Technische Universität Darmstadt
Department of Computer Science
Cryptography and Complexity Theory

How Threshold Ring Signature Schemes hide the Signers

Bachelor's Thesis by
Philipp-Florens Lehwalder

Examiner: Prof. Dr. Marc Fischlin
Adviser: Rune Fiedler

Date of Submission: April 12, 2021

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 und § 23 Abs. 7 APB TU Darmstadt

Hiermit versichere ich, Philipp-Florens Lehwalder, die vorliegende Bachelor-Thesis gemäß § 22 Abs. 7 APB der TU Darmstadt ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Mir ist bekannt, dass im Falle eines Plagiats (§38 Abs.2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden. Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung gemäß § 23 Abs. 7 APB überein. Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

12. April 2021 ______________________________

Contents

1 Introduction

Ring signatures were originally presented by Rivest, Shamir, and Tauman in 2001 [RST01] as a derivative of group signatures, introduced by Chaum and van Heyst [CvH91]. Both concepts allow creating signatures on behalf of a group of possible signers such that the actual signer remains anonymous in this set. In contrast to group signatures, ring signatures do not require a setup phase among all participants or a central group manager who is able to revoke the signer's anonymity. Instead of having a predefined group of possible signers with ring signatures, a signer can freely select any number of foreign public keys to form a ring in which it remains hidden.

Bresson et al. [BSS02] extended this concept and proposed threshold ring (thring) signatures, which enable a group of signers to form a ring signature and to prove that the number of signers is over a defined threshold, but as for ring signatures, without revealing who belongs to the signers. Therefore, standard ring signatures can be considered as a special case of thring signatures with a threshold of one.

Possible applications for thring signatures are, for example, anonymous petitions or polls where one can verify that at least a number of t parties have agreed on a decision without exposing their identities. Considering elections with well over hundreds of thousands of eligible voters per electoral district, the fact that they do not need to run a setup protocol together is a crucial benefit compared to traditional group signatures.

Another exemplary use case is shared cryptocurrency wallets where performing a transaction requires the signature of at least t parties; with thring signatures, one could use the same key-pair across multiple wallets without the need of distributed key generation. Moreover, cryptocurrencies theirselves constitute a significant field for (threshold) ring signatures, for example, Monero[1] or Bytecoin[2], both based on the CryptoNote protocol [vS13], employ ring signatures to ensure untraceable transactions.

Leaking secrets or whistleblowing are also frequently mentioned scenarios for thring signatures. While the priority here is, of course, guaranteeing uncompromising anonymity for the whistleblowers, proving that a certain number of "insiders" indeed participated in this exposure may also give it more trustworthiness as if it was verifiable signed by only a single party.

Furthermore, as stressed by Bresson et al. [BSS02] and Chow et al. [CHY05], the advent of mobile ad-hoc networks and ubiquitous computing, where ad-hoc groups may spontaneously need to communicate with sensitive data, results in increasing importance for secure thring signatures.

Since anonymity is the key feature for (threshold) ring signatures, a throughout understanding of how threshold ring signatures hide the signers

[1]`https://www.getmonero.org/`

[2]`https://bytecoin.org/`

is essential.

Therefore, this thesis aims to provide an overview of existing schemes, to explain and compare their used techniques for ensuring anonymity while considering their assumptions, building blocks, efficiency, and signature size.

In Chapter 2, we first introduce our notations, followed by complexity assumptions and building blocks used in various thring signature schemes and are required for their examination. Formal definitions of thring signatures and their security properties are provided in Chapter 3, where we also present our classification system and discuss these distinctions in terms of anonymity. In the Chapters 4, 5, and 6, we go over three major construction types we identified for thring signatures and examine concrete schemes of each construction type with a focus on their way of ensuring anonymity. For each construction type, the schemes are again separated to whether they are linkable or unlinkable. The title of each analyzed scheme refers either to the crucial building block the scheme makes use of or to the complexity assumption of its underlying public-key cryptosystem. We want to note here that we have focused on the major construction types as they give the best overview over the majority of current thring signatures. Accordingly, there may be individual schemes that do not follow any of these three construction types; we address an observed possible fourth, relatively rare construction type in the results. In Chapter 7, we conclude with recapitulating our results and discussing some trends and future work.

This thesis's main contributions are an extensive analysis of twelve exemplary thring signature schemes, belonging to one of three major construction types we identified for thring signatures, regarding the way they hide the signers while also taking into account their efficiency and resulting signature size. To allow a systematic comparison of different thring signature schemes, all examined schemes are further classified according to our classification system.

On the one hand, our contributions are beneficial when planning to integrate thring signatures into real-world applications in terms of choosing the most suitable technique for the app-specific requirements. For example, one trade-off could be the signature size and the level of anonymity. On the other hand, when constructing new thring signature schemes, such an overview and comparison of different schemes and their analyzed techniques could likewise be conducive for composing its own secure method of hiding the signers.

2 Preliminaries

In this chapter, we first introduce notations that we use throughout this thesis and provide definitions for complexity assumptions and common building blocks that are used in various thring signature schemes we will discuss in the further course of this work.

2.1 Notations

We denote an algorithm as probabilistic polynomial time by PPT and deterministic polynomial-time by DPT.

When assigning a value y to a variable x, we use the following notation $x \leftarrow y$, if x is randomly sampled out of a set A, we write $x \leftarrow\!\!\$\, A$ and the size/order of a set A is denoted by $|A|$.

To differentiate between an ordered list of elements and a set of elements, we use the notation (x_1) and $\{x_i\}$, respectively.

When writing $[t]$ for a positive integer t, we refer to the set $\{1, \ldots, t\}$.

For a polynomial f, the notation $deg(f)$ indicates the degree of f.

The security parameter is denoted by λ and often written as 1^λ.

A function $\mathsf{negl}(x)$ is called *negligible* if for any constant $c \geq 0$ there exists an integer n such that $\forall x > n : \mathsf{negl}(x) < \frac{1}{x^c}$. When calling a problem "hard", we mean that any PPT adversary has only a negligible success probability (advantage) of solving this problem.

We consider a function f as a *one-way*, if it easy to calculate $f(x)$ for all x values in the domain of f, but it is hard to invert f, i.e. to calculate a value x' such that for a given y it holds that $f(x') = y$. If this function f is bijective, we refer to it as a *one-way-permutation* and often write it as F.

Given a matrix M or a vector v, we denote M^T or v^T as the transpose of M or v, respectively. With $||v||$ we denote the norm or length of a vector v. Lastly, the weight of a vector v corresponds to all non-zero entries of v and is denoted as $wt(v)$.

Let $\mathcal{U}$ be the set of all parties represented by their unique index i. Then, we assume that every party $i \in \mathcal{U}$ has a public- and secret key-pair, indicated by pk_i and sk_i, respectively. Furthermore, we assume that everyone is able to gather any number of foreign public keys pk_i with $i \in \mathcal{U}$.

Finally, we denote a *ring* by an list of ordered public keys $\mathbf{R} = (pk_1, ..., pk_N) = (pk_i)_{i \in \mathbf{RI}}$ with $\mathbf{RI} \subseteq \mathcal{U}$. The set of *signers* is represented by the set of indices $\mathbf{SI} \subseteq \mathbf{RI}$ and their associated secret keys are denoted by $\mathbf{S} = \{sk_i\}_{i \in \mathbf{SI}}$. The remaining parties included the ring $\mathbf{NI} = \mathbf{RI} \setminus \mathbf{SI}$ are called the *non-signers*. Note that all the sets of the indices as for the ring members, signers or non-signers, are indicated with the letter "I" (except for $\mathcal{U}$).

2.2 Complexity Assumptions

2.2.1 Random Oracle Model

The *random oracle model* (ROM), formalized by Bellare and Rogaway in 1993 [BR93], is a widely used model for cryptographic schemes where one assumes that every party has access to an oracle, used as a black box, that gives truly random but fixed responses to all queries. Fixed means that the oracle will return the same response when querying it with the same input again. So, an adversary cannot compute a response on its own but has to query the random oracle.

2.2.2 Ideal Cipher Model

In the *ideal cipher model* (ICM), one assumes that there exists a publicly accessible random cipher E_k, which is chosen uniformly at random from the family of all block permutations indexed by a l_0-bit key k and operating on blocks with l-bit sizes $\{0,1\}^l$. All parties are then able to make encryption E_k as well as decryption queries E_k^{-1} to this ideal cipher. It has been shown that the ideal cipher model and the random oracle model are equivalent in a way that the security of the ideal cipher model implies the random oracle model [CDMP05] and vice-versa [DSKT16, DS16, CPS08].

2.2.3 Discrete Logarithm Problem

Definition 1 (Discrete Logarithm Problem [McC90]) *Given a group $\mathbb{G}$ and a generator $g \in \mathbb{G}$ with $\langle g \rangle$ denoting its cyclic subgroup; The discrete logarithm problem (DLP) defines the problem of finding an integer x such that for a given element $h \in \langle g \rangle$ it holds:*

$$h = g^x$$

However, given the element x, verifying that it is indeed the correct exponent is straightforward, hence, the DLP is a candidate for an one-way function [HIL99].

2.2.4 (Strong) RSA Assumption

We follow to the definitions given by [KL14]. Given a security parameter λ the PPT algorithm GenRSA outputs a RSA public- and secret key-pair (N, e) and d, respectively. The integer N is a product of two λ-bit primes, and for $e, d > 0$ it holds that $gcd(e, \phi(N)) = 1$ and $e * d \equiv 1 \bmod \phi(N)$, with $gcd(a, b)$ returning the greatest common divisor of the two given numbers and $\phi(n)$ denoting Euler's totient function. Then, consider the game $\mathsf{Inv}_{\mathcal{A}}^{\mathsf{RSA}}(\lambda)$ between a PPT adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

1. $\mathcal{C}$ generates a RSA key-pair with $((N, e), d) \leftarrow \mathsf{GenRSA}(1^\lambda)$.

2. $\mathcal{C}$ chooses a random value $y \leftarrow\!\$\, \mathbb{Z}_N^*$ and gives (N, e) and y to $\mathcal{A}$.

3. $\mathcal{A}$ outputs a value $x \in \mathbb{Z}_N^*$ and wins if $x^e \equiv y \bmod N$.

The *RSA problem* is hard relative to GenRSA, if for every PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(x)$ such that:

$$Pr[\mathcal{A}\, wins\, \mathsf{Inv}_{\mathcal{A}}^{\mathsf{RSA}}(\lambda)] \leq \mathsf{negl}(\lambda)$$

Finally, the *RSA assumption* states that there exists a GenRSA algorithm such that the RSA problem is hard. If the adversary is allowed to choose the public exponent e, we refer to this as the *strong RSA assumption.*

2.2.5 Computational and Decisional Diffie-Hellman Problem

The computational- and decisional Diffie-Hellman problem [Bon98] are both related to the discrete logarithm problem and are considered as stronger assumptions. If the DLP is easy to calculate in a group $\mathbb{G}$, then both problems can also directly be solved in $\mathbb{G}$.

Definition 2 (Computational Diffie-Hellman Problem) *Given a finite, cyclic group $\mathbb{G}$ of order N with a generator g, the computational Diffie-Hellman problem (CDHP) asks to compute g^{xy} for the tuple (g^x, g^y).*

Definition 3 (Decisional Diffie-Hellman Problem) *Given a finite, cyclic group $\mathbb{G}$ of order N with a generator g, the decisional Diffie-Hellman problem (DDHP) asks to decide between the two triples (g^x, g^y, g^{xy}) and (g^x, g^y, g^z), where $x, y, z \in \mathbb{Z}_N^*$ are random integers.*

Assuming the hardness of the DDHP, i.e., that both triples should be computationally indistinguishable, is a stronger assumption than on the CDHP since the DDHP can be trivially solved if the CDHP is easy.

2.2.6 Problems in Coding Theory

We follow the definitions of [DV09]. A $[n, k]$-binary linear code $\mathcal{C}$ is a k-dimensional vector subspace of $\mathbb{F}_2^n$. A code $\mathcal{C}$ is defined by a $(n-k) \times n$ binary parity check matrix H; for all words $x \in \mathcal{C}$ included in the code, referred to codewords, it holds that $Hx^T = 0$. For a word $x \in \mathbb{F}_2^n$ its syndrome s corresponds to $Hx^T = s$.

