Threshold Structure-Preserving Signatures with Randomizable Key

Ahmet Ramazan Ağırtaş[2], Emircan Çelik[1**], and Oğuz Yayla[1]

[1] AYWARE
a.r.agirtas@gmail.com
[2] Middle East Technical University, Ankara, Turkey
celik.emircan@metu.edu.tr,oguz@metu.edu.tr

Abstract. While digital signatures serve to confirm message integrity and the identity of the signer, the inherent link between the public key and the signer's identity can pose challenges in anonymized networks or applications focused on preserving privacy. Signatures with randomizable keys aim to disentangle the signer's identity from their public key, thus preserving the signature's validity. This approach ensures that the signature, even with a randomized key, maintains its verifiability without linking it to the signer's identity.
Although signatures with randomizable keys effectively maintain privacy, additional structural improvements are necessary in specialized signature schemes for complex cryptographic frameworks. Threshold structure-preserving signatures offer a way to construct modular protocols while retaining the benefits of structure-preserving properties. Thus, the randomizable key version of it is essential for a wide range of applications, making it the foundation of this work. In this study, signatures with randomizable key principles combined with threshold structure-preserving signatures to build a strong cryptographic base for privacy-preserving applications. This foundation makes sure that signatures are valid while also being modular and unlinkable.
An earlier version of this work appeared in the 22nd International Conference on Security and Cryptography(SECRYPT 2025) [6]; the present article extends that study by adding the formal security proofs of the introduced protocols.

Keywords: SPS · TSPS · Randomizable Keys

1 Introduction

Digital signatures are cryptographic techniques that provide a method to authenticate the integrity of the message and the identity of the signer. The message, once signed by the signer, can be verified using the signer's public key. However, in some certain scenarios, such as anonymized networks or privacy-preserving applications, the inherent connection between the public key and the

** corresponding author

signer's identity poses a problem [8]. Revealing this information conflicts with preserving anonymity. In such situations, balancing authenticity with anonymity becomes a challenging task. To tackle this issue, Signatures with Randomizable Keys (SwRK) have drawn most of the attention to dissociate the signer's identity from their public key. In this scheme, the signer signs a message under the randomized secret signing key while ensuring that the signature remains valid. Although different terminology is employed throughout the literature, a comprehensive overview of previous work on this topic, connecting key-homomorphic signatures, signatures with re-randomizable keys, signatures with key blinding, and key-updatable signatures, can be found in [15].

While SwRK is useful for privacy-preserving protocols, it is also important for the protocol to be structured in a modular design to allow for the simple implementation of complex primitives. Fortunately, the concept of Structure Preserving Signatures (SPS) offers a way to construct modular protocols. SPS are signatures based on pairings, wherein verification keys, signatures, and messages are elements in a bilinear group, and the verification equation is derived from a pairing-product equation [1]. They have numerous applications in public-key cryptography, including group signatures [33] to ensure individual signers to remain anonymous, blind signatures [29] to sign a message while keeping its content hidden from the signer, homomorphic signatures [38] to enable computations on signed data, delegatable anonymous credential application [10] to enable users to delegate their credentials without compromising security, and e-cash application [13] to allow untraceable yet verifiable transactions. The randomizable key version of them is also required for a wide range of applications. This necessity constitutes the foundational basis for our work.

1.1 Related Work

Signatures with Randomizable Keys (SwRK). Several signatures with randomizable keys have been suggested in the literature. Its related concepts encompass key homomorphic schemes, signature schemes featuring re-randomizable keys, signature schemes incorporating key blinding, and key-updatable signatures. Thus, we will explore some of the prominent protocols in each concepts.

Key-homomorphic Signatures. The first study to explicitly examine certain homomorphic properties on the key space of the signature was conducted by Derler and Slamanig [23]. They introduce a definitional framework for key-homomorphisms in signature schemes, and evaluate the existing schemes presenting key-homomorphic properties.

Signatures with Re-randomizable Keys. Signatures with re-randomizable keys is introduced by Fleischhacker et al. [27]. The randomization of the key pair guarantees that distinct signatures cannot be linked to the same signer, while yet allowing for the verification of ownership using zero-knowledge proofs. In 2019, Das et al. [21] instantiated stateful deterministic wallet from ECDSA-based and BLS [14] signature schemes featuring perfectly re-randomizable keys. In 2020, Alkadri et al. [4] designed the first post-quantum secure signature scheme featuring re-randomizable public keys by providing a construction derived from

generic lattice-based Fiat-Shamir signature schemes. In 2023, Shaw and Dutta [45] developed such signature schemes based on isogenies employing signature schemes CSI-FiSh [11] and CSI-SharK [5]. Based on the defined notion of [27], Gouvea and Komlo built Re-randomized FROST in 2024, which facilitates re-randomizable public and secret keys [32]. They provide the definition of a re-randomizable threshold signature scheme, followed by additional algorithms for FROST to produce a randomizer and to randomize the group public key and signer's secret key share.

Signatures with Key Blinding. Key blinding standardization has been established in an IETF technical specification [22]. The formal security analysis of this construction is also provided in [24]. In 2021, Eaton et al. [25] enhanced lattice-based post-quantum signature schemes by incorporating key-blinding functionality, focusing on the Dilithium-QROM variant [36], CSI-FiSh [11], LegRoast [12], and Picnic [16], and proved the security in classical random oracle model (ROM). In 2024, Balumuri et al. [8] presented quantum-safe public key blinding signature scheme derived from MultiParty Computation-in-the-Head Signature Schemes, and provided its security proof within the quantum random oracle model (QROM).

Key-updatable Signatures. Key-updatable structure in digital signatures is given in [35] to update keys and acquire signatures using the updated keys. In 2021, Cini et al. [17] presented a compact overview of updatable signatures derived from the extension of key-homomorphic schemes considering epoch time.

Structure Preserving Signatures (SPS). Although SwRK schemes play an important role in privacy-preserving protocols, it is also important that cryptographic protocols allow for the modular design to simple implementation of complex primitives. SPS schemes are compatible with the construction of a modular approach in protocols [1]. Thus, we will explore some of the SPS schemes within the framework of randomization.

In 2014, Abe et al. [2] proposed a re-randomizable SPS scheme that enables the randomization of both the signature and key after generation, while still preserving the structure of the signature. This scheme achieves direct randomization of the signature without the necessity of a message. In 2016, Ghadafi [31] proposed shorter SPS than existing SPS schemes. Its key-randomizable variant with an adaptable algorithm is given in [23].

In 2019, Crites and Lysyanskaya [20] built mercurial signatures that are malleable with respect to the message, signature, and key space, focusing on Structure-Preserving Signatures on Equivalence Classes (SPS-EQ) [34]. Later, Connolly et al. [18] proposed a new variant of it by building on the common reference string (CRS) model assumption. Inspired by SPS-EQ, Backes et al. [7] presented signatures featuring a flexible public key primitive that converts the key into an alternative representative of the same equivalence class. Recently, Mir et. al. [40] proposed the first aggregatable SPS-EQ, by introducing aggregate signatures with randomizable public key and tags, along with aggregate mercurial signatures. In 2024, Abe et al. [41] presented interactive threshold mercurial signatures that expand the scope of threshold SPS to encompass EQ.

Many SPS in the literature are inadequate for thresholding due to nonlinear processes or even require significant overhead. However, in 2023, Crites et al. [19] proposed a Threshold Structure-Preserving Signature (TSPS) by defining indexed Diffie-Hellman message space. They address thresholdizing by employing indexing that converts each scalar message m into an index id to generate partial signatures in a compatible format for aggregation in a threshold scheme.

Our Contribution The goal of this work is to define the randomizable key version of Indexed Message Structure-Preserving Signatures and Threshold Structure-Preserving Signatures. In our construction, the signer randomizes its key pair and utilizes it to sign a message, guaranteeing enhanced privacy and unlinkability and unextractability.

Paper Organization This paper is organized as follows: Section 2 provides the necessary building blocks for our construction. Section 3 introduces our proposed Threshold Structure-Preserving Signature with Randomizable Keys (TSPSwRK) and its non-threshold counterpart IM-SPSwRK, followed by a performance analysis in Section 4. Finally, Section 5 offers the conclusion of the paper.

2 Preliminaries

This section outlines the fundamental building blocks for the TSPS scheme, subsequently addressing hardness assumptions and definitions of the related signature schemes.

For the syntax, λ is the security parameter 1^λ is its unary representation, and $negl(\lambda)$ denotes negligible function. While $x \xleftarrow{\$} X$ is utilized to indicate that x is uniformly sampled from the set X, the cardinality of a set X is represented by the symbol $|X|$. Also, $\mathsf{A}(\mathsf{x}) \rightarrow \mathsf{y}$ denotes that y is an output of the algorithm A on input x, whereas $\mathsf{x} \leftarrow \mathsf{y}$ denotes the straightforward assignment process.

Definition 1 (Bilinear Map). *A bilinear map is a function $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_t$ in groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$ with prime order q, that satisfies:*

- *Bilinearity: $e(g_1^m, g_2^n) = e(g_1, g_2)^{m.n}$ for all $m, n \in \mathbb{Z}_q$, $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$.*
- *Non-degeneracy: $e(g_1, g_2) \neq 1$ for all $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$.*

The pairing map is termed symmetric (Type I) if $\mathbb{G}_1 = \mathbb{G}_2$. Asymmetric pairing occurs when $\mathbb{G}_1 \neq \mathbb{G}_2$. In this scenario, if an effectively computable isomorphism $\phi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$ exists, it is categorized as Type II; otherwise, it is classified as Type III, which will be also relied on in this study.

