id] = \perp$, then $\mathcal{B}$ queries $\mathsf{H}'(id)$, receives a base element h, and sets $\mathcal{Q}_\mathsf{H}[id] \leftarrow h$. $\mathcal{B}$ returns $\mathcal{Q}_\mathsf{H}[id]$ to $\mathcal{A}$.

Simulating Signing Oracle $\mathcal{O}_{\mathsf{PSign}}(k, id, \tilde{M})$*:* When $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{PSign}}(\cdot)$ on $(k, id, \tilde{M})$ for honest party identifier $k \in \mathcal{H}$ and message $M = (id, \tilde{M}) = (id, \vec{M}_1, \vec{M}_2)$, if $k \notin \mathcal{H}$ or $(k, id, \star) \in \mathcal{Q}_\mathsf{S}$ or $(\star, id, \tilde{M}') \in \mathcal{Q}_\mathsf{S}, \tilde{M}' \neq \tilde{M}$, $\mathcal{B}$ returns $\perp$. Otherwise, $\mathcal{B}$ does the following:

1. $\mathcal{B}$ looks up $h = \mathcal{Q}_\mathsf{H}[id]$, queries $\mathcal{O}'_{\mathsf{Sign}}(id, \vec{M}_1, \vec{M}_2)$, and receives the signature $\sigma_0 = (h, s_0)$.
2. For all $i \in \mathcal{C}$, $\mathcal{B}$ computes the partial signatures $\sigma_i = (h, s_i) = (h, h^{\mathsf{sk}_{i0}} \prod_{j=1}^{\ell} M_{1j}^{\mathsf{sk}_{ij}})$, as it knows the secret keys of corrupted parties.
3. For all $i \in \tilde{\mathcal{T}} = \mathcal{C} \cup \{0\}$, $\mathcal{B}$ computes Lagrange coefficients $\tilde{\lambda}_{ki}$ as in Equation (3).
4. $\mathcal{B}$ updates $\mathcal{Q}_\mathsf{S} \leftarrow \mathcal{Q}_\mathsf{S} \cup \{(k, id, \tilde{M})\}$ and $\mathcal{Q}_{\mathsf{EQ}} \leftarrow \mathcal{Q}_{\mathsf{EQ}} \cup \{\mathsf{EQ}(\tilde{M})\}$.
5. $\mathcal{B}$ computes $(h, s_k) = (h, s_0^{\tilde{\lambda}_{k0}} \prod_{i\in\mathcal{C}} s_i^{\tilde{\lambda}_{ki}})$ and returns $\sigma_k = (h, s_k)$ to $\mathcal{A}$.

Output. At the end of the game, $\mathcal{A}$ produces a valid forgery $\sigma^* = (h^*, s^*)$ on message $\tilde{M}^* = (\vec{M}_1^*, \vec{M}_2^*)$, and $\mathcal{B}$ returns $(\tilde{M}^*, \sigma^*)$ as its forgery.

$\mathcal{B}$ correctly simulates key generation and $\mathcal{A}$'s hash and signing queries. Since $\mathcal{A}$'s forgery satisfies $\tilde{M}^* \notin \mathcal{Q}_{\mathsf{EQ}}$ and $\mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \tilde{M}^*, \sigma^*) = 1$, $\mathcal{B}$'s winning conditions are also satisfied and $Adv^{\mathsf{T\text{-}EUF\text{-}CiMA}}_{\mathsf{TSPS},\mathcal{A}}(\kappa) \leq Adv^{\mathsf{EUF\text{-}CiMA}}_{\mathsf{IMM\text{-}SPS},\mathcal{B}}(\kappa)$. □

6 Applications to Threshold-Issuance Anonymous Credentials

Threshold-Issuance Anonymous Credential (TIAC) systems are a prime use-case of threshold SPS. TIAC systems, defined by Sonnino et al. [82], are used in various applications [64,83]. A TIAC is an anonymous credential scheme that enables a group of signers (or issuers) to jointly sign a blind message, i.e., issue a credential, without learning the original message. The core ingredient is a blind signing protocol for the used threshold signature scheme. Besides the threshold

signature, this protocol relies on two main cryptographic primitives: NIZKs and commitment schemes, defined in Appendix A.3 and Appendix A.2, respectively.

The TIAC protocol of [82], known as Coconut, lacks a rigorous security proof. Recently, Rial and Piotrowska [77] conducted a security analysis that required some modifications to the original Coconut scheme, resulting in Coconut^{++}. Coconut and Coconut^{++} are based on a threshold Pointcheval-Sanders signature scheme that supports an efficient blind signing protocol.

6.1 Blind Signing for TSPS

In Fig. 12, we show that our TSPS construction also supports threshold blind signing. In addition to the TSPS parameters, the public parameters pp now contain the common reference string (CRS) of a NIZK and the public parameters of a commitment scheme.