Definition 4 (Syndrome Decoding Problem [AMCG08]) *Given a parity check matrix $H \in \mathbb{F}_2^{(n-k)\times n}$, a non-zero target vector $y \in \mathbb{F}_2^{n-k}$, and a weight $w \in \mathbb{N}$; The syndrome decoding problem (SDP) defines the problem of finding a vector $x \in \mathbb{F}_2^n$ of weight $wt(x) \leq w$ (if any) such that:*

$$Hx^T = y^T$$

Definition 5 (Minimum Distance Problem [AMCG08]) *Given a parity check matrix $H \in \mathbb{F}_2^{(n-k)\times n}$ and a weight $w \in \mathbb{N}$; The minimum distance problem (MDP) defines the problem of finding a non-zero vector $x \in \mathbb{F}_2^n$ of weight $wt(x) \leq w$ such that:*

$$Hx^T = 0$$

2.2.7 Lattice Problems

A lattice $\mathcal{L} \subseteq \mathbb{R}^n$ is a discrete subgroup of $\mathbb{R}^n$. A basis of $\mathcal{L}$ is a set of linearly independent vectors $B = \{b_1, \dots, b_d\}$ such that:

$$\mathcal{L} = \mathcal{L}(B) = B \cdot \mathbb{Z}^n = \left\{ \sum_{i=1}^{d} x_i b_i : x_i \in \mathbb{Z} \right\}$$

Definition 6 (Shortest Vector Problem) *Given a lattice $\mathcal{L} \subseteq \mathbb{R}^n$ with $\lambda(\mathcal{L})$ denoting its minimal distance, the approximate shortest vector problem (SVP) for $\gamma \geq 1$ defines the problem of finding a non-zero vector $v \in \mathcal{L} \setminus \{0\}$ such that:*

$$||v|| \leq \gamma \cdot \lambda(\mathcal{L})$$

Definition 7 (Short Integer Solution [CLRS10b]) *Given a matrix $A \in \mathbb{Z}^{n\times m}$ and a prime number q, the short integer solution (SIS) defines the problem of finding a vector v in the lattice defined by $\Lambda_q^{\perp} = \{x \in \mathbb{Z}^m : Ax = 0 \bmod q\}$ that has a length $||v|| \leq \beta$ with $\beta < q$.*

2.2.8 Problems in Multivariate Cryptography

We follow the definitions of [PBB13]. A multivariate quadratic system over a finite field $\mathbb{F}$ is a system of m multivariate quadratic equations with N variables and can be written as follows:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij}^{(k)} \cdot x_i \cdot x_j + \sum_{i=1}^{n} p_i^{(k)} \cdot x_i + p_0^{(k)}, \text{ with } k \in [m]$$

Definition 8 (MQ-problem [PBB13]) *Given m multivariate quadratic polynomials $p_1, \dots, p_m$ in N variables over a finite field $\mathbb{F}$, the MQ-problem (MQP) defines the problem of finding a vector $x = (x_1, \dots, x_n) \in \mathbb{F}^n$ such that:*

$$p_1(x), \dots, p_n(x) = 0$$

2.3 Building Blocks

2.3.1 Hash Functions

Definition 9 (Hash Function) *A hash function* $\mathsf{H}(\cdot)$ *takes a message of arbitrary length as an input and maps it to a fixed-sized digest or hash value:*

$$\mathsf{H} : \Sigma^* \rightarrow \Sigma^l, \ \textit{with}\, l \in \mathbb{N}$$

A secure hash function has to fulfill the properties preimage resistance (or one-wayness), second preimage resistance (or weak collision resistance), and (strong) collision resistance. Intuitively, the last property, which implies the weaker ones, requires that it should be hard to find to different strings $m \neq m'$ such that $\mathsf{H}(m) = \mathsf{H}(m')$.

When proving the correctness or security of a scheme under the random oracle model, the random oracle acts as an idealized hash function and is replaced in practice with a concrete implementation of a hash function, which is believed to be secure.

2.3.2 Bilinear Pairings

Definition 10 (Bilinear Pairing [Men09, CHY05]) *A bilinear pairing is a map defined over two cyclic groups* $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, \cdot)$ *of order* q *such that the bilinear pairing* $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ *satisfies the following properties:*

1. *Bilinearity:* $\forall R, S, T \in \mathbb{G}_1 : \hat{e}(R + S, T) = \hat{e}(R, T)\hat{e}(S, T)$.

2. *Non-degeneracy:* $\exists P, Q \in \mathbb{G}_1 : \hat{e}(P, Q) \neq 1$.

3. *Computability:* $\forall P, Q \in \mathbb{G}_1$*:* $\hat{e}(P, Q)$ *can be efficiently computed.*

2.3.3 Public Key Encryption

Definition 11 (Public Key Encryption [HKSS20]) *A public-key encryption scheme is a tuple of three algorithms* $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *over a message space* $M(\lambda)$, *randomness space* $R(\lambda)$, *and ciphertext space* $C(\lambda)$*:*

- $\mathsf{KeyGen}(1^\lambda) \rightarrow (pk, sk)$*: A* PPT *algorithm that takes the security parameter* λ *as an input and outputs a public- and secret key-pair* (pk, sk).

- $\mathsf{Enc}(pk, msg) \rightarrow ct$*: A* PPT *algorithm that takes a public key* pk *and a message* $msg \in M(\lambda)$ *as an input and outputs a ciphertext* ct.

- $\mathsf{Dec}(sk, ct) \rightarrow msg$*: A* DPT *algorithm that takes a secret key* sk *and a ciphertext* $ct \in C(\lambda)$ *as an input and outputs a message* msg.

Besides *completeness*, i.e., that with an honestly generated key-pair, decryption of an honestly generated ciphertext always outputs the original message, PKE schemes need to satisfy *IND-CPA security* and *key-privacy* [HKSS20].

IND-CPA security ensures that no efficient adversary should be able to decide which message was encrypted in a ciphertext when given the used public key without the corresponding secret key.

Lastly, key-privacy ensures that no efficient adversary should be able to decide which public key has been used for the encryption of a given ciphertext.

2.3.4 Trapdoor Permutations

Trapdoor permutations are permutations that can be efficiently computed using a public key but computing its inversion is only feasible with the corresponding secret key.

Definition 12 (Trapdoor Permutation [BS20]) *A family of trapdoor permutations is a tuple of three algorithms* $\mathsf{TP} = (\mathsf{KeyGen}, \mathsf{F}, \mathsf{F}^{-1})$*:*

- $\mathsf{KeyGen}(1^\lambda) \to (pk, sk)$ *A* PPT *algorithm that takes the security parameter* λ *as an input and outputs a public- and secret key-pair* (pk, sk)*.*
- $\mathsf{F}(pk, x) \to y$ *A bijective* DPT *algorithm or permutation that takes a public key and an arbitrary value* x *as input and outputs a value* y *on the same domain.*
- $\mathsf{F}^{-1}(sk, y) \to x$ *The bijective* DPT *inverting algorithm of* F *that inputs a secret key* sk *and a value* y *and outputs* x *s.t. for a given valid key-pair* (pk, sk) *from* Gen *it holds that* $y = \mathsf{F}(pk, x)$*.*

The important property is that without knowledge of the *trapdoor*, i.e., the secret key sk, F forms a one-way permutation. With knowledge of the trapdoor, however, the inversion F^{-1} can be calculated efficiently.

2.3.5 Trapdoor Commitments

Trapdoor commitments allow computing binding commitments using a public key, while with the usage of a corresponding secret key, it is possible to change a commitment afterward.

Definition 13 (Trapdoor Commitment [HS20]) *A trapdoor commitment scheme is a tuple of five algorithms* $\mathsf{TC} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{TOpen}, \mathsf{VerifyOpen})$*:*

- $\mathsf{Setup}(1^\lambda) \to pp$*: A* PPT *algorithm that takes the security parameter* λ *as an input and returns the public parameters* pp*.*

- $\mathsf{KeyGen}(pp) \to (pk, \tau)$*: A* PPT *algorithm that takes as input the public parameters pp and outputs a public key pk and a trapdoor τ.*
- $\mathsf{Com}(pk, msg) \to (c, op)$*: A* PPT *algorithm that takes as inputs a public key pk and a message msg and returns a commitment c on msg and an opening op.*
- $\mathsf{TCom}(pk, \tau) \to (state, c)$*: A* PPT *algorithm that takes a public key pk, a trapdoor τ as an input and outputs a state state and a counterfeit commitment c.*
- $\mathsf{TOpen}(\tau, state, c, msg) \to op$*: A* PPT *algorithm that that takes as input a trapdoor τ, a state state, a commitment c, and a message msg and returns a corresponding opening op.*
- $\mathsf{VerifyOpen}(pk, c, op, msg) \to \{0, 1\}$*: A DPT algorithm that returns* 1 *if the given commitment c opens to the message msg using the auxiliary opening information op under the public key pk and* 0*, otherwise.*

A trapdoor commitment scheme must fulfill the security properties *completeness*, *hiding*, *binding*, and *trapdoor indistinguishability* [HS20]. Completeness requires that an honestly generated commitment and opening always verifies under the committed message. Hiding requires that without the opening, the commitment itself does not reveal anything about the message. Without knowledge of the trapdoor, a commitment should be binding, meaning that it is not possible to change the message of a commitment after it has been created. Lastly, when opening a commitment, it should be indistinguishable whether it was a counterfeit commitment generated with a trapdoor or a regular commitment without any trapdoor.

2.3.6 Accumulators

A (static) cryptographic accumulator ACC [BdM94] provides a compact representation of a finite set $\mathcal{X} = \{x_1, \ldots, x_n\}$ and allows to prove membership of an element to this set using this accumulator, while it should be infeasible to create a valid witness for an non-accumulated element.

Definition 14 (Accumulator [DHS15, MHOY20]) *An accumulator over a domain $\mathcal{D}$ consists of a tuple of four algorithms* $\mathsf{ACC} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Wit}, \mathsf{Verify})$*:*

- $\mathsf{Setup}(1^\lambda) \to pp$*: A* PPT *algorithm that takes as input the security parameter λ and returns the public parameters pp.*
- $\mathsf{Eval}_{pp}(\mathcal{X}) \to acc_{\mathcal{X}}$*: A* DPT *or* PPT *algorithm that takes a set $\mathcal{X} \subseteq \mathcal{D}$ as input and returns the corresponding accumulator $acc_{\mathcal{X}}$.*

- $\mathsf{Wit}_{pp}(acc_{\mathcal{X}}, x) \to \{wit_x, \perp\}$*: A* DPT *or* PPT *algorithm that takes the accumulator $acc_{\mathcal{X}}$ and an element x as input and creates a witness w_x proving that x is accumulated by $acc_{\mathcal{X}}$ or outputs $\perp$ if $x \notin \mathcal{X}$.*

- $\mathsf{Verify}_{pp}(acc_{\mathcal{X}}, wit_x, x) \to \{1, 0\}$*: A* DPT *algorithm that takes the accumulator $acc_{\mathcal{X}}$, a witness wit_x, and an element x as input and outputs 1 if wit_x is a valid witness proving that $x \in \mathcal{X}$ holds and 0 otherwise.*

2.3.7 Verifiable Random Functions

A verifiable random function (VRF), introduced by Micali et al. [MVR99], is a pseudo-random function where the owner of a secret key can create a publicly verifiable proof about the correctness of its evaluation.

Definition 15 (Verifiable Random Function [HKSS20]) *A verifiable random function is a tuple of four algorithms* $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Prove}, \mathsf{Verify})$*:*

- $\mathsf{KeyGen}(1^\lambda) \to (vk, sk)$*: A* PPT *algorithm that takes the security parameter λ as an input and outputs a verification key vk and a secret key sk.*

- $\mathsf{Eval}(sk, x) \to y$*: A* DPT *algorithm that takes a secret key sk and a value $x \in \{0,1\}^{a(\lambda)}$ as an input and outputs a value $y \in \{0,1\}^{b(\lambda)}$. Where $a(\lambda)$ and $b(\lambda)$ are polynomially bounded and efficiently computable functions in λ.*

- $\mathsf{Prove}(sk, x) \to \pi$*: A* PPT *algorithm that takes a secret key sk and an input value x as an input and outputs a proof π.*

- $\mathsf{Verify}(vk, x, y, \pi) \to \{0,1\}$*: A* DPT *algorithm that outputs* 1 *if the proof π proves that the value y is the correct output for the input value x under the verification key vk and* 0 *otherwise.*

Two important properties that a VRF should fulfill are *pseudorandomness* and *key-privacy.* Intuitively, pseudorandomness means that no efficient adversary should be able to distinguish between outputs of a PRF from uniform. Secondly, key-privacy ensures that when having only the output without the proof, no efficient adversary should be able to tell for which verification key the output was computed for.

2.3.8 Somewhere Perfectly Binding Hashing

A somewhere statistically binding hash, introduced by Hubáček and Wichs [HW15], allows committing to a database using a hashing key, which is statistically binding at a hidden position. Backes et al. [BDH$^+$19] modified this primitive to *somewhere perfectly binding hashing with private local opening*, where one can open its commitment at individual positions using secret hashing keys.

Definition 16 (Somewhere Perfectly Binding Hash [HKSS20]) *A somewhere perfectly binding hash with private local opening is a tuple of four algorithms* $\mathsf{SPB} = (\mathsf{Gen}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Verify})$*:*

- $\mathsf{KeyGen}(1^\lambda, n, ind) \rightarrow (hk, shk)$*: A* PPT *algorithm that takes the security parameter* λ*, a maximum database size* n*, and an index* ind *as an input and outputs a hashing key* hk *and a secret hashing key* shk*.*
- $\mathsf{Hash}(hk, db) \rightarrow h$*: A* DPT *algorithm that takes a hashing key* hk *and a database* db *as an input and outputs a digest* h*.*
- $\mathsf{Open}(hk, shk, db, j) \rightarrow wit$*: A* DPT *or* PPT *algorithm that takes a hashing key* hk*, a secret hashing key* shk*, a database, and an index* j *as an input and outputs a witness* wit*.*
- $\mathsf{Verify}(hk, h, j, x, wit) \rightarrow \{0, 1\}$*: A* DPT *algorithm that outputs* 1 *if* wit *witnesses that the preimage of* h *has the value* x *at index* j *under the hashing key* hk*, i.e.,* $db_j = x$*, and* 0 *otherwise.*

A SPB scheme is *somewhere perfectly binding* if it holds that when verification succeeds for an index *ind* and a value x, all valid openings also open to x at the same index *ind*. Furthermore, SPB needs to be *index-hiding*, which ensures that no efficient adversary should be able to determine the index *ind* from a public hashing key *hk*.

2.3.9 Zero-Knowledge Proofs

With a *zero-knowledge proof* (ZKP) a prover P can convince a verifier V that it knows a witness *wit* for an NP statement ϕ satisfying a relation $\mathcal{R}$, that is $(\phi, wit) \in \mathcal{R}$, but without the verifier getting any further knowledge than the validity of the statement.

While zero-knowledge proofs provide unconditional or statistical soundness, i.e., even infinitely powerful adversaries should be unable to create a proof for a wrong statement, *zero-knowledge arguments* (computationally sound proofs) only provide computational soundness, so its soundness-breaking property is only limited to PPT adversaries.

Furthermore, we differentiate between *interactive* and *non-interactive* proof or argument systems, depending whether the prover P needs to communicate with the verifier V in order to convince V over its statement or if P can create the proof/argument on its own.