Definition 2 (Diffie-Hellman Message Space $\mathcal{M}_{DH}$ [28]). *Given a Type III bilinear map $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g_1, g_2)$. A pair $(M_1, M_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ belongs to the Diffie-Hellman message space $\mathcal{M}_{DH}$ if there exist $m \in Z_p$ such that $M_1 = g_1^m$ and $M_2 = g_2^m$. Verification of $e(M_1, g_2) = e(g_1, M_2)$ determines such a pair.*

Definition 3 (Indexed Diffie-Hellman Message Space $\mathcal{M}_{iDH}^{H}$ [19]). *Given a Type III bilinear map $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g_1, g_2)$, an index set $\mathcal{I}$ ($id \in \mathcal{I}$), and a random oracle $\mathsf{H} : \mathcal{I} \to \mathbb{G}_1$. $\mathcal{M}_{iDH}^{H}$ is an Indexed Diffie-Hellman message space if the followings hold:*

(1) $\mathcal{M}_{iDH}^{H} \subset \{(id, \tilde{M}) | id \in \mathcal{I}, m \in \mathbb{Z}_p, \tilde{M} = (H(id)^m, g_2^m) \in \mathbb{G}_1 \times \mathbb{G}_2\}$
(2) For all $(id, \tilde{M}) \in \mathcal{M}_{iDH}^{H}$, $(id', \tilde{M}') \in \mathcal{M}_{iDH}^{H}$, $id = id'$, *then* $\tilde{M} = \tilde{M}'$.

Definition 4 (Shamir's Secret Sharing [44]). *Shamir's Secret Sharing is a (t, n)-threshold scheme that enables the division of a secret s into n shares such that any subset of shares containing at least t shares can reconstruct the secret s, while any subset containing fewer than t shares cannot obtain any knowledge regarding s. Consider a finite field of prime order p, denoted as $\mathbb{F}_p$. Shamir's secret sharing comprises share generation and secret reconstruction as outlined below:*

- *$ShareGen(s, p, n, t)$: The dealer selects a random polynomial $f(x) \in \mathbb{F}_p[x]$ of degree $(t-1) < p$, where coefficients selected from $\mathbb{F}_p$, such that $f(0) = s \in \mathbb{F}_p$. The dealer secretly transmits a secret share $s_i = f(id_i) \mod p$ to P_i, where $id_i \in \mathbb{F}_p$ denotes the identifier of P_i, and the vector of shares is $\vec{s} = (s_1, \ldots, s_n)$.*
- *$Reconst(\{s_i\}_{i \in \mathcal{T}})$: The secret is reconstructed by Lagrange interpolation on the polynomial as $s = f(0) = \sum_{i \in \mathcal{T}} s_i \cdot \lambda_i$, where $\mathcal{T} \subset [1, n]$ of size at least t, and the Lagrange coefficient $\lambda_i = \prod_{j \in \mathcal{T}, j \neq i} \frac{id_j}{id_j - id_i} \in \mathbb{F}_p$*

Next will remind Schwarz-Zippel lemma.

Lemma 1 (Schwarz-Zippel Lemma). *[37] Let $P \in \mathbb{F}[X_1, X_2, \ldots, X_l]$ be a non-zero polynomial of total degree $d \geq 0$. If the values $r_1, r_2, \ldots, r_l$ are independently chosen at random from a finite subset $S \subset \mathbb{F}$, then*

$$Pr[(r_1, r_2, \ldots, r_l)] \leq \frac{d}{|S|}$$

The security notion of the proposed signature scheme is based on the following hardness assumptions.

Definition 5 (Generalized PS assumption (GPS_3)[19]). *Let the advantage of an adversary $\mathcal{A}$ against the Game G^{GPS_3}, as seen in Figure 1, be as follows:*

$$Adv_{\mathcal{A}}^{GPS_3}(\lambda) = Pr[\mathsf{G}_{\mathcal{A}}^{GPS_3} = 1]$$

The GPS_3 assumption is satisfied if for all $\mathcal{A}$, $Adv_{\mathcal{A}}^{GPS_3}(\lambda) < negl(\lambda)$.

In this game, the oracle $\mathcal{O}_0^{GPS_3}$ keeps the random base element $h = g_1^r$, where $r \xleftarrow{\$} \mathbb{Z}_p^*$. The oracle $\mathcal{O}_1^{GPS_3}$ takes h and message $(M_1, M_2) = (h^m, g_2^m) \in (\mathbb{G}_1 \times \mathbb{G}_2)$ then generates PS value $s = h^{x+my}$ [43]. Also, $\mathcal{Q}_0, \mathcal{Q}_1$ represents the sets of $\mathcal{O}_0^{GPS_3}$, $\mathcal{O}_1^{GPS_3}$ queries, respectively.

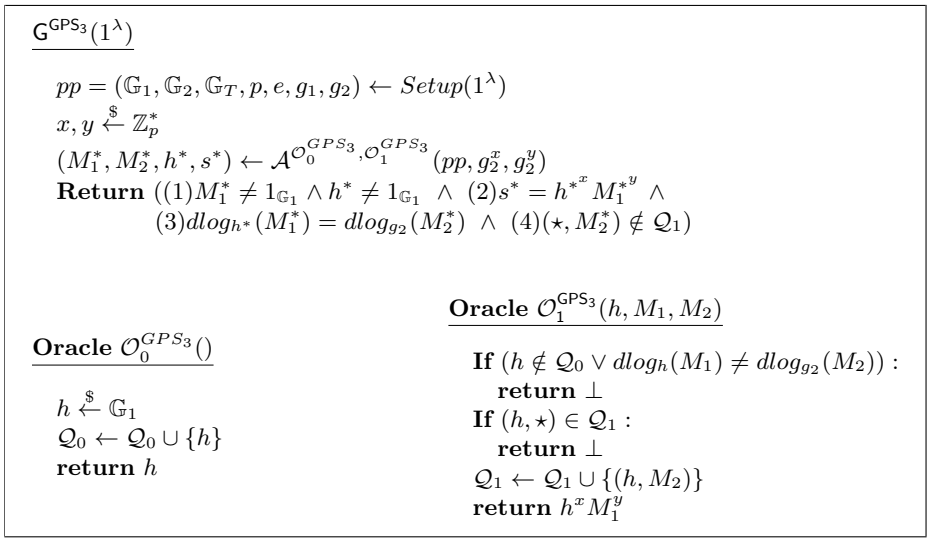

Fig. 1. Game G^{GPS_3}

Definition 6 (Unforgeability Under Randomization Game). *Let* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g}) \leftarrow \mathcal{BG}(1^\kappa)$ *be a description of a type-3 bilinear group. Choose* $x', y' \xleftarrow{\$} \mathbb{Z}_p$ *uniformly at random and pick a randomiser* $\rho = (\rho_1, \rho_2)$*. Define*

$$x = x' + \rho_1, \qquad y = y' + \rho_2, \qquad X = g^x,\ Y = g^y,\ \tilde{X} = \hat{g}^{\,x},\ \tilde{Y} = \hat{g}^{\,y}.$$

Oracle $\mathcal{O}$***.*** *On input* $m \in \mathbb{Z}_p$ *the oracle computes*

$$h \leftarrow H(\mathsf{id}), \qquad M_1 \leftarrow h^{\,m}, \qquad M_2 \leftarrow \hat{g}^{\,m},$$

where $H : \{0,1\}^* \to \mathbb{G}_1$ *is a hash-to-curve function modelled as a random oracle, and returns*

$$P \ = \ \big(h,\ h^{\,x} \cdot M_1^{\,y}\big).$$

Assumption. *Given* $(g, \hat{g}, \tilde{X}, \tilde{Y})$ *together with unlimited queries to both* $\mathcal{O}$ *and* H*, no probabilistic polynomial-time adversary can output a tuple* (h, M_1) *with* $h \neq 1_{\mathbb{G}_1}$ *and* $M_1 \neq 1_{\mathbb{G}_1}$ *that satisfies the relation above for some exponent* $m^\star$ not *previously queried to* $\mathcal{O}$*.*

Next, we restate the formal definitions of related signature schemes in the following subsections.

2.1 Pedersen's Distributed Key Generation Protocol

In Pedersen's Distributed Key Generation (DKG) protocol, each node shares a randomly selected secret via Feldman's Verifiable Secret Sharing (VSS) [26]. Let

p and q represent large primes, where q divides $p-1$. The group $\mathbb{G}$ is the unique subgroup of $\mathbb{Z}_p^*$ with prime order q, and g is a generator of $\mathbb{G}$. The protocol consists of the following steps:

- Each node N_i samples random secret $x_{i0} \xleftarrow{\$} \mathbb{Z}_p^*$ for $i \in [1, n]$, and forms a polynomial with degree t by selecting coefficients $\{x_{i\ell}\}_{\ell=1}^{t} \xleftarrow{\$} \mathbb{Z}_p^*$:

$$A_i(X) = x_{i0} + x_{i1}X + \ldots + x_{it}X^t \in \mathbb{Z}_p[x]$$

  N_i commits the coefficients by broadcasting $C_{i\ell} = g^{x_{i\ell}}$ mod q, $\forall \ell \in [0, t]$, then sends $A_i(j)$ mod p to N_j, s.t. $j \in [1, n] \setminus \{i\}$.
- Each node N_i verifies the consistency of the received shares $A_j(i)$ from N_j by calculating $g^{A_j(i)} = \prod_{\ell=0}^{t} C_{j\ell}^{i^\ell}$ mod q. A failed check for an index j is broadcasted as a complaint against N_j.
- A node is disqualified if it receives at least $t+1$ complaints; otherwise, it belongs to the set Q of non-disqualified nodes.
- The public verification values are calculated as $C_\ell = \prod_{i \in Q} C_{i\ell}$ mod q, for $\ell = 0, \ldots, t$. Each node N_i sets their secret shares as $x_i = \sum_{i \in Q} A_j(i)$ mod p. The secret shared value x, which is not computed by any party, is $s = \sum_{i \in Q} x_{i0}$ mod p. Also, public value y is $y = \prod_{i \in Q} C_{i0}$ mod q.

2.2 Signatures with Randomizable Keys

Signature with Randomizable Keys (SwRK) [15] consists of eight-tuple algorithms, namely `PPGen`, `KGen`, `SignGen`, `Verify`, `RandPK`, `RandSK`, `Adapt`, and `VerKey`, which can be defined as follows.