For intuition, we note that in $\mathsf{PrepareBlindSign}$, the index is computed as a commitment to $\vec{m}$, using the generalized Pedersen commitment scheme. The single messages are also committed in a Pedersen commitment, where one commitment parameter is computed on the fly via a random oracle as $h = \mathsf{H}(id)$. The hiding property of commitments and the zero-knowledge property of NIZK ensure the blindness.

We note that the construction in Fig. 12 follows the blind signing protocol for Coconut^{++} closely, with only minor syntactical changes due to the indexed DH message space (highlighted in the figure). Consequently, the validity of the blinding operations readily follows from that of Coconut^{++}. The key generation phase is the same as in Fig. 11.

6.2 Removing Rewinding Extractors in TIAC

The TIAC constructions Coconut and Coconut^{++} combine threshold signatures (with blind signing) with generalized Schnorr proofs [78] turned into extractable (knowledge-sound) NIZK proofs via the Fiat-Shamir (FS) heuristic [43] in the random oracle model. This, however, is problematic if used within the universal composability (UC) framework [24], as extractability for such NIZK proofs requires rewinding. For instance, Coconut^{++} is modeled in the UC framework but requires rewinding to prove that it realizes $\mathcal{F}_{\text{AC}}$ [77, Theorem 3]. This, in turn, makes the formal security guarantees in the UC framework questionable.

Fischlin's framework [44], also in the random oracle model, is a well-known technique to avoid rewinding. However, this adds significant overhead that negatively affects its practical applicability. Groth-Sahai (GS) NIZK proofs [60] are an efficient alternative NIZK proof system. GS proofs are secure in the standard model and support straight-line extraction of the witnesses, i.e., avoid the rewinding required by the Fischlin transform. This makes them particularly attractive if one is interested in achieving composable security, e.g., UC security. We note that there are known transformations like [27,57,59] to make GS proofs UC secure despite their malleability. However, GS proofs can only extract group elements.

$\mathsf{PrepareBlindSign}(\mathsf{pp}, \vec{m})$ // $\mathsf{pp} = (\mathsf{pp}_c, \mathsf{CRS}, \mathsf{H})$

parse $\vec{m} = (m_1, \ldots, m_\ell)$

$\omega \leftarrow\$ \mathbb{Z}_p^*,\ id \leftarrow \mathsf{Com}(\mathsf{pp}_c, \vec{m}, \omega) = G_0^{\omega} \prod_{i=1}^{\ell} G_i^{m_i}$

$\left(id, (\vec{M}_1, \vec{M}_2)\right) \leftarrow \mathsf{MiDH}^{\mathsf{H}}(id, \vec{m})$

for $j \in [1, \ell]$:

  $\omega_{1j}, \omega_{2j} \leftarrow\$ \mathbb{Z}_p^*$

  $(\mathsf{cm}_{1j}, \mathsf{cm}_{2j}) \leftarrow \left(g^{\omega_{1j}} M_{1j}\ ,\ \hat{g}^{\omega_{2j}} M_{2j} \right)$

$\vec{\mathsf{cm}} = \{(\mathsf{cm}_{1j}, \mathsf{cm}_{2j})\}_{j=1}^{\ell}$

$\Omega \leftarrow (\omega, \omega_{11}, \omega_{21}, \ldots, \omega_{1\ell}, \omega_{2\ell})$

$\pi_s \leftarrow \mathsf{NIZK.Prove}\Big\{ \Omega, \vec{m} \mid id = G_0^{\omega} \prod_{i=1}^{\ell} G_i^{m_i} \ \wedge$

$\qquad \left\{\mathsf{cm}_{1j} = g^{\omega_{1j}} \mathsf{H}(id)^{m_j}\right\}_{j=1}^{\ell} \ \wedge\ \left\{\mathsf{cm}_{2j} = g^{\omega_{2j}} \hat{g}^{m_j}\right\}_{j=1}^{\ell} \Big\}$

return $(\Omega, id, \vec{\mathsf{cm}}, \pi_s)$

$\mathsf{BlindSign}(\mathsf{pp}, \mathsf{sk}_i, id, \vec{\mathsf{cm}}, \pi_s)$

parse $\mathsf{sk}_i = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$

$h \leftarrow \mathsf{H}(id)$

if $\mathsf{NIZK.Verify}(\mathsf{CRS}, (id, \vec{\mathsf{cm}}, h), \pi_s) = 0$:

  return $\perp$

else : $\bar{s}_i \leftarrow h^{\mathsf{sk}_{i0}} \prod_{j=1}^{\ell} \mathsf{cm}_{1j}{}^{\mathsf{sk}_{ij}}$

  return $\bar{\sigma}_i \leftarrow (h, \bar{s}_i)$

$\mathsf{AggCred}(\mathsf{pp}, \{i, \bar{\sigma}_i\}_{i \in \mathcal{T}})$

parse $\bar{\sigma}_i = (h_i, \bar{s}_i)$

if $\exists\ i, j \in \mathcal{T}, i \neq j \mid h_i \neq h_j$: **return** $\perp$

else : $h \leftarrow h_i$

  return $\bar{\sigma} \leftarrow (h, \bar{s}) = (h, \prod_{i \in \mathcal{T}} s_i^{\lambda_i})$