For instance, a non-interactive zero-knowledge arguments of knowledge (NIZKAoK) scheme can be defined as follows:

Definition 17 (NIZKAoK [GM17, MHOY20]) *A* NIZKAoK *scheme consists of a tuple of four algorithms, namely* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{SimProve})$*:*

- $\mathsf{Setup}(1^\lambda, \mathcal{R}) :\to (crs, \tau)$ *A* PPT *algorithm that takes the security parameter λ and a relation $\mathcal{R}$ as input and returns the common reference string crs and a simulation trapdoor τ.*

- $\mathsf{Prove}(crs, \phi, wit) \to \pi$*: A* PPT *algorithm that takes the common reference string crs, a statement ϕ, and a witness wit as input and returns a corresponding proof π s.t. $(\phi, wit) \in \mathcal{R}$.*

- $\mathsf{Verify}(crs, \phi, \pi) \to \{0, 1\}$*: A* DPT *algorithm that on input the common reference string crs, a statement ϕ and a proof π returns 1, if π proves that there exists a witness wit s.t. $(\phi, wit) \in \mathcal{R}$ holds and 0 otherwise.*

- $\mathsf{SimProve}(crs, \tau, \phi) \to \pi$*: A* PPT *algorithm that inputs the common reference string crs, the simulation trapdoor τ, and a statement ϕ and outputs a simulated proof π.*

A NIZKAoK scheme is secure if it satisfies *correctness*, i.e. an honestly created proof verifies, *knowledge-soundness*, *zero-knowledge*, and *simulation-extractability.* While knowledge-soundness requires that one can always extract a valid witness from a verifying proof, simulation-extractability is a stronger property where this extraction should even hold if the adversary has access to a simulation oracle. Lastly, the zero-knowledge property for NIZKAoK and non-interactive zero-knowledge proofs of knowledge (NIZK) holds if no PPT adversary can distinguish between an honest and simulated proof, created by Prove and SimProve, respectively.

Another variant of zero-knowledge proofs are non-interactive witness-indistinguishable proofs (NIWI) proposed by Feige and Shamir [FS90], which need to fulfill the properties (perfect) completeness, *perfect-soundness*, and *witness-indistinguishability* [HKSS20]. Perfect-soundness states, similarly as knowledge-soundness, that it should be impossible to create a verifying proof for a false statement. Witness-indistinguishability is the weakened alternative to zero-knowledge and requires that, given two valid witnesses for a statement, an efficient adversary should not be able to decide which of them has eventually been used to compute a proof.

For interactive zero-knowledge proofs or arguments of knowledge, one often shows that for every verifier V, there exists a simulator $\mathsf{Sim}_{\mathsf{V}}$, only in possession of the statement ϕ without the witness w, who can reproduce a communication transcript that is indistinguishable from one resulted from an actual communication between an honest verifier and a prover. If this zero-knowledge condition only holds for *honest* verifiers, since it could reveal some information to a cheating verifier, we denote this relaxation as honest verifier zero-knowledge (HVZK).

2.3.10 Shamir's Secret Sharing

In general, a (t, N)-*threshold secret sharing scheme* allows to share a secret S over a group of N parties using N shares S_i such that a subgroup of a certain number t is able to recover the secret using their shares, but any group with fewer than t shares should learn nothing about S.

Shamir's secret sharing scheme [Sha79] is based on polynomial interpolation; to share a secret $S \in GF(q)$ with $q > N$ in a (t, N)-threshold manner, we construct a polynomial of degree $t - 1$:

$$p(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1}$$

with the coefficients set to $a_0 = S$ and to random values for $a_i \leftarrow\$\, GF(q)$ with $0 < i \leq t - 1$. We then distribute the shares S_i for the N parties by evaluating the polynomial:

$$S_1 = p(1), \ldots, S_i = p(i), \ldots, S_N = p(N)$$

Using polynomial interpolation, any group of t parties can reconstruct $p(x)$ with their shares S_i and eventually get the secret by evaluating $S = p(0)$. However, having only knowledge over $t - 1$ shares, for each possible value of the last missing share S', every interpolated polynomial is equally likely to be the correct one; thus, one learns nothing about the actual secret S.

3 Threshold Ring Signature Definitions

3.1 Threshold Ring Signature Definition

A t-out-of-N threshold ring (thring) signature σ on a message msg is created by t distinct signers $\mathbf{SI}$ with their corresponding secret keys $\mathbf{S} = \{sk_i\}_{i \in \mathbf{SI}}$ with respect to a ring $\mathbf{R}$ of size N, which includes, among $N - t$ others, the public keys of the signers. A verifier can then be convinced that at least t parties among the ring $\mathbf{R}$ have created the signature, but without learning any further details about this subgroup.

Definition 18 (Threshold Ring Signature) *A (t, N)-threshold ring signature scheme consists of a tuple of four algorithms* $\mathsf{TRS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$*:*

- $\mathsf{Setup}(1^\lambda) \rightarrow pp$*: A* PPT *algorithm that takes the security parameter λ as an input and outputs the public parameters pp.*

- $\mathsf{KeyGen}(pp) \rightarrow (pk_i, sk_i)$*: A* PPT *algorithm that takes the public parameters as an input and outputs a public and secret key-pair for party i.*

- $\mathsf{Sign}_{pp}(msg, \mathbf{R}, \mathbf{S}) \rightarrow \sigma$*: A possibly interactive* PPT *algorithm that takes as inputs a message* $msg \in \{0,1\}^*$*, a list of public keys* $\mathbf{R} = (pk_i)_{i \in \mathbf{RI}}$ *with* $\mathbf{RI} \subseteq \mathcal{U}$ *and* $|\mathbf{RI}| = N$*, and a set of secret keys* $\mathbf{S} = \{sk_i\}_{i \in \mathbf{SI}}$ *with* $\mathbf{SI} \subseteq \mathbf{RI}$ *and* $|\mathbf{SI}| = t$*, whose corresponding public keys are contained in* $\mathbf{R}$*, and outputs a signature* σ*.*

- $\mathsf{Verify}_{pp}(msg, \mathbf{R}, \sigma) \rightarrow \{0,1\}$*:: A* DPT *algorithm that outputs 1, if the signature* σ *is a correct* (t, N)*-threshold ring signature on msg w.r.t. the ring* $\mathbf{R}$ *and 0 otherwise.*

The signing phase of a TRS scheme is said to be *interactive*, if the t signers have to communicate with each other in order to form the signature. In schemes where the signing process is *non-interactive*, the Sign algorithm is often split into two methods, namely Sign and CombiSign. The first method Sign expects only a single secret key sk_i rather than the set $\mathbf{S}$, so each signer independently creates a signature σ_i using its own secret key sk_i. Finally, any party can combine the t signatures $\{\sigma_i\}_{i \in \mathbf{SI}}$ using CombiSign to a single (t, N)-TRS signature.

3.2 Linkable Threshold Ring Signature Definition

Linkable threshold ring signatures were first introduced by Liu et al. [LWW04] and allow to determine if two signatures are *linked*, meaning that they were signed by the same signer (group) with respect to the same ring.

As in [TWC$^+$05], we denote the linkability of a scheme as *group-oriented* if this criterion only depends on the ring, and as *event-oriented* if it considers the "event," the signature has been created for (in the form of an event-id or dependent on the message). Furthermore, we denote the linkability as *coalition-linkable* if the entire signer group has to be the same and as *individual-linkable* if it suffices that they include a single common signer. While *non-accusatory* linkability only detects whether different signatures are linked, *accusatory* linkability also reveals the identity of the signer of those linked signatures. Lastly, *non-slanderability* or *unframeability* ensures that no adversary can create a signature that seems to be linked to an honest user's signature.

Definition 19 (Linkable Threshold Ring Signature) *A linkable threshold ring signature scheme consists of a tuple of five algorithms* LTRS = (Setup*,* KeyGen*,* Sign*,* Verify*,* Link)*. The first four algorithms are syntactically the same as for unlinkable* TRS*, expect for* Sign*, which might also expect an event-id e for event-oriented schemes. Therefore, the only new algorithm to define is* Link*:*

- $\mathsf{Link}_{pp}(\sigma_1, \sigma_2) \rightarrow \{0,1\}$ *A* DPT *algorithm that outputs 1, if the two given signatures* σ_1, σ_2 *are linked and 0 otherwise. In case of ac-*

cusatory linkable schemes, it also outputs the public key(s) $\{pk_i\}$ of the detected "double-signer(s)".

3.2.1 Traceable Threshold Ring Signature

Traceable threshold ring signatures, introduced by Fujisaki et al. [FS07], are a nonce-based modification of linkable ring signatures, where one always includes a nonce or tag in the signing- and verification algorithm, so two signatures on the same message are only linked if the same signer signed twice using the same nonce. If a signer signs a different message using the same nonce, its identity is revealed.

Therefore, we consider traceable thring signature schemes as event-oriented and accusatory linkable thring signature schemes.

3.3 Threshold Ring Signature Security Definitions

A threshold ring signature scheme is secure, if it fulfills the properties *correctness, unforgeability, and anonymity* [MHOY20]. Note that the terms anonymity, *source-hiding, or signer ambiguous* are synonymous. Before providing the according definitions, we introduce three oracles, which the adversary $\mathcal{A}$ can query:

- $\mathcal{SO}(msg, \mathbf{R}, \mathbf{SI})$: Given a message msg a ring $\mathbf{R}$ of size N, and a signer set $\mathbf{SI}$, the signing oracle $\mathcal{SO}$ returns a correctly generated signature $\sigma \leftarrow Sign_{pp}(msg, \mathbf{R}, \mathbf{S} = \{sk\}_{i \in \mathbf{SI}})$ and updates the set of signing queries $Q_{sign} = Q_{sign} \cup (msg, \mathbf{R})$
- $\mathcal{CO}(pk_i)$: Given a public key pk_i with $i \in \mathcal{U}$, the corruption oracle $\mathcal{CO}$ returns the corresponding secret key sk_i and updates the set of corruption queries $Q_{corr} = Q_{corr} \cup i$ if $i \notin Q_{corr}$.
- $\mathcal{JO}(pk, sk)$: Given a public key pk and a secret key sk, the joining oracle $\mathcal{JO}$ adds this key-pair to the system, provided the public key is new, and returns the new corresponding index $i' \in \mathcal{U}$. Additionally, it updates the set of corruption queries $Q_{corr} = Q_{corr} \cup i'$.

If the adversary is granted access to the joining oracle $\mathcal{JO}$ in the security games of a thring signature scheme, and is thus able of adding potentially maliciously created keys, the scheme achieves *anonymity w.r.t adversarially-chosen keys* as proposed by [BKM06].

Furthermore, if it allows the adversary to participate in signing computations with other honest signers, such that $\mathcal{A}$ could potentially infer some knowledge allowing it to identify them in another signature, the scheme is secure against the presence of *active adversaries* [HS20].

Although some schemes may also satisfy these stronger definitions without any or just little modifications, if the authors did not explicitly consider these scenarios, we assume them as being not as secure in these stronger and more realistic scenarios.

Definition 20 (Correctness [MHOY20]) *A* (t, N)-TRS *scheme is correct, if for all security parameters* $\lambda \in \mathbb{N}$*, all messages* $msg \in \{1, 0\}^*$*, and all ring- and signer sets s.t.* $\mathbf{SI} \subseteq \mathbf{RI} \subseteq \mathcal{U}$*, the following holds:*

$$Pr\left[\begin{array}{l} pp \leftarrow \mathsf{Setup}(1^\lambda), \\ \{(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp)\}_{i \in \mathbf{RI}}, \\ \sigma \leftarrow \mathsf{Sign}_{pp}(msg, \mathbf{R}, \mathbf{S}) : \\ \mathsf{Verify}_{pp}(msg, \mathbf{R}, \sigma) = 1 \end{array}\right] = 1$$

Intuitively, correctness or completeness requires that an honestly generated signature verifies, i.e., the Verify algorithm outputs 1.

Definition 21 (Unforgeability [HS20]) *A* (t, N)-TRS *scheme is existentially unforgeable under a chosen message attack if no* PPT *adversary* $\mathcal{A}$ *has a non-negligible probability of winning the following game* $\mathsf{Unforge}_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$*:*

1. *The challenger* $\mathcal{C}$ *chooses a security parameter* λ *and initializes the public parameters* $pp \leftarrow Setup(1^\lambda)$*.*
2. *The adversary* $\mathcal{A}$ *is given access to the signing oracle* $\mathcal{SO}$ *and the corruption oracle* $\mathcal{CO}$ *and may query them according to any adaptive strategy.*
3. $\mathcal{A}$ *outputs* $(msg^*, \mathbf{R}, \sigma^*)$*, corresponding to a* (t, N)*-threshold ring signature* σ^* *on a message* msg^* *w.r.t. to a ring* $\mathbf{R}$*.*
4. $\mathcal{A}$ *wins if* $\mathbf{R} = (pk_i)_{i \in \mathbf{RI}}$ *with* $\mathbf{RI} \subseteq \mathcal{U} \wedge |\mathbf{RI}| = N$*, the signature* σ^* *is a valid* (t, N)*-threshold ring signature, i.e.,* $\mathsf{Verify}_{pp}(msg^*, \mathbf{R}, \sigma^*) = 1$*,* $(msg^*, \mathbf{R}) \notin Q_{sign}$*, and* $|Q_{corr} \cap \mathbf{RI}| < t$*.*

An adversary $\mathcal{A}$ *is said to* $(\tau, q_s, q_c, \epsilon)$*-break the unforgeability if within running time* τ*,* $\mathcal{A}$ *wins the game* $Unforge_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$ *with probability* ϵ *after* q_s *many* $\mathcal{SO}$ *queries and* q_c *many* $\mathcal{CO}$ *queries have been issued [AMCG08].*

Definition 22 (Computational Anonymity [HS20]) *A* (t, N)-TRS *scheme is computational anonymous if every* PPT *adversary* $\mathcal{A}$ *has the following probability of winning the game* $\mathsf{Anon}_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$*:*

$$Pr\left[\mathcal{A} \textit{ wins } \mathsf{Anon}_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

With $\mathsf{negl}(\lambda)$ *being a negligible function and* $\mathsf{Anon}_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$ *denoting the following game:*

1. *The same as Step 1 in* $\mathsf{Unforge}_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$*.*

2. *The same as Step 2 in* $\mathsf{Unforge}_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$.