- $\texttt{PPGen}(1^\lambda) \to pp$: This probabilistic polynomial time (PPT) algorithm takes security parameter 1^λ as input, and outputs the public parameters pp.
- $\texttt{KGen}(pp) \to (sk, pk)$: This PPT algorithm takes pp as input, and outputs a secret key sk and public key pk.
- $\texttt{SignGen}(pp, sk, m) \to \sigma$: This PPT algorithm takes pp, sk, and message m as inputs, and outputs a signature σ on m.
- $\texttt{Verify}(pp, m, \sigma, pk) \to 0/1$: This deterministic algorithm takes pp, m, σ, pk as inputs, and outputs $0/1$.
- $\texttt{RandPK}(pp, pk, \rho) \to pk'$: This PPT algorithm takes pp, pk, and a key randomizer ρ as inputs, and outputs a fresh public key pk', such that $pk' = T(pk, \rho)$ for some transformation T.
- $\texttt{RandSK}(pp, sk, \rho) \to sk'$: This PPT algorithm takes pp, sk, and a key randomizer ρ as inputs, and outputs a fresh secret key sk', such that $sk' = \tilde{T}(sk, \rho)$ for some transformation $\tilde{T}$.
- $\texttt{Adapt}(pp, \sigma, \rho, pk, m) \to (pk', \sigma')$: This PPT algorithm takes pp, σ, ρ, pk, and m (optional) as inputs, and outputs adapted signature σ' and fresh public key pk'.

– $\texttt{VerKey}(pp, sk, pk) \to 0/1$: This deterministic algorithm takes public parameter pp and key pair (sk, pk) as inputs, then outputs 1 if (sk, pk) constitutes a valid key pair; otherwise, it outputs 0.

$\mathrm{pp} \leftarrow \mathrm{PPGen}(1^\lambda)$; $b \leftarrow \{0,1\}$; $(\mathrm{sk}, \mathrm{pk}) \leftarrow \mathrm{KGen}(\mathrm{pp})$; $\rho_0, \rho_1 \xleftarrow{\$} \mathcal{KR}$; $\sigma \leftarrow \mathrm{Sign}(\mathrm{sk}, m)$
$\tau_0 \leftarrow \mathrm{Adapt}(m, \sigma, \rho_0, \mathrm{pk})$; $\tau_1 \leftarrow \big(\mathrm{RandPK}(\mathrm{pk}, \rho_1), \mathrm{Sign}(\mathrm{RandSK}(\mathrm{sk}, \rho_1), m)\big)$
$b' \leftarrow \mathcal{A}(\tau_b)$; return $(b = b')$

Fig. 2. $\mathrm{Game}^{\mathrm{Adapt}}_{\Gamma,A}(\lambda)$

Security. A SwRK scheme must satisfy *unforgeability*, *unlinkability*, and *unextractability* requirements, which are defined as follows:

– **Unforgeability:** This ensures that anyone cannot create a valid signature on behalf of the signer, even if they have access to several valid message/signature pairs under randomizable keys. A SwRK scheme satisfies unforgeability if for any PPT adversary, the success probability is negligible:

$$\Pr\begin{bmatrix}(sk, pk) \leftarrow \mathsf{KGen}(pp),\\ \sigma \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{SignGen}}}(pp, sk, m) \text{ for some } \quad m \in \mathcal{Q}_{\mathsf{valid}}\\ (\sigma', pk') \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Adapt}}}(pp, \sigma, \rho, pk, m) \text{ for some } \ m \in \mathcal{Q}_{\mathsf{valid}}\\ \text{Output a new pair } (m^*, \sigma^*) :\\ \mathsf{Verify}(pp, m^*, \sigma^*, pk) = 1, \quad m^* \notin \mathcal{Q}_{\mathsf{valid}} \vee\\ \mathsf{Verify}(pp, m^*, \sigma^*, \mathtt{RandPK}(pk, \rho)) = 1, \quad m^* \notin \mathcal{Q}_{\mathsf{valid}}\end{bmatrix} \leq negl(\lambda),$$

where $\mathcal{A}^{\mathcal{O}_{\mathsf{SignGen}}}$ is an oracle that executes $\mathsf{SignGen}$ algorithm in SwRK. Also, a query set $\mathcal{Q}_{\mathsf{valid}}$ represents valid private messages m.

– **Unlinkability:** This indicates that, despite having access to the long-term public key and its randomization without randomizer, it remains impossible for anyone to distinguish between them. A SwRK scheme satisfies the unlinkability if the success probability approaches $\frac{1}{2}$ negligibly:

$$\Pr\begin{bmatrix}\text{For } \ b \xleftarrow{\$} \{0,1\},\\ pk_0 \leftarrow \mathsf{KeyGen}(pp),\\ pk_1 \leftarrow \mathsf{RandPK}(pp, pk, \rho) :\\ b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{guess}}}(pk_b) \wedge b' = b\end{bmatrix} - \frac{1}{2} \leq negl(\lambda)$$

– $((\alpha, \beta)$-**Unextractability:**[15, Defition 17] This ensures that despite access to randomizations of the long-term public key and its corresponding randomizer, anyone cannot extract the long-term public key. A SwRK scheme

satisfies the (α, β)-unextractability if for all PPT adversaries, the success probability is negligible:

$$\Pr^{(\alpha,\beta)-\mathrm{UNE}}_{\mathrm{SWRK},\mathcal{A}}(\lambda) := 2 \cdot \Pr\Big[\mathrm{Game}^{(\alpha,\beta)-\mathrm{UNE}}_{\mathrm{SWRK},\,\mathcal{A}}(\lambda) \;\Rightarrow\; \mathrm{true}\Big] - 1$$

where $\mathrm{Game}^{(\alpha,\beta)-\mathrm{UNE}}_{\mathrm{SWRK},\lambda}(\lambda)$ is shown in Fig 3.

$\mathrm{st} \leftarrow \emptyset; \gamma \leftarrow \perp; b \leftarrow \{0,1\}$; if $\alpha = 0$ then $\mathrm{pp} \leftarrow \mathsf{Setup}\left(1^\lambda\right)$ else $(pp, st) \leftarrow \mathcal{A}_0\left(\mathrm{st}, 1^\lambda\right)$

if $\beta = 0$ then $\rho \leftarrow \mathcal{KC}$ else $(\rho, \mathrm{st}) \leftarrow \mathcal{A}_1(\mathrm{st}, \mathrm{pp})$; $\{\mathrm{sk}_i, \mathrm{pk}_i\}_{i\in\{0,1\}} \leftarrow \mathsf{KeyGen}(\mathrm{pp})$

$\mathrm{pk}'_b \leftarrow \mathsf{RandPK}(\mathrm{pk}_b, \rho); \mathrm{sk}'_b \leftarrow \mathsf{RandSK}(\mathrm{sk}_b, \rho); b' \leftarrow \mathcal{A}_2^{\mathsf{Sign}(\cdot,\cdot)}\left(\mathrm{pk}'_b, \rho, \mathrm{st}\right)$; return $b = b'$

Oracle $\mathrm{Sign}(m, \gamma)$:

if $\gamma = b$ return $\mathsf{SignGen}(\mathrm{sk}_b, m)$ else if $\gamma = b - 1$ return $\mathsf{SignGen}(\mathrm{sk}_{b-1}, m)$

else if $\gamma = \perp$ return $\mathsf{SignGen}(\mathsf{RandSK}(sk_b, \rho), m)$ else return $\perp$

Fig. 3. $\mathrm{Game}^{(\alpha,\beta)-\mathrm{UNE}}_{\mathrm{SWRK},\,\mathcal{A}}(\lambda)$

2.3 Indexed-Message Structure-Preserving Signature

Indexed-Message Structure-Preserving Signature (IM-SPS) over the indexed Diffie-Hellman message space [19] builds four-tuple algorithms `Setup`, `KGen`, `SignGen`, and `Verify`, which can be described as follows.

1. $\mathtt{Setup}(1^\lambda) \rightarrow pp$: This PPT algorithm takes the security parameter 1^λ as input, and it outputs public parameters $pp = ((\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g_1, g_2), \mathsf{H})$, where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are groups of order p utilized in bilinear map e, g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Also, $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{G}_1$ is a hash function that maps arbitrary length binary inputs to the group $\mathbb{G}_1$.
2. $\mathtt{KGen}(pp) \rightarrow (sk, vk)$:
   (a) Sample $sk = (sk_0, sk_1) = (x, y) \xleftarrow{\$} \mathbb{Z}_p^*$
   (b) Compute $vk = (vk_0, vk_1) = (g_2^x, g_2^y)$
3. $\mathtt{SignGen}(pp, sk, (id, M_1, M_2)) \rightarrow \sigma$:
   (a) $(id, M_1, M_2) = (id, \mathsf{H}(id)^m, g_2^m) \in \mathcal{M}^H_{iDH}$ as defined in Definition 3, where $m \in \mathbb{Z}_p$
   (b) Check $e(h, M_2) = e(M_1, g_2)$, where $h \leftarrow \mathsf{H}(id)$. If this is not the case, abort.
   (c) Compute the signature $\sigma = (h, s) = (h, h^x \cdot M_1^y)$.
4. $\mathtt{Verify}(pp, vk, (M_1, M_2), \sigma) \rightarrow 0/1$:
   (a) Accept if all the following conditions hold:

i. $h \neq 1_{\mathbb{G}_1}$,

ii. $M_1 \neq 1_{\mathbb{G}_1}$,

iii. $e(h, M_2) = e(M_1, g_2)$,

iv. $e(h, vk_0)e(M_1, vk_1) = e(s, g_2)$.

(b) Reject, otherwise.

Security. IM-SPS scheme must satisfy *Existential Unforgeability under Chosen Indexed Message Attack (EUF-CiMA)*, meaning that anyone cannot generate a valid signature on behalf of the signer even if they adaptively query the signing oracle regarding index/message pairs of their selection [19].