$\mathsf{UnBlind}(\mathsf{pp}, \mathsf{vk}, \bar{\sigma}, \Omega)$

parse $\bar{\sigma} = (h, \bar{s})$

return $\sigma := (h, s) \leftarrow (h, \bar{s} \prod_{j=1}^{\ell} (g^{y_j})^{-\omega_j})$

Fig. 12. A Threshold Blind Signature with straight-line extraction. Grey (Color figure online) boxes mark the changes from Coconut^{++}. Algorithms and notation are defined in Appendices A.2 to A.4.

Towards achieving efficient straight-line extraction without the need of rewinding, we propose to replace the blind issuance threshold Pointcheval-Sanders signature of Coconut^{++} with our blind issuance TSPS. We make the reasonable assumption that the scalar messages (attributes in the TIAC) come from some polynomially bounded message space, e.g., in practice, attributes can be encoded in small scalar values. This modification enables us to provide a GS proof of a valid signature for the showing of a credential with non-revealed messages. Noticing that GS NIZKs are commit-and-proof NIZKs, we can use an additional Schnorr NIZK obtained via Fiat-Shamir to prove a predicate over the scalar messages in the GS commitments. The interesting point is that the latter NIZK only needs to be sound, but does not need to be extractable, as GS commitments can be perfectly binding. Thus, we can avoid rewinding and, due to the polynomially bounded message space, we can extract the scalar messages (attributes in TIAC) efficiently from the straight-line extracted messages from the commitments of the GS proof.

7 Conclusion and Open Problems

In this work, we introduce the notion of a threshold structure-preserving signature (TSPS) and present an efficient fully non-interactive TSPS construction. We prove that the proposed TSPS is secure under a new variant of the generalized Pointcheval-Sanders (PS) assumption in the random oracle model. We have shown that our TSPS can be used as a drop-in replacement in TIAC systems to remove the need for rewinding extractors.

While we use a message indexing method in order to construct a non-interactive scheme, a non-interactive TSPS without indexing is an interesting open problem. Moreover, it is interesting to construct schemes that rely on weaker assumptions and avoid the use of the random oracle model. When it comes to the security model, the following two challenging problems remain open: obtaining security under adaptive corruptions more tightly than via a guessing argument from static corruptions, and achieving the strongest notion possible for fully non-interactive schemes (TS-UF-1) [14]. In general, we believe this work can open a new line of research for structure-preserving multi-party protocols, such as threshold structure-preserving encryption. Moreover, we expect that TSPS will have further applications beyond TIAC systems.

Acknowledgments. We would like to thank the anonymous reviewers for their valuable comments, and Behzad Abdolmaleki, Daniele Cozzo and Hyoseung Kim for their suggestions. Elizabeth Crites was supported by Input Output through their funding of the Blockchain Technology Lab at the University of Edinburgh. The work of Markulf Kohlweiss was done in part while visiting COSIC, KU Leuven. Mahdi Sedaghat and Bart Preneel were supported in part by the Research Council KU Leuven C1 on Security and Privacy for Cyber-Physical Systems and the Internet of Things with contract number C16/15/058 and by CyberSecurity Research Flanders with reference number VR20192203. Daniel Slamanig was supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871473 (KRAKEN) and

No. 861696 (Labyrinth) and by the Austrian Science Fund (FWF) and netidee SCIENCE under grant agreement P31621-N38 (Profet).

A Additional Definitions and Assumptions

A.1 Digital Signatures

Definition 14 (Digital Signature). *A digital signature scheme over message space $\mathcal{M}$ is a tuple of the following polynomial-time algorithms:*

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$*: Setup is a probabilistic algorithm which takes as input the security parameter 1^κ and outputs the set of public parameters* pp.
- $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$*: Key generation is a probabilistic algorithm which takes as input* pp *and outputs a pair of signing/verification keys* $(\mathsf{sk}, \mathsf{vk})$.
- $\sigma \leftarrow \mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, m)$*: The signing algorithm takes as input* pp*, a secret signing key* sk*, and a message $m \in \mathcal{M}$, and outputs a signature σ.*
- $0/1 \leftarrow \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, m, \sigma)$*: Verification is a deterministic algorithm which takes as input* pp*, a public verification key* vk*, a message $m \in \mathcal{M}$, and a purported signature σ, and outputs either 0 (reject) or 1 (accept).*

The primary security requirements for a digital signature scheme are *correctness* and *existential unforgeability against chosen message attack* (EUF-CMA).