3. *The adversary* $\mathcal{A}$ *sends a message* msg^**, a ring* $\mathbf{R} \subseteq \mathcal{U}$*, and two signer-indices sets* $\mathbf{SI}_0^*, \mathbf{SI}_1^* \subseteq \mathbf{RI}$*, where* $|\mathbf{SI}_0^*| = |\mathbf{SI}_1^*| = t$ *to the challenger* $\mathcal{C}$.

4. *The challenger* $\mathcal{C}$ *selects a random bit* $b \leftarrow\$ \{0, 1\}$ *and sends the signature* $\sigma^* \leftarrow \mathsf{Sign}_{pp}(msg^*, \mathbf{R}, S_b^*)$ *with* $\mathbf{S}_b^* = \{sk_i\}_{i \in \mathbf{SI}_b^*}$ *to* $\mathcal{A}$.

5. $\mathcal{A}$ *outputs a bit* b' *and wins if* $b' = b$ *and* $\mathbf{SI}_0^* \cup \mathbf{SI}_1^* \cap Q_{corr} = \emptyset$.
For linkable schemes it can be further required that no unique signer of the two sets $\mathbf{SI}_0^*$*,* $\mathbf{SI}_1^*$ *was included in a query to* $\mathcal{SO}$.

If the adversary $\mathcal{A}$ *also has access to the joining oracle* $\mathcal{JO}$ *in addition to the signing oracle* $\mathcal{SO}$ *and corruption oracle* $\mathcal{CO}$*, the scheme is anonymous w.r.t adversarially-chosen keys.*

Finally, an adversary $\mathcal{A}$ *is said to* $(\tau, q_s, q_c, q_j, \epsilon)$*-break the anonymity if within running time* τ*,* $\mathcal{A}$ *wins the game* $Anon_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$ *with probability* ϵ *after* q_s *many* $\mathcal{SO}$ *queries,* q_c *many* $\mathcal{CO}$ *queries, and* q_j *many* $\mathcal{JO}$ *queries have been issued.*

Definition 23 (Unconditional Anonymity) *A* (t, N)-TRS *scheme is unconditionally anonymous if Definition 22 holds with the modifications that the adversary* $\mathcal{A}$ *is computational unbounded and the requirement in Step 5 of the* $\mathsf{Anon}_{\mathcal{A}}^{\mathsf{TRS}}(\lambda)$ *game,* $\mathbf{SI}_0^* \cup \mathbf{SI}_1^* \cap Q_{corr} = \emptyset$*, is removed.*

Intuitively, anonymity for (t, N)-threshold ring signatures means, that for a given signature σ and the corresponding ring $\mathbf{R}$, an adversary $\mathcal{A}$ cannot determine which subgroup of size t among the N ring members has created the signature.

Computational anonymity is often limited to the fact that $\mathcal{A}$ does not have the secret keys for the possible signer subsets, which it could use to check its guesses, or the anonymity relies on a problem that is assumed to be hard.

Unconditional anonymity is a stronger notion in which a computationally unbounded adversary $\mathcal{A}$, who is therefore also able to compute all secret keys and try out all random values possibly used for the signature, should also have no better chance of identifying the actual signers than randomly guessing.

Note that an adversary does not necessarily need to be computationally unbounded but could also get in possession of all secret keys because, e.g., the underlying public-key cryptography has been broken or a trusted authority responsible for the key-generation has been corrupted.

Another way to define unconditional anonymity is by constructing a PPT simulation $\mathsf{Simulate}_{pp}(msg, \mathbf{R}, \mathbf{S}_{\mathbf{RI}\setminus\{i\}})$, where $\mathbf{S}_{\mathbf{RI}\setminus\{i\}}$ denotes the set of all

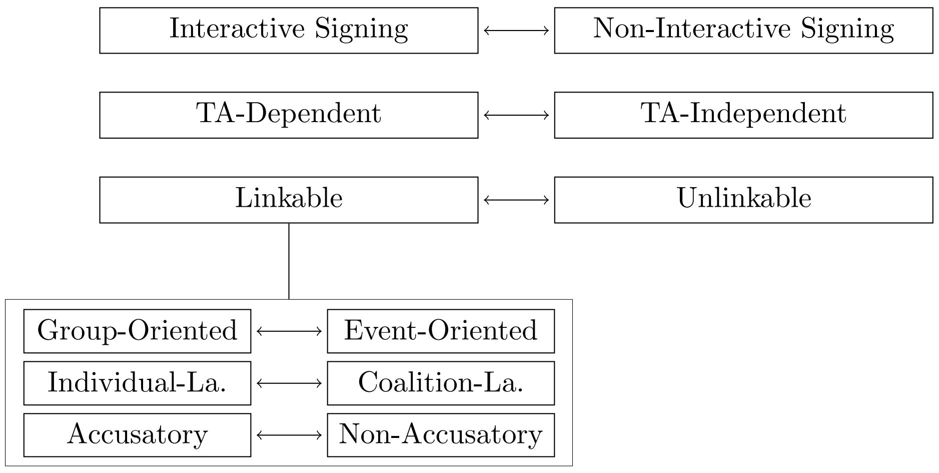

Figure 1: Classification of threshold ring signatures.

ring members' secret keys except the one from party i, that simulates the signing process [DV09, AMCG08]. For an unconditionally anonymous TRS scheme it then should hold that for any ring $\mathbf{R}$, any signer subset $\mathbf{SI}$, and any signer $u \in \mathbf{SI}$, there exists a random sequence such that:

$$\mathsf{Simulate}_{pp}(msg, \mathbf{R}, \mathbf{S}_{\mathbf{RI}\setminus\{u\}}) = \mathsf{Sign}_{pp}(msg, \mathbf{R}, \mathbf{S})$$

Intuitively, it should be possible to create two identical signatures but created by different signer subgroups, e.g., the simulation swaps the role of a previous non-signer with a signer of a given signature.

3.4 Classification

As shown in Figure 1, we classify thring signature schemes along three axes; interactive signing (IS) vs. non-interactive signing (NIS), *TA-dependent* (TAD) vs. *TA-independent* (TAI), and linkable (LA) vs. unlinkable (ULA). Linkable ring signatures are again divided into group-oriented vs. event-oriented, accusatory vs. non-accusatory, and individual-linkable vs. coalition-linkable as proposed by [TWC$^+$05].
IS vs. NIS: The first distinction depends on whether the signers have to interact and cooperate during the signing process or if each signer can create a signature on its own and any third party can combine them later on.
TAD vs. TAI: If some form of a trusted authority or trusted third party is needed in order to create a signature, we call this signing process TA-dependent, otherwise TA-independent. Note that a signing phase can be non-interactive (among the signers), but each signer still has to communicate with a trusted authority.
LA vs. ULA: Last but not least, thring signatures can provide linkability or unlinkability. For linkable signatures, we follow the classification by

[TWC+05] and distinguish between group-oriented schemes, where the linkability property only depends on the signer group, and event-oriented schemes where it further considers the "event" the signature was created for. If it suffices that only a single signer signs twice with respect to the same ring independently of the other co-signers, we denote them as individual-linkable and otherwise as coalition-linkable. Additionally, if linked signatures also reveal the identities of the signers, we denote them as accusatory and otherwise as non-accusatory.

3.4.1 Considerations regarding Anonymity

IS vs. NIS: Regarding anonymity, in schemes with an interactive signing process, the signers have to trust each other because a corrupt signer could betray the other honest signers by, e.g., disclosing their communication, which could prove their cooperation of signing. Of course, this attack-type of deanonymization by corrupt co-signers or active adversaries does not apply to non-interactive schemes.
TAD vs. TAI: For TA-dependent signing protocols, the trusted authority can be considered as a single point of failure; if the trusted authority gets corrupted, in the worst case, no one can create signatures anymore and all secret keys are revealed. Depending on its role and influence in specific schemes, it can be able to revocate the signers' anonymity. While the spontaneity property of ring signatures still holds, i.e., signers can still freely select the ring of public keys, such a revocation property can be seen as a violation of the original idea behind ring signatures by Rivest et al. [RST01].
LA vs. ULA: Linkable thring signatures intentionally restrict the signers' anonymity to detect if the same group of signers signed more than once, which can be useful for applications such as e-voting. However, it is important to emphasize that all linkable schemes are *culpable*, which means that the actual signer group can confirm that it has created a given signature by creating another signature that is linked to the given one. Analogously, a group of non-signers can prove the opposite for themselves.

While this anonymity restricting property does not apply to unlinkable thring signatures, they can still provide *claimability*, meaning that signers have the possibility of proving that they are the creators of a signature or that a party didn't sign [LWW04]. To do so, the signers could reveal the seeds with which they pseudo-randomly generated the non-signer parts of the signature, which proves that they have generated the signature and that the non-signers indeed did not participate in the singing process [RST01]. The crucial difference remains that claimability gives the signers only the possibility of claiming their signature by choice, while for culpable schemes the signers and non-signers can be identified by breaking the scheme's underlying hardness assumption.

Following this fact, all linkable thring signatures can only provide computational anonymity. Because as soon as an adversary computed the secret key(s) of a possible signer or a signer subset for a given signature, depending on if it is individual- or coalition linkable, it can verify its guess by creating a signature on its own and checking if they are linked. Furthermore, we can consider the linkability properties group-oriented, individual-linkable, and accusatory as most anonymity restrictive. Those signatures are already linked if a single signer signs twice, regardless of the other co-signers or the event/message the signature has been created for, and additionally reveals the signer's identity.

4 Construction with Secret Sharing

The main idea of using Shamir's secret sharing for constructing (t, N)-thring signatures is to prove that at least t out of N parties have used their secret key in order to compute their shares for a polynomial interpolated by $N - t$ shares belonging to the non-signers.

We observe two commonly used variants of implementing Shamir's secret sharing for thring signatures: In both variants, a polynomial f is interpolated by $N - t$ random shares for the non-signers (plus one initial point), and afterward, the signers evaluate this polynomial and make use of their secret keys to receive their own valid shares of the signature.

In the first variant, the non-signers' shares are generated using a trapdoor permutation, and after the polynomial that passes through those shares has been interpolated, the signers evaluate it and compute their own shares by inverting their trapdoor permutations with their secret keys.

In the second variant, all shares are randomly sampled beforehand and committed to a hash for the polynomial's initial point. $f(0)$. This initial point and the other $N - t$ shares of the non-signers are used to interpolate the polynomial. Since the output of the random oracle (the hash function) is unpredictable and thus not known in advance, it is ensured that all shares are "fixed," so the signers cannot choose their shares after the interpolation of f. Therefore, the signers need to adjust other values that are connected to these fixed shares using their secret keys and the evaluation of the polynomial.

This construction hides the signers in a way that any polynomial of degree $N + t$ can be interpolated by $N - t + 1$ points, and if all points are randomly chosen independently of the signer group, the polynomial and all resulting shares do not reveal any information about the signers behind the signature. Furthermore, it is obviously not possible to tell which of the N shares were used to interpolate the polynomial or have been obtained afterward by evaluating it and thus belonging to the signers.

4.1 Linkable Variants

There currently does not exist any linkable thring signature scheme that constructs the signature using Shamir's secret sharing to the best of our knowledge. Even though it may be possible to design a linkable scheme with Shamir's secret sharing, this construction type is not reasonable for linkable scenarios. The linkability is mostly used to ensure that indeed t-out-of-N distinct signers have created a signature; with Shamir's secret sharing, however, this criterion can be readily ensured while remaining unlinkable.

4.2 Unlinkable Variants

4.2.1 Based on Trapdoor Permutations

As an exemplary thring signature scheme that uses Shamir's secret sharing in the first variant, we examine the first scheme proposed by Bresson et al. [BSS02] based on RSA, the random oracle model, and the ideal cipher model.

The random permutation under the ideal cipher model is denoted with $E_{k,i}$, with the additional parameter i acting as an index. The trapdoor permutation TP is instantiated with an extended RSA trapdoor permutation, which ensures the efficient combination of unique RSA keys with different moduli N_i.

Then, if a group of t signers $\mathbf{SI}$ want to sign a message msg w.r.t. a ring $\mathbf{R} = (pk_i)_{i \in \mathbf{RI}}$, they proceed as follows:

1. $k \leftarrow \mathsf{H}(msg)$
   Set the symmetric key to the hash of the message msg.

2. $\forall i \in \mathbf{NI} : x_i \leftarrow\!\$ \{0,1\}^l$ and $y_i \leftarrow \mathsf{TP.F}(pk_i, x_i)$
   Sample random values x_i of length l for all non-signers and compute y_i using the trapdoor permutation $\mathsf{TP.F}$ in the forward direction.

3. Compute the polynomial $f \in GF(2^l)$ s.t.
   $deg(f) = N - t,\ f(0) = y_0 = \mathsf{H}(\mathbf{R}),\ \forall i \in \mathbf{NI} : f(i) = E_{k,i}(y_i)$

4. $\forall i \in \mathbf{SI}$: Set $y_i = f(i)$ and compute $x_i \leftarrow \mathsf{TP.F}^{-1}(sk_i, E_{k,i}^{-1}(y_i))$
   The signers evaluate the polynomial to receive y_i and make use of their secret key to invert the $\mathsf{TP.F}$ and compute x_i.

5. Output $\sigma := ((x_i)_{i \in \mathbf{RI}}, f)$

The verification is straightforward: The verifier just recovers k, v, and all y_i values using x_i and $\mathsf{TP.F}$ with pk_i, and finally checks if $deg(f) = N - t$, $f(0) = \mathsf{H}(\mathbf{R})$, and if $f(i) = E_{k,i}(y_i)$ holds for all $i \in \mathbf{RI}$.

In the same manner, the signature is constructed in the code-based scheme by Dallot et al. [DV09]. The trapdoor permutation in their scheme is based on a specific instance of the syndrome decoding problem. With each non-signers' public key, the signers can compute the syndrome of any random word x_i with which they likewise interpolate a polynomial f. Using their secret keys, the signers can decode the syndrome that includes the evaluation of $f(i)$ to receive the corresponding word that weighs less than a public set weight.