2.4 Threshold Structure-Preserving Signature

Threshold Structure-Preserving Signatures (TSPS) for indexed message spaces [19] builds six-tuple algorithms `Setup`, `KGen`, `PSignGen`, `PSignVerify`, `Reconst`, and `Verify`, which can be described as follows.

1. $\texttt{Setup}(1^\lambda) \rightarrow pp$: This PPT algorithm takes the security parameter 1^λ as input, and outputs public parameters $pp = ((\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g_1, g_2), \mathsf{H})$, where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are groups of prime order p, utilized in an efficient bilinear map e, g_1, g_2 are generators of $\mathbb{G}_1, \mathbb{G}_2$, respectively, and $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{G}_1$ is a hash function that maps arbitrary length binary inputs to the group $\mathbb{G}_1$.
2. $\texttt{KGen}(pp, n, t) \rightarrow (\vec{sk}, \vec{vk}, vk)$:
   (a) Run Pedersen's Distributed Key Generation (PedDKG) protocol [42] as follows:
   - Each node N_i samples random two integers $(x_{i0}, y_{i0}) \xleftarrow{\$} \mathbb{Z}_p^*$ for $i \in [1, n]$, and forms two polynomials with degree t by sampling $2t$ random integer coefficient $\{x_{i\ell}, y_{i\ell}\}_{\ell=1}^{t} \xleftarrow{\$} \mathbb{Z}_p^*$:

$$f_i(X) = x_{i0} + x_{i1}X + \ldots + x_{it}X^t \in \mathbb{Z}_p[x]$$
$$g_i(X) = y_{i0} + y_{i1}Y + \ldots + y_{it}Y^t \in \mathbb{Z}_p[x]$$

     Each N_i commits the coefficients by broadcasting $B_{i\ell} = g_2^{x_{i\ell}}$, $C_{i\ell} = g_2^{y_{i\ell}}$ $\forall \ell \in [0, t]$. Then, N_i sends $f_i(j), g_i(j)$ to N_j, such that $j \in [1, n] \setminus \{i\}$.
   - Node N_i confirms the consistency of the received shares $f_j(i), g_j(i)$ from N_j by calculating $g_2^{f_j(i)} = \prod_{\ell=0}^{t} B_{j\ell}^{i^\ell}$ and $g_2^{g_j(i)} = \prod_{\ell=0}^{t} C_{j\ell}^{i^\ell}$. If these conditions are met, N_i accepts the shares; otherwise, it rejects them and complaints against malicious node N_j.
   - A node is deemed disqualified if it receives at least t of complaints. At the conclusion of this phase, at least t nodes from the set of qualified nodes Q proceed to the following steps.

– The global verification key vk is determined as follows:

$$vk := (vk_{00}, vk_{01}) = (\prod_{i\in Q} B_{i0}, \prod_{i\in Q} C_{i0}) = (g_2^{\sum_{i\in Q} x_{i0}}, g_2^{\sum_{i\in Q} y_{i0}}) = (g_2^x, g_2^y)$$

Also, its corresponding secret key will be $sk := (\sum_{i\in Q} x_{i0}, \sum_{i\in Q} y_{i0})$

– Each qualified node N_i defines its secret key sk_i and its corresponding verification key vk_i as follows:

$$sk_i := (sk_{i0}, sk_{i1}) = (\sum_{j\in Q} f_j(i), \sum_{j\in Q} g_j(i)) = (x_i, y_i)$$

$$vk_i := (vk_{i0}, vk_{i1}) = (\prod_{j\in Q}\prod_{\ell=0}^{t} (B_{j\ell})^{i^\ell}, \prod_{j\in Q}\prod_{\ell=0}^{t} (C_{j\ell})^{i^\ell}) = (g_2^{f_j(i)}, g_2^{g_j(i)}),$$

where $f(X) = \sum_{j\in Q} f_j(X)$ and $g(X) = \sum_{j\in Q} g_j(X)$.

(b) Set two vectors of size n of signing keys $\vec{sk} \leftarrow (sk_1, sk_2, \ldots, sk_n)$, and verification keys $\vec{vk} \leftarrow (vk_1, vk_2, \ldots, vk_n)$

3. $\texttt{PSignGen}(pp, sk_i, (id, M_1, M_2)) \rightarrow (\sigma_i, \perp)$:
   (a) $(id, M_1, M_2) = (id, \mathsf{H}(id)^m, g_2^m) \in \mathcal{M}_{iDH}^H$, where $m \in \mathbb{Z}_p$
   (b) Check $e(h, M_2) = e(M_1, g_2)$, where $h \leftarrow \mathsf{H}(id)$. If it does not hold, abort.
   (c) Compute the partial signatures $\sigma_i = (h_i, s_i) = (h, h^{sk_{i0}} \cdot M_1^{sk_{i1}})$.
4. $\texttt{PSignVerify}(pp, vk_i, (M_1, M_2), \sigma_i = (h_i, s_i)) \rightarrow (0/1)$:
   (a) Accept if all the following conditions hold:
      i. $h_i \neq 1_{\mathbb{G}_1}$,
      ii. $M_1 \neq 1_{\mathbb{G}_1}$,
      iii. $e(h_i, M_2) = e(M_1, g_2)$,
      iv. $e(h_i, vk_{i0})e(M_1, vk_{i1}) = e(s_i, g_2)$.
   (b) Reject, otherwise.
5. $\texttt{Reconst}(pp, \vec{vk}, (M_1, M_2), \{i, \sigma_i\}_{i\in\mathcal{T}}) \rightarrow (\sigma, \perp)$:
   (a) Abort if at least one of the following conditions holds:
      i. $\exists i \in \mathcal{T} \mid \texttt{PSignVerify}(pp, vk_i, (M_1, M_2), \sigma_i) = 0$,
      ii. $\exists i, j \in \mathcal{T}, i \neq j \mid h_i \neq h_j$.
   (b) Otherwise, set $h \leftarrow h_i$ and compute the signature $\sigma = (h, s) = (h, \prod_{i\in\mathcal{T}} s_i^{\lambda_i})$, where λ_i is an appropriate Lagrange coefficient.
6. $\texttt{Verify}(pp, vk, (M_1, M_2), \sigma) \rightarrow (0/1)$:
   (a) Accept, if all of the following conditions hold:
      i. $h \neq 1_{\mathbb{G}_1}$
      ii. $M_1 \neq 1_{\mathbb{G}_1}$

iii. $e(h, M_2) = e(M_1, g_2)$

iv. $e(h, vk_{00})e(M_1, vk_{01}) = e(s, g_2)$,

(b) Reject, otherwise.

The three primary security properties for TSPS defined over indexed message spaces are *partial verification correctness*, *evaluation correctness*, and *threshold existential unforgeability against chosen indexed message attacks (Threshold EUF-CiMA)*:

– **Partial Verification Correctness:** It signifies that any correctly produced partial signature through the `PSignGen` passes the `PSignVerfiy` algorithm. TSPS fulfills the partial verification if for all correctly indexed message $(id, M_1, M_2) \in \mathcal{M}_{iDH}^{H}$, $pp \leftarrow \texttt{Setup}(1^\lambda)$, $(\vec{sk}, \vec{vk}, vk) \leftarrow \texttt{KGen}(pp, n, t)$ that

$$\Pr\left[\texttt{PSignVerify}(pp, vk_i, (M_1, M_2), \texttt{PSignGen}(pp, sk_i, (id, M_1, M_2)) = 1\right] = 1$$

– **Evaluation Correctness:** It signifies that `Reconst` algorithm applied to a collection of well-formed (all with the same index) partial signatures $\{\sigma_i, i\}_{i\in\mathcal{T}}$ yields a valid aggregated signature. TSPS fulfills the evaluation correctness if if for all appropriately indexed message $(id, M_1, M_2) \in \mathcal{M}_{iDH}^{H}$, $pp \leftarrow \texttt{Setup}(1^\lambda)$, $(\vec{sk}, \vec{vk}, vk) \leftarrow \texttt{KGen}(pp, n, t)$, and $\mathcal{T} \in [1, n]$, $|\mathcal{T}| = t$ that

$$\Pr\begin{bmatrix}\sigma \leftarrow \texttt{Reconst}(pp, \vec{vk}, (M_1, M_2)\{i, \texttt{PSignGen}(pp, sk_i, (id, M_1, M_2))\}_{i\in\mathcal{T}}) : \\ Verify(pp, vk, (M_1, M_2), \sigma) = 1\end{bmatrix} = 1$$

– **Threshold Existential Unforgeability Against Chosen Indexed Message Attacks (Threshold EUF-CiMA):** It ensures that an adversary controlling fewer than the threshold t participants cannot generate a valid signature on behalf of the signer even if he adaptively queries the signing oracle regarding index/message pairs of their selection. TSPS satisfies Threshold EUF-CiMA if for all PPT adversaries playing game $\mathcal{G}^{T-EUF-CiMA}$ (Figure 4), the success probability is negligible:

$$Adv_{\mathcal{A}}^{T-EUF-CiMA}(\lambda) = Pr[\mathcal{G}_{\mathcal{A}}^{T-EUF-CiMA}(1^\lambda) = 1] \leq negl(\lambda) = 1,$$

where

$$\mathcal{Q}_S = \{(id_i, M_1, M_2)\}$$

is a set of signed messages, and

$$\mathcal{Q}_{EQ} = \{EQ(M_1, M_2)\}_i$$

is the set of trivially forgeable messages, which includes all equivalent messages within the verification message space.
The security proof for TSPS can be found in [19, Section 5.2]