Definition 15 (Correctness). *A digital signature is correct if we have:*

$$\Pr\left[\begin{array}{l}\forall\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa), (\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp}), m \in \mathcal{M} : \\ \mathsf{Verify}\left(\mathsf{pp}, \mathsf{vk}, m, \mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, m)\right) = 1\end{array}\right] \geq 1 - \nu(\kappa) \ .$$

Definition 16 (Existential Unforgeability under Chosen Message Attack (EUF-CMA)). *A digital signature scheme over message space $\mathcal{M}$ is EUF-CMA secure if for all PPT adversaries $\mathcal{A}$ playing game $\mathbf{G}^{\mathsf{EUF\text{-}CMA}}$ (Fig. 13), there exists a negligible function ν such that*

$$Adv_{\mathcal{A}}^{\mathsf{EUF\text{-}CMA}}(\kappa) = \Pr\left[\,\mathbf{G}_{\mathcal{A}}^{\mathsf{EUF\text{-}CMA}}(1^\kappa) = 1\right] \leq \nu(\kappa) \ .$$

$\mathbf{G}_{\mathcal{A}}^{\mathrm{EUF\text{-}CMA}}(1^\kappa)$	$\mathcal{O}_{\mathsf{Sign}}(m)$
1 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$	1 : $\sigma \leftarrow \mathsf{Sign}\,(\mathsf{pp}, \mathsf{sk}, m)$
2 : $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$	2 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{m\}$
3 : $(m^*, \sigma^*) \leftarrow\!\!\$\ \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(\mathsf{pp}, \mathsf{vk})$	3 : **return** σ
4 : **return** $(m^* \notin \mathcal{Q} \ \wedge\ \mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, m^*, \sigma^*))$	

Fig. 13. The EUF-CMA security game.

A.2 Commitment Schemes

Definition 17 (Commitment Scheme). *A commitment scheme over message space $\mathcal{M}$, opening space $\mathcal{T}$, and commitment space $\mathcal{C}$, consists of the following polynomial-time algorithms:*

- $\mathsf{pp}_c \leftarrow \mathsf{CSetup}(1^\kappa)$*: Setup is a probabilistic algorithm which takes as input the security parameter 1^κ and outputs the set of public parameters pp_c.*
- $\mathsf{cm} \leftarrow \mathsf{Com}(\mathsf{pp}_c, m, \tau)$*: The commitment algorithm takes as input pp_c and a message $m \in \mathcal{M}$ along with a trapdoor $\tau \in \mathcal{T}$, and outputs a commitment $\mathsf{cm} \in \mathcal{C}$.*
- $0/1 \leftarrow \mathsf{CVerify}(\mathsf{pp}_c, \mathsf{cm}, m', \tau')$*: Verification is a deterministic algorithm which takes as input pp_c, a commitment $\mathsf{cm} \in \mathcal{C}$, a message $m' \in \mathcal{M}$, and an opening value $\tau' \in \mathcal{T}$, and outputs either 0 (reject) or 1 (accept).*

Informally, the primary security requirements for a commitment scheme are *correctness*, *hiding*, and *binding*. Correctness ensures that correctly generated commitments pass the verification phase. The hiding property guarantees that the commitment does not reveal any information about the hidden value, while binding ensures that a committer cannot open a commitment to two distinct messages.

Pedersen Commitment Scheme [75]**.** Over a cyclic group $\mathbb{G}$ of prime order p with generator g, the Pedersen commitment scheme allows to commit to a scalar message $m \in \mathbb{Z}_p$ and is perfectly hiding and computationally binding. It consists of the following polynomial-time algorithms:

- $\mathsf{pp}_c \leftarrow \mathsf{CSetup}(1^\kappa)$: Sample $r \leftarrow_\$ \mathbb{Z}_p$ and set $G_1 \leftarrow g^r$. Output $\mathsf{pp}_c \leftarrow (G_0 = g, G_1, \mathcal{M})$, where $\mathcal{M} = \mathbb{Z}_p$.
- $\mathsf{cm} \leftarrow \mathsf{Com}(\mathsf{pp}_c, m, \tau)$: Compute $\mathsf{cm} \leftarrow G_0^\tau G_1^m$. Output cm.
- $0/1 \leftarrow \mathsf{CVerify}(\mathsf{pp}_c, \mathsf{cm}, m', \tau')$: Compute $\mathsf{cm}' \leftarrow G_0^{\tau'} G_1^{m'}$. Return 1 if $\mathsf{cm} = \mathsf{cm}'$ and 0 otherwise.