Classification Since the t signers have to cooperate with each other in a way that, e.g., a "leader" of the signer group creates f, sends it to the other $t - 1$ signers and receives their x_i values, the signing process of this scheme is interactive (IS). Since no trusted authority is needed in order to create the signature, it is TA-independent (TAI).

Anonymity For all given values x_i, where each value belongs to a ring member $i \in \mathbf{RI}$, the polynomial f passes through the points $(i, E_{k,i}(\mathsf{TP.F}(pk_i, x_i)))$, and through the point $(0, \mathsf{H}(\mathbf{R}))$. Having $N+1$ points in total but the polynomial's degree is $N-t$, we know that $N-t+1$ points have been used to interpolate f and the remaining t points have been obtained through evaluating f. Since all $N-t$ random values y_i of the non-signers and the output of the random oracle for the initial value $y_0 = H(\mathbf{R})$ used for the interpolation of f are independent of the signer group, any other polynomial of the same degree in $GF(2^l)$ could have been obtained likewise. Due to the trapdoor permutation $\mathsf{TP.F}$, when evaluating f to receive $y_i \leftarrow E^{-1}_{k,i}(f(i))$, one has to make use of the corresponding trapdoor sk_i to invert $\mathsf{TP.F}$ and to compute a valid x_i such that $y_i = \mathsf{TP.F}(pk_i, x_i)$ holds.

However, no adversary $\mathcal{A}$ can tell which of the N points have been received through evaluation of the polynomial, instead of being used to interpolate it. That holds even if $\mathcal{A}$ is in possession of all the secret keys, still, all N parties are equally likely to be one of the (non-)signers.

Similarly to [DV09], one can prove the unconditional anonymity by constructing a simulation $\mathsf{Simulate}_{pp}(msg, \mathbf{R}, \mathbf{S}_{\mathbf{R}\setminus\{u\}})$, to which is given the set $\mathbf{S}_{\mathbf{R}\setminus\{u\}}$ containing all ring members' secret keys except the one from party u. It should output the same random sequence as a given signature σ but with having swapped a signer $u \in \mathbf{SI}$ with a non-signer $v \in \mathbf{NI}$. For this scheme, the simulation is constructed as follows:
The key k and the initial value y_0 is set to the same value, and for every $i \in \mathbf{NI} \cup \{u\} \setminus \{v\}$ there are random values such that $x'_i = x_i$, thus it holds that $y'_i = y_i$ and $f' = f$. Then, the x'_i values for all $i \in \mathbf{SI} \setminus \{u\}$ are calculated the same as in Step 4, hence $x'_i = x_i$. The last value $x'_v \leftarrow \mathsf{TP.F}^{-1}(E^{-1}_{k,v}(sk_v, f'(v)))$ is computed by using the secret key of party v. Finally, it holds for the simulated signature $\sigma' = ((x'_i)_{i \in \mathbf{RI}}, f')$ that $\sigma' := \sigma$.

Hence, the t signers remain unconditionally anonymous among the N other parties of the ring.

Efficiency and Signature Size Signing requires $N-t$ forward computations and t inversions of the RSA trapdoor permutation $\mathsf{TP.F}$ and N polynomial evaluations. Verification requires N forward computations of $\mathsf{TP.F}$ and N polynomial evaluations. If the public exponent of RSA e is set to 3, $\mathcal{O}(N-t)$ modular multiplications and $\mathcal{O}(t)$ modular exponentiations are needed for signing and exactly $2N$ modular multiplications for verifying.

Even though the signature size is independent of t, to satisfy unforgeability under a chosen message attack, the authors state that the security parameter l has to be much larger than N such that the signature size is at least $\mathcal{O}(N^2)$.

Regarding the scheme from Dallot et al. [DV09], their signing process requires $N - t$ syndrome computations, N polynomial evaluations and approximately $t(c!)$ syndrome decodings. Verifying a signature requires $N + 1$ polynomial evaluations and N syndrome computations.

Their signature size is in $\mathcal{O}(N)$ and more specifically, the authors state that with the parameters $m = 16$ and $c = 9$, to achieve a security level of $2^{63.3}$, the size is only $579N - 198t$ bits. As for many code-based schemes, a downside are the large public keys, which would require 1.2 MBytes in this case.

4.2.2 Based on Bilinear Pairings

The first identity based thring signature was presented by Chow et al. [CHY05], which relies on the computational Diffie-Hellman problem under the random oracle model and uses bilinear pairings.

In identity based signatures schemes every user already has a public key without having to run a key-gen algorithm before being able to participate in the public-key system. The public key just corresponds to the output of a public function, e.g., a hash function, for some identifying string of a user. When using identity based schemes for ring signatures, the spontaneity property is "improved" when considering real world scenarios, since every signer can freely select public keys of parties that haven't even intentionally joined any public-key system by generating and publishing their public keys. However, this approach requires a trusted authority from which every user has to request their secret key if they want to create a signature.

The scheme operates on two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q as defined for bilinear pairings 10 and uses two hash functions $\mathsf{H} : \{0,1\}^* \to \mathbb{G}_1$ and $\mathsf{H}_0 : \{0,1\}^* \to \mathbb{Z}_q^*$. In $\mathbb{G}_1$, the CDHP needs to be hard but the DDHP should be efficiently solvable, e.g., as on a specific elliptic curve. With a generator point P of $\mathbb{G}_1$, the master secret key of the trusted authority is a random value $sk_{TA} \leftarrow\$ \mathbb{Z}_q^*$ and the public key corresponds to $PK_{TA} = P \cdot sk_{TA}$. A public key of a party with identity ID is of the form $PK_{ID} = \mathsf{H}(ID)$ and the secret key equals $SK_{ID} = \mathsf{H}(ID) \cdot sk_{TA}$.

The signing process for a message msg is of the second variant and works as follows:

1. $\forall i \in \mathbf{NI} : z_i, y_i \leftarrow\$ \mathbb{Z}_q^*$ and compute $U_i \leftarrow z_i P - y_i PK_{TA}$ and $V_i \leftarrow z_i PK_{ID_i}$.

2. $\forall i \in \mathbf{SI} : r_i \leftarrow\$ \mathbb{Z}_q^*$ and compute $U_i \leftarrow r_i P$.

3. Set $y_0 \leftarrow \mathsf{H}_0(\mathbf{R}, t, msg, \bigcup_{k=1}^{N}\{U_k\})$ and interpolate the polynomial $f \in GF(q)$ s.t. $deg(f) = N - t,\ f(0) = y_0,\ \forall i \in \mathbf{NI} : f(i) = y_i$.

4. $\forall i \in \mathbf{SI}$: Set $y_i \leftarrow f(i)$ and compute $V_i \leftarrow r_i PK_{ID_i} + y_i SK_{ID_i}$.
5. Output $\sigma := (\bigcup_{k=1}^{N}\{U_k\}, \sum_{k=1}^{N} V_k, f)$.

To verify a given signature σ, the verifier checks if $deg(f) = N - t$, $f(0) = \mathsf{H}_0(\mathbf{R}, t, msg, \bigcup_{k=1}^{N}\{U_k\})$, and if the following equation holds with the use of a bilinear pairing $\hat{e}$:

$$\prod_{k=1}^{N} \hat{e}(PK_{ID_k}, U_k + f(k)PK_{TA}) = \hat{e}(P, \sum_{k=1}^{N} V_k)$$

Classification The signing process is interactive (IS) and TA-dependent (TAD) since one need to request a secret key from the trusted authority before being able to sign.

Anonymity Similarly to 4.2.1, all non-signers' values y_i used to interpolate the polynomial f are randomly generated and the initial point y_0 is the output of a random oracle, the resulting polynomial f can be considered as chosen randomly from all polynomials of degree $N - t$ over $\mathbb{Z}_q$. Thus, the evaluated values y_i for $i \in \mathbf{SI}$ are random as well and since all z_i, r_i values are also randomly generated and independent from SK_{ID_i}, the distribution of the final signature $\{\bigcup_{k=1}^{N}\{U_k\}, \sum_{k=1}^{N} V_k, f\}$ is uniformly distributed and independent of the signer set for any message msg and ring $\mathbf{R}$. Following this observation, this scheme also provides unconditional anonymity.

Even if an adversary would get in possession of the master secret key sk_{TA} of the trusted authority, and thus can compute every party's secret key, the adversary still had no better chance of determining the signer set than just randomly guessing. The only risk that would remain is that the trusted authority itself is corrupt or gets corrupted and logs all secret key requests; if then a valid signature appears, where $N - t$ parties of the ring have never requested their secret key, the trusted authority can immediately infer the signer group.

Efficiency and Signature Size Signing requires $3N$ group multiplications and N polynomial evaluations. Verification requires $\mathcal{O}(N)$ group multiplications, N polynomial evaluations, and N pairing operations, which are still rather expensive, according to the authors.

Lastly, the signature size is $\mathcal{O}(N)$.

4.2.3 Based on Trapdoor Commitments

Haque and Scafuro [HS20] presented a post-quantum secure thring signature based on trapdoor commitments, which holds under the quantum random oracle model (QROM) and is secure against active adversaries and adversarially-chosen public keys.

Here, the public- and secret key-pair (pk_i, sk_i) of a party $i \in \mathcal{U}$ corresponds to the output of the KeyGen algorithm of a trapdoor commitment scheme TC such that pk_i equals to the public key for TC and the secret key equals to the according trapdoor $sk_i = \tau$.

The main idea of the signing process is of the second variant and works as follows for a message msg over some field $\mathbb{F}$ with respect to a ring $\mathbf{R}$:

1. $\forall i \in \mathbf{NI} : y_i \leftarrow\$ \mathbb{F}$ and $(c_i, op_i) \leftarrow \mathsf{Com}(pk_i, y_i)$.
   Sample random values y_i for all non-signers and create a commitment c_i under pk_i to it.

2. $\forall i \in \mathbf{SI} : (state_i, c_i) \leftarrow \mathsf{TCom}(pk_i, sk_i)$.
   Every signer creates a trapdoor commitment c_i.

3. Set $z_0 \leftarrow H(msg, c_1, \ldots, c_N)$ and interpolate the polynomial f s.t. $deg(f) = N - t,\ f(0) = z_0,\ \forall i \in \mathbf{NI} : f(i) = y_i$.

4. $\forall i \in \mathbf{SI} : y_i = f(i)$ and compute $op_i \leftarrow \mathsf{TOpen}(sk_i, state_i, c_i, y_i)$.
   The signers evaluate f to receive y_i and use their trapdoor sk_i to compute an opening op_i such that c_i opens to y_i.

5. Output $\sigma := ((c_i, y_i, op_i)_{i \in \mathbf{RI}})$

However, this construction cannot be proven to be unforgeable under the quantum oracle model. To reduce the unforgeability to the commitment's binding property, one would classically rewind the adversary and adapt the random oracle so that it produces two openings for the same commitment of a non-signer; but this form of extraction is not guaranteed to work for adversaries having quantum access to the random oracle. This is why the authors applied the Unruh [Unr15] transformation and achieved "on-line extractability" for the unforgeability proof, i.e., all needed outputs are already contained in the signature, so there is no need of rewinding the adversary. Rather than having a single polynomial f, now there are m possible initial points $z_0, \ldots, z_m$ resulting in m different polynomials that should be interpolated by the same single set of commitments $com = (c_i, \ldots, c_N)$. Therefore, the signers have to create m sets of openings, where the non-signer openings are still the same but their own openings are adjusted to the respective polynomial f_j with $1 \leq j \leq m$. All these openings are encrypted using a random one-way permutation G with a random salt r_i to $(g_i, \ldots, g_N)$ for each point z_j. Yet, only one set of openings is revealed using a random oracle with $J = \mathsf{H}_1(msg, com, (g_{i,j})_{i \in \mathbf{RI}, j \in [m]})$, where H_1 maps to $[m]$. This process is repeated K times to increase the probability of extraction, so that we have K "lines" of commitments with each of them having m rows of possible openings (includes y_i, op_i, and r_i), whereas only one row will be revealed. The final signature then consists of $\sigma = (\sigma_k)_{k=1}^{K}$ with $\sigma_k = (\{(c_i^k, y_{i,J_k}^k, op_{i,J_k}^k, r_{i,J_k}^k)\}_{i=1}^{N}, (g_{i,j}^k)_{i \in \mathbf{RI}, j \in [m]})$.

A signature σ on a message msg w.r.t a ring $\mathbf{R}$ is valid if the following checks pass for $1 \leq k \leq K$:

- $\forall i \in \mathbf{RI} : \mathsf{VerifyOpen}(pk_i, c_i^k, op_{i,J_k}^k, y_{i,J_k}^k) = 1$
- For $com^k = (c_i^k)_{i \in \mathbf{RI}}$, $J^k \leftarrow \mathsf{H}(msg, com^k, (g_{i,j})_{i \in \mathbf{RI}, j \in [m]})$, and $z_{J_k}^k \leftarrow \mathsf{H}_1(msg, com^k, J_k)$ it holds $\forall i \in \mathbf{RI} : g_{i,J_i}^k = \mathsf{G}(y_{i,J_k}^k \| op_{i,J_k}^k \| r_{i,J_k}^k)$
- For the polynomial f defined by $(0, z_{J_k}^k)$ and $N - t$ (i, y_{i,J_k}^k) values it holds that $deg(f) \leq N - t$ and $\forall i \in \mathbf{RI} : f(i) = y_{i,J_k}^k$

Classification The signing process is interactive (IS) and TA-independent (TAI).

Anonymity First of all, as in 4.2.1, the polynomial f or rather the given points $(0, z_{J_k}^k)$ and (i, y_{i,J_k}^k) for $i \in \mathbf{RI}$ are randomly distributed over $\mathcal{F}$ independent of the signer group and thus perfectly hide the signers.