$\underline{\mathcal{G}_{\mathcal{A}}^{\mathsf{T-EUF-CiMA}}(1^\lambda)}$

$pp \leftarrow \mathtt{Setup}(1^\lambda)$;
$Cor \xleftarrow{\$} \mathcal{A}(pp)$ `// set of corrupt signers`
If $Cor \notin [1,n] \vee |Cor| > t-1$: **return** $\bot$
else: $\mathcal{H} \leftarrow [1,n] \setminus Cor$ `// set of honest signers`
$(\vec{sk}, \vec{vk}, vk) \leftarrow \mathtt{KGen}(pp, n, t)$ and $(M_1^* M_2^*, \sigma^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{PSign}}}(\{sk_i\}_{i \in Cor}, \vec{vk}, vk)$
return $((M_1^*, M_2^*) \notin \mathcal{Q}_{EQ} \wedge \mathtt{Verify}(pp, vk, (M_1^*, M_2^*), \sigma^*))$

Oracle $\mathcal{O}_{\mathsf{PSign}}(k, id, (M_1, M_2))$

If $(k \notin H \vee (k, id, .) \in \mathcal{Q}_S \vee (., id, M_1', M_2') \in \mathcal{Q}_S, (M_1', M_2') \neq (M_1, M_2)$:
return $\bot$
else: $\sigma_k \leftarrow \mathtt{PSign}(pp, sk_k, (id, M_1, M_2))$,
$\mathcal{Q}_S \leftarrow \mathcal{Q}_S \cup \{(k, id, M_1, M_2)\}$, $\mathcal{Q}_{EQ} \leftarrow \mathcal{Q}_{EQ} \cup \{EQ(M_1, M_2)\}$

Fig. 4. Game $\mathcal{G}_{\mathcal{A}}^{\mathsf{T-EUF-CiMA}}$

3 IM SPS and Threshold SPS with Randomizable Keys

Now, we introduce the randomizable key-enabled variant of the IM SPS and TSPS schemes described in Section 2.3 and 2.4.

3.1 Indexed Message Structure Preserving Signatures with Randomizable Key(IM-SPSwRK)

To clarify the foundational structure of our protocol, we initially present its non-threshold version, the IM-SPS scheme in Section 2.3. Figure 5 shows the details of IM-SPS with a randomizable key, IM-SPSwRK. It is equivalent to IM-SPS, with the added specification of SwRK algorithms `RandPK`, `RandSK`, and `Adapt`. While `RandSK` introduces randomness to the secret key, `RandPK` applies same randomness to the public key. Furthermore, `Adapt` modifies an existing signature to align with the randomized key, while keeping the original signing key confidential.

3.2 Security

In this section we will prove that IM-SPSwRK satisfies unforgeability, unlinkability and unextractability.

Next we will give the unforgeability proof for IM-SPSwRK.

Theorem 1. *Assume the unforgeability game stated in Definition 6. In the generic bilinear group model, after at most q queries to the oracle $\mathcal{O}$ and q_G*

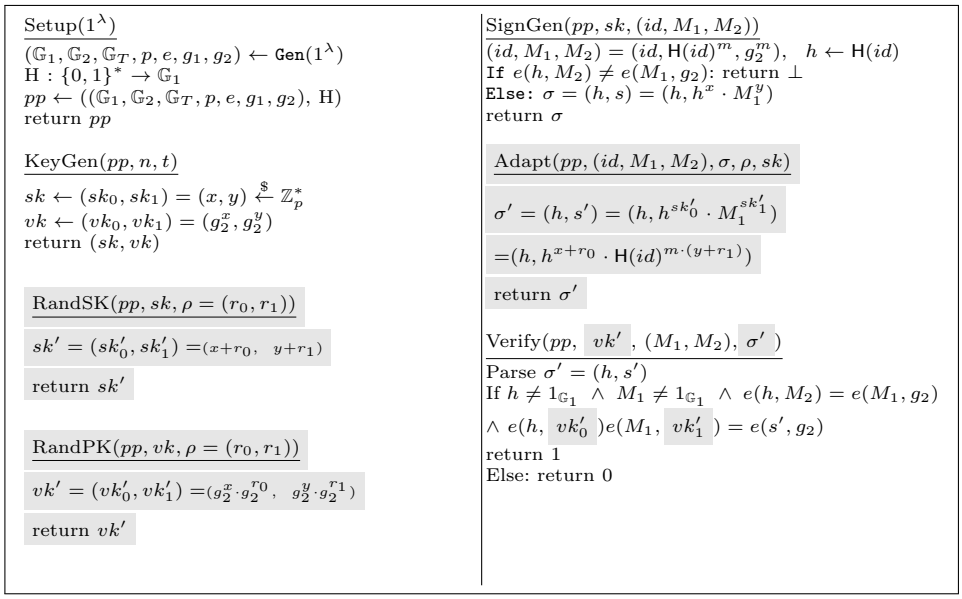

Fig. 5. Our proposed IM-SPS scheme with a randomizable key (IM-SPSwRK). The differences with the traditional IM-SPS are highlighted using grey boxes

queries to the generic group oracles, the probability that any probabilistic polynomial-time adversary outputs a fresh scalar m^* together with a pair (h, M_1) satisfying

$$P \ = \ \big(h, \ h^{\,x} \cdot {M_1}^{\,y}\big) \quad \textit{with} \quad h \neq 1_{\mathbb{G}_1}, \ M_1 \neq 1_{\mathbb{G}_1}, \quad \textit{and} \quad m^* \notin \mathcal{Q}_{\mathcal{O}}$$

is bounded by the probability

$$\frac{6(q+q_G)^2}{p}$$

which is negligible in the security parameter κ.

Proof. Using the same method in [43] we will show that the above assumption holds, independent from the randomizer. Assume we have the same setup as Definition 6, i.e. let $pp = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \hat{g})$ and let $\rho = (\rho_0, \rho_1)$ be randomizer and let $x', y' \in \mathbb{Z}_p$ random and let $x = x' + \rho_0$ and similarly $y = y' + \rho_1$. And finally let $\mathcal{A}$ be an algebraic adversary under Algebraic Asymmetric Bilinear Group Model [9]. In particular, assume that $U_i = (h_i, t_i)$ where $t_i = h_i^x \cdot M_{1,i}^y = h_i^{x+m_i \cdot y}$ is the result of the ith oracle query on input m_i for $i = 1, 2, \ldots, q$. Following Definition [9, Definition 2.4] adversary is able to write any element in $Z \in \mathbb{G}_1$ as the combination of the elements received from oracle. Now assume on input $m^* \neq m_i$, $\forall i \in [1, q]$ the output (h^*, t^*) is valid. This means we can write h^* and t^* as the combination of elements received so far;

$$h^* = g^{r^*} = \prod_{i=1}^{q} h_i^{u_{i,1}} \cdot t_i^{v_{i,1}}$$

and

$$t^* = g^{z^*} = \prod_{i=1}^{q} h_i^{u_{i,2}} \cdot t_i^{v_{i,2}}$$

for some scalars $(u_{i,1}, u_{i,2}, v_{i,1}, v_{i,2}) \in \mathbb{Z}_p^4, \forall i \in [1, q]$. Since h_i is element in $\mathbb{G}_1$ we can write $h_i = g^{r_i}$ for $r_i \in \mathbb{Z}_p \forall i \in [1, q]$. And it follows that

$$r^* = \sum_{i=1}^{q} (u_{i,1} \cdot r_i + v_{i,1}(x + m_i \cdot y) \cdot r_i)$$
$$z^* = \sum_{i=1}^{q} (u_{i,2} \cdot r_i + v_{i,2}(x + m_i \cdot y) \cdot r_i)$$

Since the output (h^*, t^*) is valid, the equation $z^* = r^*(x + m^* \cdot y)$ must hold. Then,

$$\begin{aligned}
&\sum_{i=1}^{q} ((u_{i,1} \cdot r_i \cdot x) + (v_{i,1} \cdot r_i \cdot x^2) + (v_{i,1} \cdot m_i \cdot r_i \cdot x \cdot y)) + \\
&+ \sum_{i=1}^{q} ((u_{i,1} \cdot m^* \cdot r_i \cdot y) + (v_{i,1} \cdot m^* \cdot r_i \cdot x \cdot y) + (v_{i,1} \cdot m_i \cdot m^* \cdot r_i \cdot y^2)) + \\
&+ \sum_{i=1}^{q} (-(u_{i,2} \cdot r_i) - (v_{i,2} \cdot r_i \cdot x) - (v_{i,2} \cdot m_i \cdot r_i \cdot y)) = 0
\end{aligned}$$

If $r_i = 0, \ \forall i \in [1, q]$ it would mean that $h_i = 1$. Therefore, from the above multivariate polynomial it is easy to see that $u_{i,2} = 0$ because of the monomial with variable r_i, $v_{i,1} = 0$ because of the monomial with variable x^2 and $u_{i,1} = v_{i,2}$ due to monomial with variable x for all i. Then the equation above can be simplified to

$$\sum_{i=1}^{q} ((u_{i,1} \cdot m^* \cdot r_i \cdot y) - (v_{i,2} \cdot m_i \cdot r_i \cdot y) = 0$$

Thus, it will indicate that $m^* = m_i$, which results in contradiction. Therefore, we can dedicate that an adversary can not produce a valid output for an unknown value. What's left to calculate is the probability that matching answer from the oracle randomly. Let the polynomial f be the answer of oracle $\mathcal{O}$. f is degree of maximum 2. With addition of the group element of degree 1 used in pairing calculations, f can be regarded as a polynomial of max degree 3. Let number of group-query be q_G. Then there exist at most $3 + 2q + q_G$, therefore there exist at most

$$\binom{3 + 2q + q_G}{2} = (3 + 2q + q_G)^2/2$$

distinct polynomials that could result in equal value. With the Schwartz-Zippel lemma 1, the probability that two polynomials have equal value randomly is

bounded by

$$\frac{(3+2q+q_G)^2}{2p} \leq \frac{6(q+q_G)^2}{p}$$

which would be negligible.

Next, we will show that the our scheme provides unlinkability and unextractability.

Proposition 1. *IM-SPSwRK provides unlinkability.*

Proof. Unlinkability of public keys. Since the randomizer is selected uniformly random from $\mathbb{Z}_p$ and multiplied with group element, the result is statistically independent from public key.
Unlinkability of signatures Since IM-SPSwRK provides Adapt algorithm, unlinkability follows directly.