The Pedersen commitment scheme can be extended to allow commitment to more than one message. More precisely, the message space can be $\mathcal{M} = \mathbb{Z}_p^\ell$, where ℓ is an upper bound for the number of committed scalar messages. The extended Pedersen commitment scheme is as follows:

- $\mathsf{pp}_c \leftarrow \mathsf{CSetup}(1^\kappa)$: Sample $\alpha_1, \ldots, \alpha_\ell \leftarrow_\$ \mathbb{Z}_p$ and set $G_j \leftarrow g^{\alpha_j}$ for all $j \in [1, \ell]$. Output $(G_0 = g, G_1, \ldots, G_\ell, \mathcal{M})$, where $\mathcal{M} = \mathbb{Z}_p^\ell$.
- $\mathsf{cm} \leftarrow \mathsf{Com}(\mathsf{pp}_c, \vec{m}, \tau)$: For $\vec{m} = (m_1, \ldots, m_\ell)$, compute $\mathsf{cm} \leftarrow G_0^\tau \prod_{j=1}^{\ell} G_i^{m_i}$. Output cm.
- $0/1 \leftarrow \mathsf{CVerify}(\mathsf{pp}_c, \mathsf{cm}, \vec{m}', \tau')$: Compute $\mathsf{cm}' \leftarrow G_0^{\tau'} \prod_{j=1}^{\ell} G_i^{m_i'}$. Return 1 if $\mathsf{cm} = \mathsf{cm}'$ and 0 otherwise.

A.3 Non-interactive Zero-Knowledge Proofs

Zero-knowledge proofs enable a prover to convince a skeptical verifier of the validity of a statement without revealing any other information. *Non-interactive* zero-knowledge proofs (NIZKs) [15,56] only require one round of communication.

Definition 18 (Non-interactive Zero-Knowledge Proof [56]). *Consider an NP-relation $\mathcal{R}$ defined over a language $L = \{x \mid \exists\ w \text{ s.t. } (x, w) \in \mathcal{R}\}$, where x and w denote statement and witness, respectively. A NIZK over the relation $\mathcal{R}_L$ consists of the following PPT algorithms:*

- $\mathsf{CRS} \leftarrow \mathsf{Setup}(1^\kappa)$*: The setup algorithm takes as input the security parameter 1^κ and outputs a common reference string* CRS*.*
- $\pi \leftarrow \mathsf{Prove}(\mathsf{CRS}, x, w)$*: The prove algorithm takes as input a* CRS*, a statement x, and a witness w, and outputs a proof π.*
- $0/1 \leftarrow \mathsf{Verify}(\mathsf{CRS}, x, \pi)$*: The verification algorithm is a deterministic algorithm that takes as input a* CRS*, a statement x, and a proof π, and outputs either* 0 *(reject) or* 1 *(accept).*

A NIZK proof system is said to be *complete* if all pairs of statements and witnesses, $(x, w) \in \mathcal{R}_L$, pass verification. The *zero-knowledge* property guarantees that the proof does not reveal any information about the witness w. The *knowledge soundness* property guarantees that a malicious prover cannot convince the verifier of a false statement unless he knows the witness.

A.4 Threshold Blind Signatures

Here we recall the definition of threshold blind signatures as stated in [82]. Let pp be a given set of public parameters.

- $(\vec{\mathsf{vk}}, \vec{\mathsf{sk}}, \mathsf{vk}) \leftarrow \mathsf{TTPKeyGen}(\mathsf{pp}, \ell, n, t)$: The probabilistic key generation algorithm takes the public parameters pp and length ℓ along with two integers $t, n \in \mathsf{poly}(1^\kappa)$ such that $1 \leq t \leq n$ as inputs. It returns two vectors of size n of signing/verification keys $\vec{\mathsf{sk}} = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$ and $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$ such that each party P_i for $i \in [n]$ receives a pair $(\mathsf{sk}_i, \mathsf{vk}_i)$ along with the global verification key vk.
- $(\Omega, id, \vec{cm}, \pi_s) \leftarrow \mathsf{PrepareBlindSign}(\mathsf{pp}, \vec{m})$: This algorithm is run by the user to blind the message $\vec{m}$ under some random blinding factors Ω.
- $(\perp, \bar{\sigma}_i) \leftarrow \mathsf{BlindSign}(\mathsf{pp}, \mathsf{sk}_i, id, \vec{cm}, \pi_s)$: The blind signing algorithm is run by each signer with secret signing key sk_i to blindly sign the messages. It either returns a blind partial signature $\bar{\sigma}_i$ as output or responds with $\perp$.
- $(\perp, \bar{\sigma}) \leftarrow \mathsf{AggCred}(\mathsf{pp}, \{i, \bar{\sigma}_i\}_{i \in \mathcal{T}})$: The reconstruction algorithm is run by the user to aggregate the received partial signatures. If a sufficient number of well-formed partial signatures are available, it returns an aggregated blind signature; otherwise it returns $\perp$.
- $\sigma \leftarrow \mathsf{Unblind}(\mathsf{pp}, \bar{\sigma}, \Omega)$: The user who knows the blinding factors Ω runs this algorithm to unblind the aggregated signature $\bar{\sigma}$. It returns σ as output.