Nevertheless, an adversary $\mathcal{A}$ can deanonymize the signers if G is not hiding and thus enables $\mathcal{A}$ to learn about its preimages or if $\mathcal{A}$ can break the trapdoor indistinguishability of the underlying trapdoor commitment scheme TC.

For the first case, note that for $1 \leq j \leq m$ the encryptions $(g_{i,j}^k)$ with $g_{i,j}^k = \mathsf{G}(y_{i,j}^k \| op_{i,j}^k \| r_{i,j}^k)$ always contain the same openings for the non-signers, whereas the signer openings differ so that they open the commitment c_i^k to the correct point $y_{i,j}^k$ for the polynomial $f_{j,k}$ with the constant term of $z_{j,k}$. Hence, if G is not hiding and an adversary $\mathcal{A}$ can eventually infer the preimages of the given encryptions $(g_{i,j}^k)$, it can identify the signers and non-signers by observing which openings differ and which stay the same.

Assuming the permutation G is hiding, the anonymity of this scheme is reduced to the trapdoor indistinguishability. This follows from the fact that all non-signer commitments are created "honestly" without a trapdoor and all signer commitments are trapdoor commitments so that they can adjust their openings after the polynomial interpolation. So, if an adversary is able to distinguish honest and trapdoor commitments, it can deanonymize the signers of any signature.

Additionally, this scheme achieves anonymity against adversarially-chosen public keys and active adversaries by having the signers check the correctness of each step during the signing phase, especially the correctness of each received commitment.

Efficiency and Signature Size Following from the Unruh transformation, the parameters m (how many possible openings the signers have to create) and K (how often they have to repeat the signing process) are statistical security parameters for the unforgeability of this scheme. Creating a signature requires at least $K(N - t)$ honest- and Kt trapdoor commitment creations, KmN polynomial interpolations and t trapdoor opening computations. Verification requires KN commitment verifications and KN^2 polynomial evaluations.

The signature size amounts to $\mathcal{O}(KmN)$

5 Construction with Ring Hashing

For this type of construction, the signature is created by "hashing along the ring," where the unpredictability property of the random oracle ensures that the ring is only possible to construct it in one direction. Therefore, the signer is required to compute a hash in advance and to use its secret key to adapt its share of the signature, which eventually closes the ring. However, when considering a valid signature, each ring member's share of the signature equally leads to the fulfillment of the ring equations and it is impossible to determine at which position one closed the ring.

Regarding the threshold scenario, the extension differs from linkable to unlinkable schemes. In the case of linkable schemes, every signer just creates a standard ring signature, such that the final thring signature consists of the combination of these t 1-out-of-N signatures, and if all appear to be unlinked, one can be convinced that they are indeed from t different signers. However, other techniques are necessary for unlinkable thring signatures to guarantee that the number of signers is above the specific threshold, such as using fair partitions or by relying on a trusted authority.

Since this construction type relies on the random oracle, all following schemes also require the random oracle model for their security proofs.

5.1 Linkable Variants

5.1.1 Based on the DLP

The first linkable (threshold) ring signature was presented by Liu et al. [LWW04] in 2004, which is based on the DLP and constructs the signature using ring hashing.

Given the cyclic group G of order q defined by the generator g, it holds for a public- and secret key-pair (pk_i, sk_i) of party $i \in \mathcal{U}$ that $pk_i = g^{sk_i}$. Additionally, the scheme utilizes two hash functions $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{Z}_q$ and $\mathsf{H}_2 : \{0,1\}^* \to G$.

The 1-out-of-N signature of round $k \in [t]$ on a message msg w.r.t to a ring $\mathbf{R}$ is generated as follows by the signer $i_s \in \mathbf{SI} = \{i_s\}$:

1. Compute $h = \mathsf{H}_2(\mathbf{R})$ and $\tilde{y} \leftarrow h^{sk_{i_s}}$.
2. Set $r \leftarrow\$\, \mathbb{Z}_q$ and compute $c_{i_s+1} \leftarrow \mathsf{H}_1(\mathbf{R}, \tilde{y}, msg, g^r, h^r)$.
3. $\forall i \in \mathbf{NI} : s_i \leftarrow\$\, \mathbb{Z}_q$ and compute $c_{i+1} \leftarrow \mathsf{H}_1(\mathbf{R}, \tilde{y}, msg, g^{s_i} pk_i^{c_i}, h^{s_i} \tilde{y}^{c_i})$.

4. Compute $s_{i_s} \leftarrow r - sk_{i_s} c_{i_s} \bmod q$.
5. Output $\sigma_k := (c_1, (s_i)_{i \in \mathbf{RI}}, \tilde{y})$

To verify the signature, the verifier recalculates $h = \mathsf{H}_2(\mathbf{R})$ and $\forall i \in \mathbf{RI} : z_i' \leftarrow g^{s_i} pk_i^{c_i}$, $z_i'' \leftarrow h^{s_i} \tilde{y}^{c_i}$ and $c_{i+1} \leftarrow \mathsf{H}_1(\mathbf{R}, \tilde{y}, msg, z_i', z_i'')$ if $i \neq N$. Then, it checks if the final ring equation $c_1 = \mathsf{H}_1(\mathbf{R}, \tilde{y}, msg, z_N', z_N'')$ holds.

The Link algorithm checks whether two signatures σ and σ' are linked if they are valid and the so-called *linkability tags* are equal $\tilde{y} = \tilde{y}'$.

Last but not least, the extension to a t-out-of-N signature corresponds to the concatenation of all t 1-out-of-N signatures: $\sigma := \{\sigma_k\}_{\in[1,t]} = \{\sigma_1, \ldots \sigma_t\}$, where all different pairs σ_k, $\sigma_{k'}$ need to be unlinked.

Classification Since every signer can independently create a signature and combine it with the other signers' signatures, the signing process is non-interactive (NIS) and TA-independent (TAI). Additionally, it is individual linkability is event-oriented and non-accusatory.

Anonymity We observe that every ring member's value s_i is a valid share of the resulted signature since it contributes to the fulfillment of the ring equations. These equations are, as the name suggests, cyclic, in a way that for every $i \in \mathbf{RI}$ the digest c_i includes the previous c_{i-1} and s_{i-1} as input for the hash function. Since we assume that it is impossible to predict the output of the hash function under the random oracle, we know that a ring member has initially generated a digest, computed the remaining ones afterward, and made use of its secret key to close the ring. This ring closure happens in Step 4 with the computation of s_{i_s} using the signer's secret key. However, its secret key is hidden through the random value r and the random digest c_{i_s} such that s_{i_s} is also randomly distributed over $\mathbb{Z}_q$ as all other s_i values. This results to $t \cdot q^N$ variations of the final threshold signature σ with the same probability regardless of the signer group.

Hence, when excluding the $\tilde{y}$ value (and thus h) created in Step 1, the scheme would provide unconditional anonymity. Yet, the addition of $\tilde{y}$ (the linkability tag) ensures linkability and reduces the anonymity to the DDHP, and, as for all linkable schemes, to the underlying assumption of its public-key cryptography, which is the hardness of the DLP in this case.

For a given signature σ, one can consider the public key of a ring member pk_i, the h value, and $\tilde{y}$ as a triple $(pk_i, h, \tilde{y}) = (g^{sk_i}, g^{\alpha}, g^{\alpha sk_\pi})$. Now, if an adversary $\mathcal{A}$ is able to break the DDHP, it can decide between $(g^{sk_\pi}, g^{\alpha}, g^{\alpha sk_\pi})$ and $(g^{sk_i}, g^{\alpha}, g^{\alpha sk_\pi})$ with $i \neq \pi$, and $\mathcal{A}$ is therefore also able to determine the signer of any valid signature.

Efficiency and Signature Size Creating a t-out-of-N signature requires

$t(4N-2)$ modular exponentiations and $t(2N-1)$ modular multiplications. Verification requires $4tN$ modular exponentiations and $2tN$ modular multiplications.

The signature size results to $\mathcal{O}(tN)$.

5.2 Unlinkable Variants

5.2.1 Based on Trapdoor Permutations

The second thring signature scheme proposed by Bresson et al. [BSS02] is a modification of the initial ring signature scheme by Rivest et al. [RST01], where the permutation under ideal cipher assumption is replaced with a hash function under the random oracle model.

The extension to threshold signatures is achieved through fair partitions. In this scenario, a partition $\pi = (\pi_1, \ldots, \pi_t)$ of the ring indices $\mathbf{RI}$ is a fair partition for the signer set $\mathbf{SI}$ with $|\mathbf{SI}| = t$ if every signer $i \in \mathbf{SI}$ belongs to a different subset π_j, which we call a *sub-ring.*

As in the first scheme by Bresson et al., discussed in 4.2.1, each user $i \in \mathcal{U}$ has a RSA key-pair (pk_i, sk_i) that is used for the extended RSA trapdoor permutation scheme TP.

The so-called *combining function* looks as follows:

$$C_{v,msg}(\gamma, y_1, \ldots, y_N) = \mathsf{H}(msg, y_N \oplus \mathsf{H}(msg, y_{N-1} \oplus \ldots \mathsf{H}(msg, \gamma \oplus y_1 \oplus v))$$

With a hash function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^l$ and each y_i relating to $y_i = \mathsf{TP.F}(pk_i, x_i)$ for some x_i value, and γ figuring as a "*gap value.*" The resulting ring equation is verified if $C_{v,msg}(\cdot) = v$ holds.

To close such a ring, the signer $i_s \in \mathbf{SI}$ of a sub-ring extracts its $x_{i_s} = \mathsf{TP.F}^{-1}(sk_i, y_{i_s})$ value with $y_{i_s} = C^{-1}_{i_s,\gamma,msg}(v_s, (y_i)_{i \in \mathbf{RI}, i \neq i_s})$, which works as follows for a given seed v_s and random y_i values for all non-signers:

1. Compute $v_{i_s+1} \leftarrow \mathsf{H}(msg, v_s)$.
2. Compute $\forall i \in \mathbf{NI} : v_{i+1} \leftarrow \mathsf{H}(msg, v_i \oplus y_i)$.
3. Set $y_{i_s} \leftarrow v_s \oplus v_{i_s}$.
   Which closes the ring: $v_{i_s+1} = \mathsf{H}(msg, v_{i_s} \oplus y_{i_s}) = \mathsf{H}(msg, v_s)$.

Essentially, the signing process then operates on a set of partitions $\Pi = (\pi^1, \ldots, \pi^p)$ with $p = 2^t \log N$ such that for every subset of t ringer members there exists a fair partition π for it in Π. The fair partition π^s, referred to as the signer partition because there is a signer in every sub-ring, enables the signers to close a *super-ring* formed over all partitions using the gap-values.

In more detail, the signing algorithm, which includes another hash function $\mathsf{H}_1 : \{0,1\}^* \to \{0,1\}^{tl}$, works as follows with i iterating over the ring members, j over the sub-rings, and k over the partitions:

1. $\forall k \in [p]: \ \forall j \in [t]: \ v_j^k \leftarrow\!\$\, \{0,1\}^l$.
   Sample random seeds of length l for every sub-ring of each partition.
2. For all non-signer partitions $k = 1, \dots, p$, $k \neq s$:
   (a) $\forall i \in \mathbf{RI} : x_i^k \leftarrow\!\$\, \{0,1\}^l$ and $y_i^k \leftarrow \mathsf{TP.F}(pk_i, x_i^k)$
   Sample random x_i^k values and compute y_i^k using $\mathsf{TP.F}$.
   (b) $\forall j \in [t]: \ z_j^k \leftarrow C_{v_j^k, msg}(0, (y_i^k)_{i \in \pi_j^k(\mathbf{RI})})$ and $\gamma_j^k \leftarrow v_j^k \oplus z_j^k$
   Simulate the sub-rings of the non-signer partition π^k.
3. Set $v'_s \leftarrow\!\$\, \{0,1\}^{t \cdot l}$ and $u_{s+1} \leftarrow \mathsf{H}_1(v'_s)$, and compute a super-ring using the gap-values: $\forall k \in [p], k \neq s : u_{k+1} = \mathsf{H}_1(u_k \oplus (\gamma_1^k \| \dots \| \gamma_t^k))$.
4. Set $(\gamma_1^s \| \dots \| \gamma_t^s) \leftarrow u_s \oplus v'_s$
   Compute the gap-values for the sub-rings of the signer partition π^s.
5. For the signer partition π^s:
   (a) $\forall i \in \mathbf{NI} : x_i^s \leftarrow\!\$\, \{0,1\}^l$ and $y_i^s \leftarrow \mathsf{TP.F}(pk_i, x_i^s)$.
   (b) $\forall i \in$ **SI**: Compute $y_i^s \leftarrow C^{-1}_{i, \gamma_j^s, msg}(v_j^s, (y_i^s)_{i \in \pi_j^s(\mathbf{RI})})$ with j s.t. $i \in \pi_j^s$ and $x_i^s \leftarrow \mathsf{TP.F}^{-1}(sk_i, y_i^s)$, and update $v_j^s \leftarrow v_j^s \oplus \gamma_j^s$ [3].
6. Set $\nu \leftarrow\!\$\, [p]$ and output $\sigma := \Big(\nu, u_\nu, \big(x_1^k, \dots, x_N^k, \, v_1^k, \dots v_t^k\big)_{i \in [p]}\Big)$

Given a signature σ, a verifier needs to recalculate all y_i^k values and all gap-values z_j^k as in Step 2b. Finally, σ is valid if the super-ring equation formed out of those gap-values validates:

$$u_\nu = \mathsf{H}_1(\gamma_1^{\nu-1} \| \dots \| \gamma_t^{\nu-1} \oplus \mathsf{H}_1(\dots \mathsf{H}_1(\gamma_1^\nu \| \dots \| \gamma_t^\nu \oplus u_\nu) \dots))$$

Classification The scheme has an interactive signing phase (IS) and is independent of a trusted authority (TAI).