Proposition 2. *IM-SPSwRK provides* $(0,1)-$*unextractability[15, Definition 17].*

Proof. Assume challenger runs $\mathsf{Setup}(1^\lambda)$ to get the public paramters pp and runs KeyGen twice and obtains the key pairs (sk_i, vk_i) for $i = 0, 1$ where $sk_i = (sk_{i0}, sk_{i1})$ and $vk_i = (vk_{i0}, vk_{i1})$. Also assume challenger samples a random bit $b \in \{0,1\}$ and a randomizer $\rho = (\rho_0, \rho_1) \in \mathbb{Z}_p$ and computes the challenge key $vk'_b = \mathsf{RandPK}(vk_b, \rho)$. Challenger returns (vk'_b, ρ) to the adversary. Adversary $\mathcal{A}$ also has access to the oracle $\mathcal{O}_{\mathsf{Sign}}$ which in query on indexed message (id, M_1, M_2), $\mathcal{O}_{\mathsf{Sign}}$ returns $\mathsf{Adapt}(\mathsf{SignGen}(pp, sk, \rho, (id, M_1, M_2)))$. Since the distribution of the secret keys and randomizer are uniform in $\mathbb{Z}_p$ and independent from the choice of b, the distribution of vk'_b is also uniform in $\mathbb{G}_2$. A valid signature $\sigma = (h, s)$ on indexed message (id, M_1, M_2) after randomization

$$s = h^{sk_{b0}+\rho_0} \cdot M_1^{sk_{b1}+\rho_1}$$

depends on single group element h which is uniform in $\mathbb{G}_1$ and independent from the choice of b. In conclusion, even with the knowledge of (vk'_b, ρ) and signatures on the indexed messages, adversary $\mathcal{A}$ does not gain any information on b. Hence IM-SPSwRK satisfies $(0,1)-$unextractability.

3.3 Threshold Structure Preserving Signatures with Randomizable Key(TSPSwRK)

Our proposed Threshold Structure-Preserving Signatures with Randomizable Keys (TSPSwRK) starts with a distributed key generation process, which allows each node to generate its secret and verification key independently. Subsequently, the messages are indexed with indexed diffie-hellman message space to preserve threshold structure. For the randomizable key property, each node introduces additional randomness to its existing key pair using the key randomizer chosen by a central authority. In the signing process, each node employs its randomized key to produce a partial signature. The partial signatures generated from randomized

keys are subsequently combined by satisfying the threshold criterion, typically t partial signatures from n nodes, to create the main signature. More formally, we describe our protocol in the following.

Threshold Protocol. Figure 7 shows the details of TSPSwRK. TSPSwRK is equivalent to TSPS, with the added specification of SwRK algorithms `RandPK`, and `Adapt`. In addition, the threshold version requires `RandGen`, `RandSKShare`, and `RandPKShare` algorithms.

The `Setup` algorithm in Figure 7 accepts the security parameter 1^λ and outputs the public parameters pp for the TSPS. The public parameters consist of groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of orders p used in bilinear map e, along with the generators g_1 and g_2 for $\mathbb{G}_1$, and $\mathbb{G}_2$, respectively. Additionally, there is a hash function that maps to $\mathbb{G}_1$ allowing messages to be securely represented as elements within the group.

`KeyGen` algorithm allows to each node N_i to independently generate its secret key sk_i and its corresponding verification key vk_i. After the generation of individual secret keys $\vec{sk} = (sk_1, \ldots, sk_n)$ and their corresponding verification keys $\vec{vk} = (vk_1, \ldots, vk_n)$, the algorithm generates the main verification key vk.

`RandGen` algorithm is run with a Distributed Key Generation (DKG) protocol without the involvement of a trusted entity. In scenarios where a trusted authority is responsible for randomizing the signature and public key, a Verifiable Random Function (VRF) [39] can be employed. Alternatively, a Distributed Verifiable Random Function (DVRF) [30,3] can be also utilized by the signers to achieve the same purpose.

`RandSKShare` and `RandPKShare` algorithm are executed by each node to randomize its secret key as sk'_i and its corresponding public verification key vk'_i, respectively. Also, `RandPK` algorithm generates the randomized main verification key vk'. Furthermore, since the Pedersen Distributed Key Generation (DKG) algorithm intrinsically includes its own proofs, a distinct VerKey algorithm is not established to validate correctness of the randomization process.

`PSignGen` algorithm generates the partial signatures σ_i on the indexed message (id, M_1, M_2) under the node's secret key sk_i, while `Adapt` algorithm outputs the randomized partial signature σ'_i under the node's randomized secret key sk'_i.

`Reconst` algorithm generates the randomized main signature σ aggregating threshold t randomized partial signatures σ'_i after verifying the correctness of them via `PSignVerify`. At the end of the scheme, the randomized main signature σ' is verified under randomized main verification key vk'.

3.4 Security

Correctness. First, we demonstrate that it fulfills the criteria for *partial evaluation correctness*, ensuring that a correctly created partial signature via the `PSignGen` successfully passes the `PSignVerfiy` checks after `Adapt` algorithm. Indeed, for all $i \in [1, n]$ and correctly indexed message $(id, M_1, M_2) \in \mathcal{M}^H_{iDH}$, it holds that:

$$e(h_i, vk'_{i0})e(M_1, vk'_{i1}) = e(h_i, g_2^{x_i + r_{i0}})e(h_i^m, g_2^{y_i + r_{i1}}) = e(s'_i, g_2) \tag{1}$$

Next, we show that it satisfies evaluation correctness, ensuring that for a set of partial signatures $\{i, \sigma'_i\}_{i\in\mathcal{T}}, \mathcal{T} \subset [1,n], |\mathcal{T}| = t$ on an indexed message $(id, M_1, M_2) = (id, \mathsf{H}(id)^m, g_2^m) \in \mathcal{M}_{iDH}^H$ with the same $h \leftarrow \mathsf{H}(id)$, `Reconst` algorithm successfully generates the valid aggregated signature $\sigma = (h, s)$ as follows:

$$\begin{aligned} s &= \prod_{i\in\mathcal{T}} s_i'^{\lambda_i} = \prod_{i\in\mathcal{T}} (h^{sk'_{i0}} M_1^{sk'_{i1}})^{\lambda_i} \\ &= h^{\sum_{i\in\mathcal{T}}(x_i\cdot\lambda_i + r_{i0}\cdot\lambda_i)} M_1^{\sum_{i\in\mathcal{T}}(y_i\cdot\lambda_i + r_{i1}\cdot\lambda_i)} \\ &= h^{(\sum_{i\in\mathcal{T}}(x_i\cdot\lambda_i) + \sum_{i\in\mathcal{T}}(r_{i0}\cdot\lambda_i))} M_1^{(\sum_{i\in\mathcal{T}}(y_i\cdot\lambda_i) + \sum_{i\in\mathcal{T}}(r_{i1}\cdot\lambda_i))} \\ &= h^{x+r_0} M_1^{y+r_1} \end{aligned} \tag{2}$$

Next we start unforgeability proof by introducing the Unforgeability Under Randomization game.

$\underline{\mathcal{G}_{\mathcal{A}}^{\mathsf{T-EUF-CiMA-UR}}(1^\lambda)}$

$pp \leftarrow \mathtt{Setup}(1^\lambda)$;
$Cor \xleftarrow{\$} \mathcal{A}(pp)$ `// set of corrupt signers`
If $Cor \notin [1,n] \vee |Cor| > t-1$: **return** $\perp$
else: $\mathcal{H} \leftarrow [1,n] \setminus Cor$ `// set of honest signers`
$(\vec{sk}, \vec{vk}, vk) \leftarrow \mathtt{KGen}(pp, n, t)$ and $vk^* \leftarrow \mathtt{RandPK}(pp, vk, \rho_k = (\rho_0, \rho_1))$ and $(M_1^* M_2^*, \sigma^*) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{PSign}}}(\{sk_i\}_{i\in Cor}, \vec{vk}, vk)$
return $((M_1^*, M_2^*) \notin \mathcal{Q}_{EQ} \wedge \mathtt{Verify}(pp, vk, (M_1^*, M_2^*), \sigma^*))$ or $((M_1^*, M_2^*) \notin \mathcal{Q}_{EQ} \wedge \mathtt{Verify}(pp, vk^*, (M_1^*, M_2^*), \sigma^*))$

Oracle $\mathcal{O}_{\mathsf{PSign}}(k, id, (M_1, M_2))$

If $(k \notin H \vee (k, id, .) \in \mathcal{Q}_S \vee (., id, M_1', M_2') \in \mathcal{Q}_S, (M_1', M_2') \neq (M_1, M_2)$:
return $\perp$
else: $\sigma_k \leftarrow \mathtt{PSign}(pp, sk_k, (id, M_1, M_2))$,
$\mathcal{Q}_S \leftarrow \mathcal{Q}_S \cup \{(k, id, M_1, M_2)\}$, $\mathcal{Q}_{EQ} \leftarrow \mathcal{Q}_{EQ} \cup \{EQ(M_1, M_2)\}$

Fig. 6. Game $\mathcal{G}_{\mathcal{A}}^{\mathsf{T-EUF-CiMA-UR}}$

Theorem 2. *Let the TSPS scheme defined as same as in 7. If TSPS is EUF-CiMA secure against the game $\mathcal{G}_{\mathcal{A}}^{\mathsf{T-EUF-CiMA}}$ defined in Figure 4, then TSPSwRK is existentially unforgeable under the game $\mathcal{G}_{\mathcal{A}}^{\mathsf{T-EUF-CiMA-UR}}$ defined in Figure 6.*

Proof. Assume there exists an efficient adversary $\mathcal{A}$ that can break $\mathcal{G}_{\mathcal{A}}^{\mathsf{T\text{-}EUF\text{-}CiMA\text{-}UR}}$. Then we can construct an efficient adversary $\mathcal{A}'$ that breaks $\mathcal{G}_{\mathcal{A}}^{\mathsf{T\text{-}EUF\text{-}CiMA}}$ as follows: $\mathcal{A}$ forwards every signing query it receives from $\mathcal{A}'$ to its own signing

oracle and relays the answer back to $\mathcal{A}'$. When $\mathcal{A}$ outputs the tuple (m^*, σ^*, ρ^*), where m^* has not been queried to the oracle, $\sigma^* = (h, s^*)$ is the signature and $\rho^* = (\rho_0^*, \rho_1^*)$, the reduction outputs (m^*, σ) with $\sigma = (h, s)$, where s is defined as

$$s := \begin{cases} s^*, & \text{if } \sigma^* \text{ is a valid signature for } m^* \text{ under the public key } vk^* \\ \frac{s}{h^{\rho_0^*} \cdot M_1^{\rho_0^*}}, & \text{otherwise} \end{cases}$$

This allows $\mathcal{A}$ to revert the randomisation of the signature σ. Thus, if $\mathcal{A}$ outputs a valid forgery for a randomisation of vk, the adversary $\mathcal{A}'$ can use it to generate a forgery for vk.