Informally, a threshold blind signature scheme satisfies two main security properties: one-more unforgeability and blindness. One-more unforgeability requires an adversary to produce $k+1$ valid signatures (representing a group of signers) having only queried its signing oracle k times, guaranteeing that at least one signature is a forgery. The blindness property guarantees that an adversarial signer cannot learn meaningful information about the message $\vec{m}$.

References

1. Abe, M., Ambrona, M., Ohkubo, M., Tibouchi, M.: Lower bounds on structure-preserving signatures for bilateral messages. In: Catalano, D., De Prisco, R. (eds.) SCN 2018. LNCS, vol. 11035, pp. 3–22. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98113-0_1
2. Abe, M., Chase, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Constant-size structure-preserving signatures: generic constructions and simple assumptions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 4–24. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_3
3. Abe, M., Chow, S.S.M., Haralambiev, K., Ohkubo, M.: Double-trapdoor anonymous tags for traceable signatures. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 183–200. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21554-4_11
4. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_12
5. Abe, M., Groth, J., Haralambiev, K., Ohkubo, M.: Optimal structure-preserving signatures in asymmetric bilinear groups. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 649–666. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_37
6. Abe, M., Groth, J., Ohkubo, M.: Separating short structure-preserving signatures from non-interactive assumptions. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 628–646. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_34
7. Abe, M., Groth, J., Ohkubo, M., Tibouchi, M.: Unified, minimal and selectively randomizable structure-preserving signatures. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 688–712. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_29
8. Abe, M., Jutla, C.S., Ohkubo, M., Pan, J., Roy, A., Wang, Y.: Shorter QA-NIZK and SPS with tighter security. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11923, pp. 669–699. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34618-8_23
9. Attrapadung, N., Libert, B., Peters, T.: Computing on authenticated data: new privacy definitions and constructions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 367–385. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_23
10. Bacho, R., Loss, J.: On the adaptive security of the threshold bls signature scheme. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, pp. 193–207. Association for Computing Machinery, New York (2022). https://doi.org/10.1145/3548606.3560656

11. Barthe, G., Fagerholm, E., Fiore, D., Scedrov, A., Schmidt, B., Tibouchi, M.: Strongly-optimal structure preserving signatures from type II pairings: synthesis and lower bounds. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 355–376. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_16
12. Bauer, B., Fuchsbauer, G., Loss, J.: A classification of computational assumptions in the algebraic group model. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 121–151. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_5
13. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_7
14. Bellare, M., Crites, E.C., Komlo, C., Maller, M., Tessaro, S., Zhu, C.: Better than advertised security for non-interactive threshold signatures. In: Advances in Cryptology - CRYPTO 2022. LNCS, vol. 13510, pp. 517–550. Springer (2022). https://doi.org/10.1007/978-3-031-15985-5_18
15. Ben-Or, M., Goldreich, O., Goldwasser, S., Håstad, J., Kilian, J., Micali, S., Rogaway, P.: Everything provable is provable in zero-knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 37–56. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_4
16. Blakley, G.R.: Safeguarding cryptographic keys. In: Proceedings of AFIPS 1979 National Computer Conference 48, pp. 313–317 (1979)
17. Blazy, O., Canard, S., Fuchsbauer, G., Gouget, A., Sibert, H., Traoré, J.: Achieving optimal anonymity in transferable e-cash with a judge. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 11. LNCS, vol. 6737, pp. 206–223. Springer, Heidelberg (2011)
18. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36288-6_3
19. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. J. Cryptol. **17**(4), 297–319 (2004). https://doi.org/10.1007/s00145-004-0314-9
20. Brandão, L.T., Davidson, M., Vassilev, A., et al.: Nist roadmap toward criteria for threshold schemes for cryptographic primitives. In: National Institute of Standards and Technology Internal or Interagency Report 8214A (2020). https://doi.org/10.6028/NIST.IR.8214A
21. Brandão, L., Peralta, R.: NIST first call for multi-party threshold schemes. https://nvlpubs.nist.gov/nistpubs/ir/2023/NIST.IR.8214C.ipd.pdf (2023)
22. Camenisch, J., Drijvers, M., Lehmann, A., Neven, G., Towa, P.: Short threshold dynamic group signatures. In: Galdi, C., Kolesnikov, V. (eds.) SCN 2020. LNCS, vol. 12238, pp. 401–423. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57990-6_20
23. Camenisch, J., Dubovitskaya, M., Haralambiev, K., Kohlweiss, M.: Composable and modular anonymous credentials: definitions and practical constructions. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015, Part II. LNCS, vol. 9453, pp. 262–288. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48800-3_11
24. Canetti, R.: Universally composable security: a new paradigm for cryptographic protocols. In: 42nd FOCS, pp. 136–145. IEEE Computer Society Press, October 2001. https://doi.org/10.1109/SFCS.2001.959888