Anonymity First of all, we observe that similarly to [RST01], in each signer's sub-ring, all $N-1$ non-signer values x_i are randomly sampled out of the set $\{0,1\}^l$. Only at the closing point i_s, corresponding to the signer of this sub-ring, the last x_{i_s} is uniquely determined by the inversion of the combining function. Following this fact, for every signer ring π_j^s and for any message msg, random seed v_j^s, and gap value γ_j^s, the ring equation $C_{v,msg}(\cdot) = v$ has $(2^l)^{(N-1)}$ solutions all of which can be chosen independently of the signer of this ring. Therefore, the signer remains unconditionally anonymous in its sub-ring.

[3] In the paper's description, the authors most likely forgot to update the signer seeds, which is necessary for the super-ring equation to hold.

While only a single fair partition for the signers $\mathbf{SI}$ would reduce their anonymity to its defined sub-rings and their possible combinations of t signers, the set Π of p fair partitions provides complete anonymity. This is due to the fact that every possible subset of t ring members has a fair partition in Π and thus can equally likely be the signer group who has created a given signature.

Efficiency and Signature Size Signing and verifying requires $\mathcal{O}(2^t N \log N)$ forward computations of the RSA trapdoor permutation $\mathsf{TP.F}$, while additional t computations of its inversion $\mathsf{TP.F}^{-1}(\cdot)$ are necessary for creating a signature. In both cases, modular exponentiations are the computationally most expensive operations.

The signature size equals $\mathcal{O}(2^t N \log N)$ due to the fair partitions; if one would just list all subgroups with t ring members, the size would result to $\binom{N}{t} = \mathcal{O}(N^t)$.

5.2.2 Based on Short-Time Keys

Okamoto et al. [OTYO18] presented a "flexible" unlinkable thring signature scheme, where a t-out-of-N signature can be updated to a $(t+\alpha)$-out-of-N signature without having all the previous signers to sign again.

The public and secret key-pair of a user $i \in \mathcal{U}$ corresponds to $pk_i = (\langle g_i \rangle, g_i, p_i, q_i, y_i, \mathsf{H}_i, \bar{\mathsf{H}}_i)$ and $sk_i = x_i$ with the following properties: $\langle g_i \rangle$ is a prime subgroup of $\mathbb{Z}_{p_i}^*$ of order q_i, $\mathsf{H}_i : \{0,1\}^* \to \mathbb{Z}_{q_i}$, $\bar{\mathsf{H}}_i : \{0,1\}^* \to \langle g_i \rangle$ are random hash functions, and it holds for the random value $x_i \in \mathbb{Z}_{q_i}$ that $y_i = g_i^{x_i} \bmod p_i$.

A t-out-of-N signature consists of t 1-out-of-N signatures and to ensure that they were all signed by distinct signers while providing unlinkability, the authors make use of *short-term keys*. On request by the signers, the short-time key-pairs are generated by a dealer with respect to a message msg and a ring $\mathbf{R}$ as follows: Set $t_i \leftarrow\!\$\, \mathbb{Z}_{q_i}$, $w_i \leftarrow\!\$\, \{0,1\}^\lambda$ s.t. w_i is unique for all ring members, and compute $z_i \leftarrow g_i^{t_i} \cdot (\bar{\mathsf{H}}_i(msg \| w_i \| \mathbf{R}))^{-1} \bmod p_i$. Then, the short-time public key corresponds to z_i and the secret key to t_i, whereas the tuple (t_i, w_i) is encrypted and sent to the signer i. The dealer additionally sends the list $\mathbf{z}_{msg,\mathbf{R}} = (z_1, z_2, \ldots, z_N)$ to the signers and saves the history $(msg, \mathbf{R}, \{t_i\}_{i \in \mathbf{RI}}, \{w_i\}_{i \in \mathbf{RI}})$.

After the communication with the dealer, each signer i_s computes its 1-out-of-N signature in a round $k \in [t]$. This signing process is similar to the one from [LWW04] discussed in 5.1.1 and works as shown:

1. Set $w \leftarrow w_i$, $r_{i_s} \leftarrow\!\$\, \mathbb{Z}_{q_{i_s}}$, and compute $a_{i_s} \leftarrow g_{i_s}^{r_{i_s}} \bmod p_{i_s}$ and $c_{i_s+1} \leftarrow \mathsf{H}_{i_s+1}(msg \| \mathbf{R} \| \mathbf{z}_{msg,\mathbf{R}} \| a_{i_s})$.
2. $\forall i \in \mathbf{RI} \setminus \{i_s\}$: $s_i \leftarrow\!\$\, \mathbb{Z}_{q_i}$, compute $a_i \leftarrow z_i \cdot \bar{\mathsf{H}}_i(msg \| w \| \mathbf{R}) g^{s_i} y_i^{c_i} \bmod p_i$ and $c_{i+1} \leftarrow \mathsf{H}_{i+1}(msg \| w \| \mathbf{R} \| \mathbf{z}_{msg,\mathbf{R}} \| a_i)$.

3. Compute $s_{i_s} \leftarrow r_{i_s} - t_{i_s} - x_{i_s} c_{i_s} \bmod q_{i_s}$.
4. Output $\sigma_k := (c_1, (s_i)_{i \in \mathbf{RI}}, w)$ and $\mathbf{z}_{msg,\mathbf{R}}$.

The t-out-of-N signature then results to $\sigma := (\{\sigma_k\}_{k \in [t]}, \mathbf{z}_{msg,\mathbf{R}})$. If α previous non-signers want to extend it to a $(t+\alpha)$-out-of-N signature, they first need to request their short-time keys or rather (t_i, w_i) from the dealer for this specific ring $\mathbf{R}$ and message msg, before they can create the signatures.

To verify a signature σ, the verifier needs to check if all w values of the 1-out-of-N signatures σ_k are unique and if they all are valid by proceeding as follows: For all $i \in \mathbf{RI}$ compute $a_i \leftarrow z_i \cdot \bar{\mathsf{H}}_i(msg\|w\|\mathbf{R}) g^{s_i} y_i^{c_i} \bmod p_i$ and $c_{i+1} \leftarrow \mathsf{H}_{i+1}(msg\|w\|\mathbf{R}\|\mathbf{z}_{msg,\mathbf{R}}\|a_i)$, if $i \neq N$. Finally, the signature is valid if $c_1 = \mathsf{H}_1(msg\|w\|\mathbf{R}\|\mathbf{z}_{msg,\mathbf{R}}\|a_N)$ holds.

Classification The signature generation is non-interactive (NIS).
Although the authors state that the requirement of their dealer is not different from the cooperation among the signers for schemes with an interactive signing phase, we still classify this signing process as TA-dependent (TAD). We see it this way because no party is able to create a signature without having to contact the dealer beforehand, who is in possession of all short-time key-pairs of all created signatures. And when arguing that we could consider the dealer as the cooperation among the signers or as a leader of the signers, their scheme would then turn into a scheme with an interactive signing phase. Also, new signers would then have first to contact the previous signers to extend the existing signature. Therefore, we consider, as do Haque et al. [HKSS20], that this scheme requires a fully trusted authority.

Although signers cannot create another valid signature on the same message with respect to the same ring (depending on the implementation of the dealer), we agree with the authors that their scheme is still unlinkable. This is because the short-time keys, which prevent signers from creating a duplicate signature, are completely independent of a ring member's actual key-pair.

Anonymity This scheme provides unconditional anonymity under the short-time secret keys $\{t_i\}_{i \in \mathbf{RI}}$ of the ring members. Since if an adversary $\mathcal{A}$ gets in possession of the short-time secret keys $\{t_i\}_{i \in \mathbf{RI}}$ for a given signature σ, e.g., by corrupting the dealer, it can tell the signers and non-signers apart. Given a single short-time secret key t', $\mathcal{A}$ can infer whether a ring member $i \in \mathbf{RI}$ is the signer of a 1-out-of-N signature σ_k by checking if $g^{t'} = z_i \cdot \bar{\mathsf{H}}_i(msg\|w\|\mathbf{R})$ holds.

However, without knowledge of the short-time secret keys, this previous equation has two unknown variables, t' and w, because the adversary further does not know to which ring member w belongs. Also notice that the $\{w_i\}_{i \in \mathbf{RI}}$ values have $\binom{2^\lambda - 1}{N} \cdot N!$ variations with the same probability, making it impossible for $\mathcal{A}$ to infer the other $N - t$ unknown w_i values. Consequently, under the DLP assumption, each signer can prove that it was the creator of a signature, while non-signers can prove the opposite for themselves when they have received their tuple (t_i, w_i) from the dealer for the signature in question.

Besides that, the signers are perfectly hidden against any computational unbounded adversary. First of all, through to the random short-time secret keys $t_i \in \mathbb{Z}_{q_i}^*$, the set of short-time public keys $\{z_i\}_{i \in \mathbf{RI}}$ has $\prod_{i \in [n]} q_i$ variations all of which could also have been chosen with equal probability. During the signing process, the signer of σ_k chooses random $s_i \in \mathbb{Z}_{q_i}^*$ values for each non-signer in Step 2, and its own s_{i_s} value is likewise randomly distributed over $\mathbb{Z}_{q_i}^*$ through r_{i_s}, t_{i_s}, and the output of the random oracle c_{i_s}. In conclusion, considering the values $(c_1, (s_i)_{i \in \mathbf{RI}})$ of each signature σ_k, this results to $t \cdot \prod_{i \in [N]} q_i$ variations for the final threshold signature σ, which holds independently of the signer group.

Efficiency and Signature Size After having received the short-time keys, the creation of a t-out-of-N signature requires $t(2N - 1)$ modular exponentiations and $t(3N - 2)$ modular multiplications. The verification requires $2tN$ modular exponentiations and $3tN$ modular multiplications.

The signature size results to $\mathcal{O}(tN)$.

5.2.3 Based on Message Block Sharing

Built on the quantum secure signature scheme based on the shortest vector problem (SVP) from Lyubashevsky [Lyu08], Melchor et al. [MBB+13] proposed a ring signature scheme, which in turn was extended to the threshold setting by Chen et al. in 2019 [CHGL19].

The scheme mostly operates on the quotient ring $\mathcal{D} = \mathbb{Z}_p[x]/\langle x^k + 1 \rangle$, where $x^k + 1$ is irreducible, k is a power of two and $k > \lambda$, and p is a prime s.t. $p = 3 \bmod 9$. Its elements $a \in \mathcal{D}$ are polynomials of degree $k - 1$ and $\hat{a} \in \mathcal{D}^m$ are vectors of m polynomials.

Furthermore, it makes use of a family of collision-resistant hash functions $\mathcal{H} : \mathcal{D}_\times^m \to \mathcal{D}$, which is defined as follows for an integer m and $\mathcal{D}_\times \subseteq \mathcal{D}$: $\mathcal{H} = \{h_{\hat{a}} : \hat{a} \in \mathcal{D}^m\}$ such that for any vector $\hat{z} \in \mathcal{D}_\times^m$ it holds that $h_{\hat{a}}(\hat{z}) = \hat{a}\hat{z} = \sum a_i z_i$. For any $\hat{y}, \hat{z} \in \mathcal{D}^m$ and $c \in \mathcal{D}$ these hash functions satisfy $h(\hat{y} + \hat{z}) = h(\hat{y}) + h(\hat{z})$ and $h(\hat{y}c) = h(\hat{y})c$. Lyubashevsky showed that when $\mathcal{D}_\times$ is restricted to a set of small norm polynomials, the collision problem for hash functions in $\mathcal{H}$ is as hard as solving the SVP in the worst case over lattices corresponding to ideals in $\mathcal{D}$.

The extension from Melchor's ring signature to the threshold setting is achieved by applying a message block sharing technique. Here, signers can't flexibly sign any self chosen message, but a trusted authority first has to process a message msg and distribute it to all N parties so that only subgroups of order t can reassemble the original message. The known hash of the message is denoted by $r \leftarrow \mathsf{H}(msg)$. Each parties' share of the padded and permuted message is denoted by Γ_i for $i \in \mathbf{RI}$ and such a set of message blocks can be combined to get $msg[i] \leftarrow msg_{i_1} \| \ldots \| msg_{i_k}$.

For a publicly known element $C \leftarrow\!\!\$\, \mathcal{D}$, the public key of a party $i \in \mathcal{U}$ corresponds to a hash function $pk_i = h_i$ in $\mathcal{H}$ defined by a self-chosen $\hat{a}_i$ and the secret key equals $sk_i = \hat{s}_i$ such that $h_i(\hat{s}_i) = C$.

Given $\mathcal{D}_y, \mathcal{D}_z, \mathcal{D}_{s,c} \subseteq \mathcal{D}$ and another hash function $\mathsf{H} : \{0,1\}^* \to \mathcal{D}_{s,c}$ the signing process then works as follows:

1. $\forall i \in \mathbf{NI} : \hat{y}_i \leftarrow\!\!\$\, \mathcal{D}_z^m$ and set $\hat{z}_i \leftarrow \hat{y}_i$.
2. $\forall i \in \mathbf{SI} : \hat{y}_i \leftarrow\!\!\$\, \mathcal{D}_y^m$ and compute $\Psi \leftarrow \sum_{i \in \mathbf{RI}} h_i(\hat{y}_i)$ and $y' \leftarrow \mathsf{H}(h_1(\hat{y}_1), \ldots, h_N(\hat{y}_N), \Psi)$.
3. $\forall i \in \mathbf{SI}$: Compute $e_i \leftarrow \mathsf{H}(\Psi, msg[i], y', r)$ and set $\hat{z}_i \leftarrow \hat{s}_i e_i + \hat{y}_i$ if $\hat{z}_i \notin \mathcal{D}_z^m$, go to Step 1.
4. Output $\sigma := \{(\hat{z}_i)_{i \in \mathbf{RI}}, (e_i)_{i \in \mathbf{SI}}, y_i\}$.

Given the signature σ and all message block sets from the signers $\{\Gamma_i\}_{i \in [t]}$, a signature is verified if $r = \mathsf{H}(msg)$ holds for the recombined message and if the following ring equation holds:

$$\forall i \in [t] : e_i = \mathsf{H}(\sum_{j \in \mathbf{RI}} h_j(\hat{z}_j) - C \sum_{i \in [t]} e_i, msg[i], y', r)$$

Classification The signing phase is interactive (IS) and TA-dependent (TAD) since it requires a trusted authority, which distributes padded and permuted message blocks of messages that can be signed.