Unlinkability and Unextractability . Assuming an adversary holds at most $t - 1$ corrupt participants where t is the threshold in TSPSwRK, the arguments in proof of unlinkability 1 and proof of unextractability 2 both apply also to TSPSwRK since the reconstructed signature in TSPSwRK and IM-SPSwRK calculated with identical parameters.

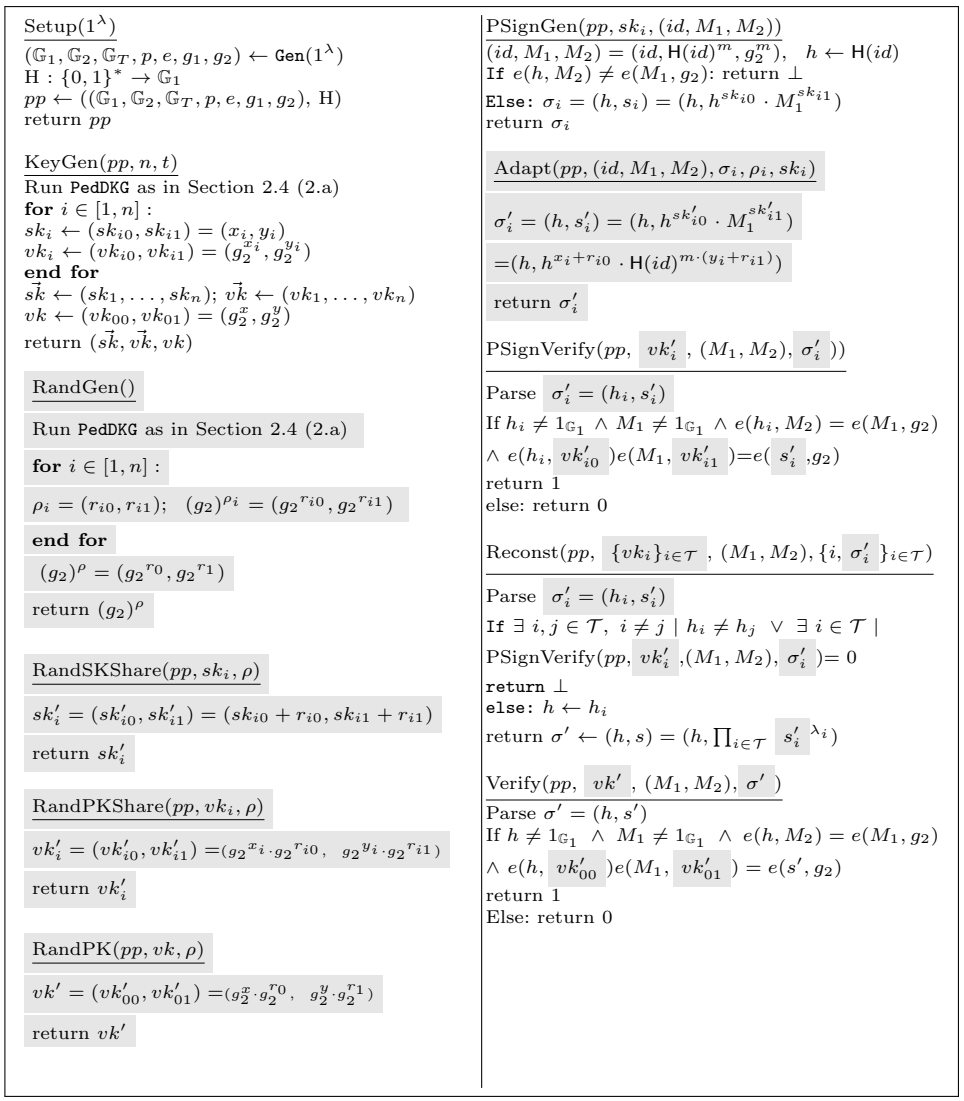

Fig. 7. Our proposed TSPSwRK scheme. The differences with the traditional TSPS are highlighted using grey boxes

4 Analysis

4.1 Computational Cost Analysis

Since our scheme relies on bilinear maps, our computations are performed over the groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, with pairing operation $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Table **??** shows the detailed computational cost of each phase in our proposed TSPSwRK scheme relative to a standard TSPS. We also provide a comparison with its non-threshold variant, IM-SPSwRK, with the original IM-SPS. In this table, while $\text{Exp}_{\mathbb{G}_i}$ represents the exponentiation in the Group $\mathbb{G}_i$, $\text{Mul}_{\mathbb{G}_i}$ denotes the multiplication in the Group $\mathbb{G}_i$, where $i \in \{1, 2, T\}$. Also, pairing represents the pairing operation $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, and Hash represents the hash operation in $\mathbb{G}_1$.

Phases	IM-SPS	IM-SPSwRK	TSPS	TSPSwRK
Key Generation	$2\,\text{Exp}_{\mathbb{G}_2}$	$4t + 2\,\text{Exp}_{\mathbb{G}_2}$, $4t - 4\,\text{Mul}_{\mathbb{G}_2}$	$n + 1\,\text{Exp}_{\mathbb{G}_2}$	$4t + 2\,\text{Exp}_{\mathbb{G}_2}$, $4t - 4\,\text{Mul}_{\mathbb{G}_2}$
RandGen				$4t + 4\,\text{Exp}_{\mathbb{G}_2}$, $4t - 4\,\text{Mul}_{\mathbb{G}_2}$
RandPKShare				$2\,\text{Exp}_{\mathbb{G}_2}$, $2\,\text{Mul}_{\mathbb{G}_2}$
RandPK		$2\,\text{Exp}_{\mathbb{G}_2}$, $2\,\text{Mul}_{\mathbb{G}_2}$		$2\,\text{Exp}_{\mathbb{G}_2}$, $2\,\text{Mul}_{\mathbb{G}_2}$
SignGen/PSignGen	2 pairing, $2\,\text{Exp}_{\mathbb{G}_1}$, $1\,\text{Mul}_{\mathbb{G}_1}$	$3\,\text{Exp}_{\mathbb{G}_1}$, $1\,\text{Mul}_{\mathbb{G}_1}$, $1\,\text{Exp}_{\mathbb{G}_2}$, $1\,\text{Hash}(H_1)$, 1 pairing	2 pairing, $1\,\text{Mul}_{\mathbb{G}_1}$, $1\,\text{Mul}_{\mathbb{G}_1}$	$3\,\text{Exp}_{\mathbb{G}_1}$, $1\,\text{Mul}_{\mathbb{G}_1}$, $1\,\text{Exp}_{\mathbb{G}_2}$, $1\,\text{Hash}(H_1)$, 1 pairing
Adapt		$2\,\text{Exp}_{\mathbb{G}_1}$, $1\,\text{Mul}_{\mathbb{G}_1}$		$2\,\text{Exp}_{\mathbb{G}_1}$, $1\,\text{Mul}_{\mathbb{G}_1}$
PSignVerify			5 pairing, $1\,\text{Mul}_{\mathbb{G}_1}$	5 pairing $1\,\text{Mul}_{\mathbb{G}_1}$
Reconst			5 pairing, $t - 1\,\text{Mul}_{\mathbb{G}_1}$, $t\,\text{Exp}_{\mathbb{G}_1}$, $t\,\text{Exp}_{\mathbb{G}_1}$	5 pairing, $t\text{Exp}_{\mathbb{G}_1}$, $t\text{Mul}_{\mathbb{G}_1}$
Verify	5 pairing, $1\,\text{Mul}_{\mathbb{G}_1}$	5 pairing	5 pairing, $1\,\text{Mul}_{\mathbb{G}_1}$	5 pairing

4.2 Implementation and evaluation

The PoC implementation of the TSPSwRK procotol can be found on [**?**]. All experiments were carried out on a 64-bit operating system with a 6-core CPU running at approximately 3.2 GHz and 16 GB of RAM. The system did not have any specialized hardware acceleration. The protocol was executed at least 10

times for each configuration, parameterized by the total number of participants (n) and the threshold (t). For each run, the execution time of protocol phases was recorded. The mean execution time for each component was then computed for each (n, t) configuration. Figure 8 shows these results.