25. Canetti, R., Gennaro, R., Goldfeder, S., Makriyannis, N., Peled, U.: UC non-interactive, proactive, threshold ECDSA with identifiable aborts. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020, pp. 1769–1787. ACM Press, November 2020. https://doi.org/10.1145/3372297.3423367
26. Canetti, R., Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Adaptive security for threshold cryptosystems. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 98–116. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_7
27. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable proof systems and applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 281–300. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_18
28. Couteau, G., Hartmann, D.: Shorter non-interactive zero-knowledge arguments and ZAPs for algebraic languages. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12172, pp. 768–798. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_27
29. Cramer, R., Damgård, I., Nielsen, J.B.: Multiparty computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–300. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44987-6_18
30. Cramer, R., Franklin, M., Schoenmakers, B., Yung, M.: Multi-authority secret-ballot elections with linear work. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 72–83. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_7
31. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 103–118. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_9
32. Crites, E., Kohlweiss, M., Preneel, B., Sedaghat, M., Slamanig, D.: Threshold Structure-Preserving Signatures. Cryptology ePrint Archive, Report 2022/839 (2022). https://eprint.iacr.org/2022/839
33. Crites, E.C., Komlo, C., Maller, M.: Fully adaptive schnorr threshold signatures. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Santa Barbara, CA, USA, August 20–24, 2023, Proceedings, Part I. LNCS, vol. 14081, pp. 678–709. Springer (2023). https://doi.org/10.1007/978-3-031-38557-5_22
34. Crites, E.C., Lysyanskaya, A.: Delegatable anonymous credentials from mercurial signatures. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 535–555. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12612-4_27
35. Damgård, I., Koprowski, M.: Practical threshold RSA signatures without a trusted dealer. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 152–165. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44987-6_10
36. Damgård, I., Nielsen, J.B.: Universally composable efficient multiparty computation from threshold homomorphic encryption. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 247–264. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_15
37. De Santis, A., Desmedt, Y., Frankel, Y., Yung, M.: How to share a function securely. In: 26th ACM STOC, pp. 522–533. ACM Press, May 1994. https://doi.org/10.1145/195058.195405
38. Desmedt, Y.G.: Making conditionally secure cryptosystems unconditionally abuse-free in a general context. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 6–16. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_2

39. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_28
40. Doerner, J., Kondi, Y., Lee, E., shelat, A.: Threshold ECDSA from ECDSA assumptions: the multiparty case. In: 2019 IEEE Symposium on Security and Privacy, pp. 1051–1066. IEEE Computer Society Press, May 2019. https://doi.org/10.1109/SP.2019.00024
41. Drijvers, M., et al.: On the security of two-round multi-signatures. In: 2019 IEEE Symposium on Security and Privacy, pp. 1084–1101. IEEE Computer Society Press, May 2019. https://doi.org/10.1109/SP.2019.00050
42. El Kaafarani, A., Ghadafi, E., Khader, D.: Decentralized traceable attribute-based signatures. In: Benaloh, J. (ed.) CT-RSA 2014. LNCS, vol. 8366, pp. 327–348. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04852-9_17
43. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12
44. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 60–77. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_4
45. Fuchsbauer, G.: Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. Cryptology ePrint Archive, Report 2009/320 (2009). https://eprint.iacr.org/2009/320
46. Fuchsbauer, G.: Commuting signatures and verifiable encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 224–245. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_14
47. Fuchsbauer, G., Hanser, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 233–253. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_12
48. Fuchsbauer, G., Hanser, C., Slamanig, D.: Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. J. Cryptol. **32**(2), 498–546 (2019). https://doi.org/10.1007/s00145-018-9281-4
49. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 33–62. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_2
50. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for cryptographers. Discret. Appl. Math. **156**(16), 3113–3121 (2008). https://doi.org/10.1016/j.dam.2007.12.010
51. Gay, R., Hofheinz, D., Kohl, L., Pan, J.: More efficient (almost) tightly secure structure-preserving signatures. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 230–258. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_8
52. Ghadafi, E.: Short structure-preserving signatures. In: Sako, K. (ed.) CT-RSA 2016. LNCS, vol. 9610, pp. 305–321. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29485-8_18
53. Ghadafi, E.: How low can you go? Short structure-preserving signatures for Diffie-Hellman vectors. In: O'Neill, M. (ed.) 16th IMA International Conference on Cryptography and Coding. LNCS, vol. 10655, pp. 185–204. Springer, Heidelberg (2017)
54. Ghadafi, E.: More efficient structure-preserving signatures - or: bypassing the type-III lower bounds. In: Foley, S.N., Gollmann, D., Snekkenes, E. (eds.) ESORICS

2017. LNCS, vol. 10493, pp. 43–61. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66399-9_3
55. Ghadafi, E.: Further lower bounds for structure-preserving signatures in asymmetric bilinear groups. In: Buchmann, J., Nitaj, A., Rachidi, T. (eds.) AFRICACRYPT 2019. LNCS, vol. 11627, pp. 409–428. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23696-0_21
56. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th ACM STOC, pp. 218–229. ACM Press, May 1987. https://doi.org/10.1145/28395.28420
57. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (2006). https://doi.org/10.1007/11935230_29
58. Groth, J.: Efficient fully structure-preserving signatures for large messages. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 239–259. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_11
59. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_21
60. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24
61. Gurkan, K., Jovanovic, P., Maller, M., Meiklejohn, S., Stern, G., Tomescu, A.: Aggregatable distributed key generation. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 147–176. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_6
62. Hofheinz, D., Jager, T.: Tightly secure signatures and public-key encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 590–607. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_35
63. Jutla, C.S., Roy, A.: Improved structure preserving signatures under standard bilinear assumptions. In: Fehr, S. (ed.) PKC 2017. LNCS, vol. 10175, pp. 183–209. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54388-7_7
64. Kiayias, A., Kohlweiss, M., Sarencheh, A.: Peredi: privacy-enhanced, regulated and distributed central bank digital currencies. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, pp. 1739–1752. Association for Computing Machinery, New York (2022). https://doi.org/10.1145/3548606.3560707
65. Kiltz, E., Pan, J., Wee, H.: Structure-preserving signatures from standard assumptions, revisited. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 275–295. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_14
66. Kim, H., Lee, Y., Abdalla, M., Park, J.H.: Practical dynamic group signature with efficient concurrent joins and batch verifications. Cryptology ePrint Archive, Report 2020/921 (2020). https://eprint.iacr.org/2020/921
67. Kim, H., Sanders, O., Abdalla, M., Park, J.H.: Practical Dynamic Group Signatures Without Knowledge Extractors. Designs, Codes and Cryptography, October 2022. https://doi.org/10.1007/s10623-022-01129-w
68. Komlo, C., Goldberg, I.: FROST: flexible round-optimized schnorr threshold signatures. In: Dunkelman, O., Jacobson, Jr., M.J., O'Flynn, C. (eds.) SAC 2020. LNCS, vol. 12804, pp. 34–65. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-81652-0_2

69. Kondi, Y., Magri, B., Orlandi, C., Shlomovits, O.: Refresh when you wake up: proactive threshold wallets with offline devices. In: 2021 IEEE Symposium on Security and Privacy (SP), pp. 608–625 (2021). https://doi.org/10.1109/SP40001.2021.00067
70. Libert, B., Joye, M., Yung, M.: Born and raised distributively: fully distributed non-interactive adaptively-secure threshold signatures with short shares. Theor. Comput. Sci. **645**, 1–24 (2016). https://doi.org/10.1016/j.tcs.2016.02.031
71. Libert, B., Peters, T., Joye, M., Yung, M.: Linearly homomorphic structure-preserving signatures and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 289–307. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_17
72. Libert, B., Peters, T., Yung, M.: Short group signatures via structure-preserving signatures: standard model security from simple assumptions. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 296–316. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_15
73. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: 40th FOCS, pp. 120–130. IEEE Computer Society Press, October 1999. https://doi.org/10.1109/SFFCS.1999.814584
74. Mizuide, T., Takayasu, A., Takagi, T.: Tight reductions for Diffie-Hellman variants in the algebraic group model. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 169–188. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12612-4_9
75. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-46766-1_9
76. Pointcheval, D., Sanders, O.: Short randomizable signatures. In: Sako, K. (ed.) CT-RSA 2016. LNCS, vol. 9610, pp. 111–126. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29485-8_7
77. Rial, A., Piotrowska, A.M.: Security analysis of coconut, an attribute-based credential scheme with threshold issuance. Cryptology ePrint Archive, Report 2022/011 (2022). https://eprint.iacr.org/2022/011
78. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_22
79. Shamir, A.: How to share a secret. Commun. Assoc. Comput. Mach. **22**(11), 612–613 (1979)
80. Shoup, V.: Practical threshold signatures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 207–220. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_15
81. Shoup, V., Gennaro, R.: Securing threshold cryptosystems against chosen ciphertext attack. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 1–16. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054113
82. Sonnino, A., Al-Bassam, M., Bano, S., Meiklejohn, S., Danezis, G.: Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. In: NDSS 2019. The Internet Society, February 2019
83. Tomescu, A., Bhat, A., Applebaum, B., Abraham, I., Gueta, G., Pinkas, B., Yanai, A.: UTT: Decentralized Ecash with Accountable Privacy. IACR Cryptol. ePrint Arch. p. 452 (2022), https://eprint.iacr.org/2022/452
84. Tymokhanov, D., Shlomovits, O.: Alpha-Rays: Key Extraction Attacks on Threshold ECDSA Implementations. Cryptology ePrint Archive, Report 2021/1621 (2021). https://ia.cr/2021/1621