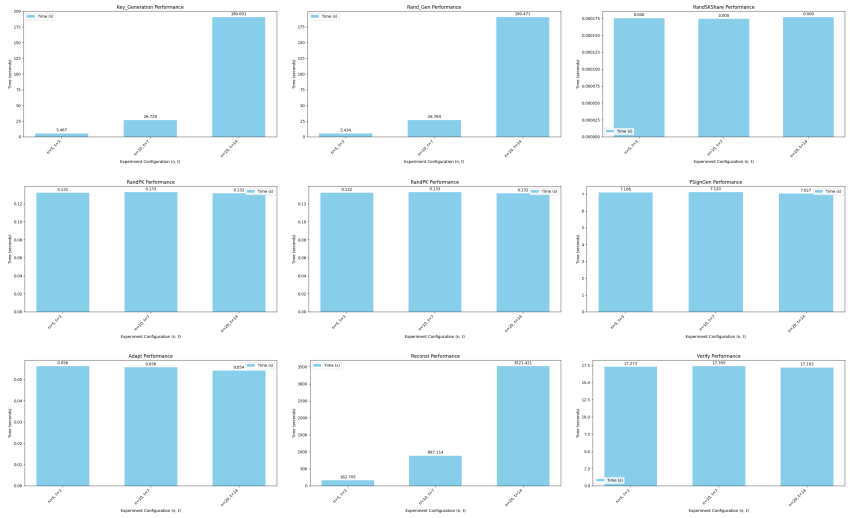

Fig. 8. Permormance metrics of the implementation

5 Conclusion

In this study, we introduced Threshold Structure-Preserving Signatures with Randomizable Keys (TSPSwRK). The fundamental concept of TSPSwRK is to guarantee that the signing key stays unlinkable throughout key randomization while yet permitting the valid aggregation of partial signatures into a final, verifiable signature.

References

1. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Advances in Cryptology–CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings 30. pp. 209–236. Springer (2010)
2. Abe, M., Groth, J., Ohkubo, M., Tibouchi, M.: Structure-preserving signatures from type II pairings. In: Advances in Cryptology–CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I 34. pp. 390–407. Springer (2014)

3. Ağırtaş, A.R., Özer, A.B., Saygı, Z., Yayla, O.: Distributed verifiable random function with compact proof. In: Dolev, S., Elhadad, M., Kutyłowski, M., Persiano, G. (eds.) Cyber Security, Cryptology, and Machine Learning. pp. 119–134. Springer Nature Switzerland, Cham (2025)
4. Alkeilani Alkadri, N., Das, P., Erwig, A., Faust, S., Krämer, J., Riahi, S., Struck, P.: Deterministic wallets in a quantum world. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. pp. 1017–1031 (2020)
5. Atapoor, S., Baghery, K., Cozzo, D., Pedersen, R.: CSI-SharK: CSI-Fish with sharing-friendly keys. In: Australasian Conference on Information Security and Privacy. pp. 471–502. Springer (2023)
6. Ağırtaş, A., Çelik, E., Kocaman, S., Sulak, F., Yayla, O.: Threshold structure-preserving signatures with randomizable key. In: Proceedings of the 22nd International Conference on Security and Cryptography - Volume 1: SECRYPT. pp. 632–637. INSTICC, SciTePress (2025). https://doi.org/10.5220/0013528900003979
7. Backes, M., Hanzlik, L., Kluczniak, K., Schneider, J.: Signatures with flexible public key: Introducing equivalence classes for public keys. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 405–434. Springer (2018)
8. Balumuri, S., Eaton, E., Lamontagne, P.: Quantum-safe public key blinding from MPC-in-the-head signature schemes. Cryptology ePrint Archive (2024)
9. Bauer, B., Fuchsbauer, G., Loss, J.: A classification of computational assumptions in the algebraic group model. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020. pp. 121–151. Springer International Publishing, Cham (2020)
10. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Advances in Cryptology-CRYPTO 2009: 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings. pp. 108–125. Springer (2009)
11. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-Fish: efficient isogeny based signatures through class group computations. In: International conference on the theory and application of cryptology and information security. pp. 227–247. Springer (2019)
12. Beullens, W., Delpech de Saint Guilhem, C.: LegRoast: Efficient post-quantum signatures from the legendre PRF. In: International Conference on Post-Quantum Cryptography. pp. 130–150. Springer (2020)
13. Blazy, O., Canard, S., Fuchsbauer, G., Gouget, A., Sibert, H., Traoré, J.: Achieving optimal anonymity in transferable e-cash with a judge. In: International Conference on Cryptology in Africa. pp. 206–223. Springer (2011)
14. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. Journal of cryptology **17**, 297–319 (2004)
15. Celi, S., Griffy, S., Hanzlik, L., Kempner, O.P., Slamanig, D.: Sok: Signatures with randomizable keys. Cryptology ePrint Archive (2023)
16. Chase, M., Derler, D., Goldfeder, S., Orlandi, C., Ramacher, S., Rechberger, C., Slamanig, D., Zaverucha, G.: Post-quantum zero-knowledge and signatures from symmetric-key primitives. In: Proceedings of the 2017 acm sigsac conference on computer and communications security. pp. 1825–1842 (2017)
17. Cini, V., Ramacher, S., Slamanig, D., Striecks, C., Tairi, E.: Updatable signatures and message authentication codes. In: IACR International Conference on Public-Key Cryptography. pp. 691–723. Springer (2021)

18. Connolly, A., Lafourcade, P., Perez Kempner, O.: Improved constructions of anonymous credentials from structure-preserving signatures on equivalence classes. In: IACR International Conference on Public-Key Cryptography. pp. 409–438. Springer (2022)
19. Crites, E., Kohlweiss, M., Preneel, B., Sedaghat, M., Slamanig, D.: Threshold structure-preserving signatures. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 348–382. Springer (2023)
20. Crites, E.C., Lysyanskaya, A.: Delegatable anonymous credentials from mercurial signatures. In: Cryptographers' Track at the RSA Conference. pp. 535–555. Springer (2019)
21. Das, P., Faust, S., Loss, J.: A formal treatment of deterministic wallets. In: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security. pp. 651–668 (2019)
22. Denis, F., Eaton, E., Lepoint, T., Wood, A, C.: Key blinding for signature schemes, `https://datatracker.ietf.org/doc/draft-irtf-cfrg-signature-key-blinding/`, Internet Engineering Task Force, September 2024
23. Derler, D., Slamanig, D.: Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. Designs, Codes and Cryptography **87**, 1373–1413 (2019)
24. Eaton, E., Lepoint, T., Wood, C.A.: Security analysis of signature schemes with key blinding. Cryptology ePrint Archive (2023)
25. Eaton, E., Stebila, D., Stracovsky, R.: Post-quantum key-blinding for authentication in anonymity networks. In: Progress in Cryptology–LATINCRYPT 2021: 7th International Conference on Cryptology and Information Security in Latin America, Bogotá, Colombia, October 6–8, 2021, Proceedings 7. pp. 67–87. Springer (2021)
26. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: 28th Annual Symposium on Foundations of Computer Science (sfcs 1987). pp. 427–438. IEEE (1987)
27. Fleischhacker, N., Krupp, J., Malavolta, G., Schneider, J., Schröder, D., Simkin, M.: Efficient unlinkable sanitizable signatures from signatures with re-randomizable keys. In: Public-Key Cryptography–PKC 2016: 19th IACR International Conference on Practice and Theory in Public-Key Cryptography, Taipei, Taiwan, March 6-9, 2016, Proceedings, Part I. pp. 301–330. Springer (2016)
28. Fuchsbauer, G.: Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. Cryptology ePrint Archive (2009)
29. Fuchsbauer, G., Vergnaud, D.: Fair blind signatures without random oracles. In: Progress in Cryptology–AFRICACRYPT 2010: Third International Conference on Cryptology in Africa, Stellenbosch, South Africa, May 3-6, 2010. Proceedings 3. pp. 16–33. Springer (2010)
30. Galindo, D., Liu, J., Ordean, M., Wong, J.M.: Fully distributed verifiable random functions and their application to decentralised random beacons. In: 2021 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 88–102. IEEE (2021)
31. Ghadafi, E.: Short structure-preserving signatures. In: Cryptographers' Track at the RSA Conference. pp. 305–321. Springer (2016)
32. Gouvea, C.P., Komlo, C.: Re-randomized FROST. Cryptology ePrint Archive (2024)
33. Groth, J.: Fully anonymous group signatures without random oracles. In: Advances in Cryptology–ASIACRYPT 2007: 13th International Conference on the Theory

and Application of Cryptology and Information Security, Kuching, Malaysia, December 2-6, 2007. Proceedings 13. pp. 164–180. Springer (2007)
34. Hanser, C., Slamanig, D.: Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In: Advances in Cryptology–ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, ROC, December 7-11, 2014. Proceedings, Part I 20. pp. 491–511. Springer (2014)
35. Jaeger, J., Stepanovs, I.: Optimal channel security against fine-grained state compromise: The safety of messaging. In: Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I 38. pp. 33–62. Springer (2018)
36. Kiltz, E., Lyubashevsky, V., Schaffner, C.: A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In: Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29-May 3, 2018 Proceedings, Part III 37. pp. 552–586. Springer (2018)
37. Kim, H., Sanders, O., Abdalla, M., Park, J.H.: Practical dynamic group signatures without knowledge extractors. Cryptology ePrint Archive, Paper 2021/351 (2021), https://eprint.iacr.org/2021/351
38. Libert, B., Peters, T., Joye, M., Yung, M.: Linearly homomorphic structure-preserving signatures and their applications. Designs, Codes and Cryptography **77**, 441–477 (2015)
39. Micali, S., Rabin, M., Vadhan, S.: Verifiable random functions. In: 40th annual symposium on foundations of computer science (cat. No. 99CB37039). pp. 120–130. IEEE (1999)
40. Mir, O., Bauer, B., Griffy, S., Lysyanskaya, A., Slamanig, D.: Aggregate signatures with versatile randomization and issuer-hiding multi-authority anonymous credentials. In: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security. pp. 30–44 (2023)
41. Nanri, M., Kempner, O.P., Tibouchi, M., Abe, M.: Interactive threshold mercurial signatures and applications. Cryptology ePrint Archive (2024)
42. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Annual international cryptology conference. pp. 129–140. Springer (1991)
43. Pointcheval, D., Sanders, O.: Short randomizable signatures. In: Topics in Cryptology-CT-RSA 2016: The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29-March 4, 2016, Proceedings. pp. 111–126. Springer (2016)
44. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979)
45. Shaw, S., Dutta, R.: Compact stateful deterministic wallet from isogeny-based signature featuring uniquely rerandomizable public keys. In: International Conference on Cryptology and Network Security. pp. 392–413. Springer (2023)