Anonymity Essentially, the subtraction of $C \sum_{i \in [t]} e_i$ in the ring equation ensures that for it to be satisfied t parties must have used their secret keys s_i in order to to adapt their $\hat{z}_i$ vector. At first, all random $h_i(\hat{y}_i)$ values for each ring member are included in the sum Ψ and committed in the hash for e_i. Since $\mathsf{H}(\cdot)$ is modeled as a random oracle, an equation of the form $e_i = \mathsf{H}(\Psi - Ce_i)$ is not directly solvable. Hence, a digest e_i has to be computed beforehand and, in this case, needs to be combined with a matching secret key so that the subtraction of Ce_i is eliminated. However, for each e_i it is unknown at which index $j \in \mathbf{RI}$ the given $\hat{z}_j$ vector evaluates to $h_j(\hat{z}_j) = Ce_i$, which consequently corresponds to a signer index. This relies on the fact that all $\hat{z}_j$ vectors are random elements out of the same domain $\mathcal{D}_z^m$. Even though the signers' $\hat{z}_j$ vectors aren't randomly sampled but computed as $\hat{z}_j \leftarrow \hat{s}_i e_i + \hat{y}_i$, the secret key $\hat{s}_i$ is hidden by multiplication with the hash digest e_i and by the addition of the random vector $\hat{y}_i$.

Lyubashevsky further showed that for any hash function h of $\mathcal{H}$ and two secret keys $\hat{s}, \hat{s}'$ s.t. $h(\hat{s}) = h(\hat{s}')$ the statistical difference between the signatures generated with each of them is negligible. Analogously, this applies not only to the ring signature scheme by Melchor et al. but also to Chen's et al. threshold extension.

All in all, this scheme provides unconditional anonymity even the trusted authority has no influence on the signer's anonymity because it only distributes messages that can be signed.

Efficiency and Signature Size Creating a signature requires at least N inner products and t scalar multiplications of m dimensional polynomial vectors of degree $k-1$ in the ring $\mathcal{D}$. Verification requires exactly N inner products of those vectors and t polynomial multiplications in $\mathcal{D}$.

The signature size is independent of t and results to $\mathcal{O}(N)$.

6 Construction with Zero-Knowledge Proofs

Schemes that fall under this category construct the thring signature by performing one or multiple zero-knowledge proofs that witness the signature's correctness, while the zero-knowledge property ensures anonymity for the signers. Given such a signature, a verifier can be convinced that it is valid if the scheme's specific zero-knowledge proof verifies, which should only be possible for honestly created signatures due to the soundness of the proof.

A common approach to obtain a thring signature scheme for this construction type is to apply the Fiat-Shamir transform on an HVZK threshold identification protocol.

6.1 Linkable Variants

6.1.1 Based on RSA

Tsang et al. [TWC$^+$05] proposed a linkable thring signature based on the strong RSA assumption, the DDHP, and the random oracle, which uses two transformed zero-knowledge proofs for its construction. A signature is constructed by applying the Fiat-Shamir transformation on two three-move interactive honest-verifier zero-knowledge proofs of knowledge (HVZK) to non-interactive *signatures based on proofs of knowledge* (SPK).

A key-pair of user $i \in \mathcal{U}$ corresponds to $pk_i = (l_i, N_i, g_i, y_i, \mathsf{H}_i)$ and $sk_i = (p_i, q_i, x_i)$, where N_i is the product of the two $(l_i - 1)$-bit primes p_i, q_i, H_i is a hash function, and g_i is a random generator of the quadratic residue of N_i. Also, it holds for the randomly chosen value x_i that $y_i = g_i^{x_i}$.

The first HVZK protocol PK_1 allows a prover to prove its knowledge of t-out-of-N integers $x_1, \dots, x_N$ such that $y_i = g_i^{x_i}$ and $v_i = h_i^{x_i}$ holds for $1 \leq i \leq N$ to a verifier.

Typical for HVZK protocols, it consists of a commit, challenge, and response phase. By applying the Fiat-Shamir transformation, the authors created a signature scheme, i.e., by replacing the challenge sent by the verifier in the challenge phase with a hash of the commitment together with the message to be signed.

Considering the scenario, where t signers **SI** among the ring **RI** want to prove their knowledge of t-out-of-N discrete logarithms for y_i and h_i, the transformed signature of knowledge for a message msg is denoted by:

$$\mathsf{SPK}_1 \left\{ (\alpha_1, \ldots, \alpha_N) : \bigvee_{\mathbf{SI} \subseteq \mathbf{RI},\ |\mathbf{SI}|=t} \left(\bigwedge_{i \in \mathbf{SI}} y_i = g_i^{\alpha_i} \wedge v_i = h_i^{\alpha_i} \right) \right\} (msg)$$

And works as follows:

1. $\forall i \in \mathbf{NI} : c_i \leftarrow\$\, \mathbb{Z}_{p_1}$ and $\forall i \in \mathbf{RI} : r_i \leftarrow\$\, \mathbb{Z}_{p_2}$.
   Sample values r_i for every signer and c_i values for all parties randomly out of two sets $\mathbb{Z}_{p_1}$, $\mathbb{Z}_{p_2}$ with defined parameters p_1 and p_2 and calculate the commitments t_i and T_i:
   $$t_i \leftarrow \begin{cases} g_i^{r_i}, & i \in \mathbf{SI}; \\ g_i^{r_i} y_i^{c_i}, & i \in \mathbf{NI} \end{cases} \text{ and } T_i \leftarrow \begin{cases} h_i^{r_i}, & i \in \mathbf{SI}; \\ h_i^{r_i} v_i^{c_i}, & i \in \mathbf{NI} \end{cases}$$

2. $c = \mathsf{H}((g_1, y_1, h_1, v_1) \| \ldots \| (g_N, y_N, h_N, v_N) \| t_1 \| \ldots \| t_N \| T_1 \| \ldots \| T_N, msg)$
   Set the challenge to the hash of the commitment and the message.

3. Compute the polynomial f s.t. $deg(f) \leq N - t$, $f(0) = c$, $\forall \in \mathbf{RI} \setminus \mathbf{SI} : f(i) = c_i$ holds. Then, $\forall i \in \mathbf{SI}$ set $c_i \leftarrow f(i)$ and compute:
   $$s_i \leftarrow \begin{cases} r_i - c_i x_i, & i \in \mathbf{SI}; \\ r_i, & i \in \mathbf{NI} \end{cases}$$

4. Output $\sigma = (f, s_1, \ldots, s_N)$

To verify this signature, one needs to check if $deg(f) \leq N - t$ and if the following equation holds with $c_i = f(i)$:

$$\begin{aligned} f(0) = \mathsf{H}(&(g_1, y_1, h_1, v_1) \| \ldots \| (g_N, y_N, h_N, v_N) \| \\ & y_1^{c_1} g_1^{s_1} \| \ldots \| y_N^{c_N} g_N^{s_N} \| v_1^{c_1} h_1^{s_1} \| \ldots \| v_N^{c_N} h_N^{s_N}, msg) \end{aligned}$$

The second signature scheme, SPK_2, is created analogously by performing the Fiat-Shamir transform on the HVZK protocol PK_2. PK_2 allows a prover to prove the knowledge about all discrete logarithms $x_1, \ldots, x_N$ such that $y_i = g_i^{x_i}$ holds for $1 \leq i \leq N$ to a verifier.

A group of signers **SI** having their secret keys $\mathbf{S} = \{sk_i\}_{i \in \mathbf{SI}}$ then creates a (t, N)-thring signature on a message msg and event-id e as follows:

1. $\forall i \in \mathbf{RI} : h_{i,e} \leftarrow \mathsf{H}(param, pk_i, e)$ with $param$ denoting the system parameters and calculate

$$\tilde{y}_{i,e} \leftarrow \begin{cases} h_{i,e}^{x_i}, & i \in \mathbf{SI}; \\ h_{i,e}^{r_i} \text{ for } r_i \leftarrow\!\$\, \mathbb{Z}_{\lfloor N_i/4 \rfloor}, & i \in \mathbf{NI} \end{cases}$$

2. Compute a signature σ_1 with

$$\mathsf{SPK}_1 \left\{ (\alpha_1, \ldots, \alpha_N) : \bigvee_{\mathbf{SI} \subseteq \mathbf{RI},\, |\mathbf{SI}|=t} \left(\bigwedge_{i \in \mathbf{SI}} y_i = g_i^{\alpha_i} \wedge \tilde{y}_{i,e} = h_{i,e}^{\alpha_i} \right) \right\} (msg)$$

3. Compute a signature σ_2 with

$$\mathsf{SPK}_2 \left\{ (\beta_1, \ldots, \beta_N) : \bigwedge_{i=1}^{N} \tilde{y}_{i,e} = h_{i,e}^{\beta_i} \right\} (msg)$$

4. Output $\sigma = ((\tilde{y}_{i,e})_{i \in \mathbf{RI}}, \sigma_1, \sigma_2)$

While in Step 2 the signers prove that they know t secret keys, i.e., the discrete logarithm x_i of y_i, and x_i has been used as the exponent of $h_{i,e}$ to calculate $\tilde{y}_{i,e}$. In Step 3 they prove that they know all discrete logarithms of $\tilde{y}_{i,e}$ and not just t ones.

To verify a given signature σ, the verifier calculates all $h_{i,e}$ values and verifies both SPK signatures, σ_1 and σ_2.

The Link algorithm checks if two signatures σ, σ' are linked if they are valid and there exists an index i such that $\tilde{y}_{i,e} = \tilde{y}'_{i,e}$, additionally it outputs pk_i, the public key of the detected double-signer. Note that this is also why the second proof resulting from SPK_2 is necessary; without this proof, adversaries could break the non-slanderability property by creating a signature that includes at least one $\tilde{y}_{i,e}$ value of an honest user's signature so that these signatures appear to be linked.

Fujisaki et al. [FS07] achieved an improved efficiency compared to this scheme with their traceable thring signature, which only requires the first signature of knowledge SPK_1. Hence, besides the DDHP, their proof of anonymity is reduced to the zero-knowledge of the first HVZK protocol PK_1, rather on both HVZK protocols.

Classification The signing process is interactive (IS) and TA-independent (TAI). Since the linkability property considers the event-id and is even independent of the co-signers, it is event-oriented and individual-linkable, and since two linked signatures reveal the identity of the double-signer, it is accusatory.

Anonymity When considering only the construction of σ_1 using SPK_1 while omitting the second condition with $\tilde{y}$, this would represent an unlinkable thring signature that perfectly hides the signers. This is because both HVZK protocols PK_1, PK_2 are zero-knowledge, which can be proven using a simulator Sim who can foretell the challenge (or has control over the random oracle for SPK_1 and SPK_2). For the first and more complex protocol PK_1, knowing the challenge c, Sim interpolates the polynomial f in advance and then sets all s_i, t_i, T_i values as previously done for the non-signers. Given the simulated transcript (t_i, T_i, f, s_i), it is indistinguishable from a real transcript for PK_1 even though Sim does not have a single secret key of any ring member. However, as in the scheme by Liu [LWW04] discussed in 5.1.1, the addition of $\tilde{y}$ is required to provide the linkability property of this scheme and reduces the anonymity also to the DDHP and to the strong RSA assumption. Here, the strong RSA assumption is the hardness assumption of the scheme's underlying public-key cryptography; if an adversary $\mathcal{A}$ can break it, $\mathcal{A}$ can also deanonymize all honest signers owed to the provided likability.

Given the generator g_i of a ring member i, we can consider the y_i value of its public key and the values $h_{i,e}$, $\tilde{y}_{i,e}$ as a triple $(y_i, h_{i,e}, \tilde{y}_{i,e}) = (g_i^{x_i}, g_i^{\alpha}, g_i^{\gamma})$. If an adversary $\mathcal{A}$ is able to break the DDHP, then it can decide between γ being a random integer $\gamma = r_i$ or being the product of the first two discrete logarithms of the triple s.t. $\gamma = x_i\alpha$. Since the first case applies to all non-signers and the latter one to all signers, $\mathcal{A}$ can deanonymize the signers of a given signature.

Efficiency and Signature Size Signing requires $6n + 2t$ modular exponentations and N polynomial evaluations. Verification requires $6N$ modular exponentations and N polynomial evaluations.

The signature size is $\mathcal{O}(N)$.

6.1.2 Based on the DLP

In 2020, Munch-Hansen et al. [MHOY20] proposed the first thring signature scheme that is size-independent of N. It is based on the DLP, RSA accumulators, non-interactive zero-knowledge arguments of knowledge (NIZKAoK) and the random oracle model.

Given a group $\mathbb{G}$ of order p and its generator g, the key-pair (pk_i, sk_i) of a party $i \in \mathcal{U}$ is created as $sk_i \leftarrow\$\, \mathbb{Z}_p$ and $pk_i \leftarrow g^{sk_i}$ such that pk_i is a prime.

The RSA accumulator ACC introduced by Benaloh and de Mare [BdM94] works as follows:

- $\mathsf{Setup}(1^\lambda)$: Sample random 1^λ bit safe primes $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also primes and sets $m \leftarrow pq$. It samples

$g' \leftarrow\!\$\, \mathbb{Z}_m^*$ and sets $g \leftarrow (g')^2 \bmod m$. Finally, it returns $pp = (m, g)$

- $\mathsf{Eval}_{pp}(\mathcal{X})$: Return $acc_{\mathcal{X}} = g^{\prod_{x \in \mathcal{X}} x} \bmod m$.
- $\mathsf{Wit}_{pp}(acc_{\mathcal{X}}, x)$: Return $wit_x = g^{\prod_{y \in \mathcal{X}, y \neq x} y} \bmod m$.
- $\mathsf{Verify}_{pp}(acc_{\mathcal{X}}, wit_x, x)$: If $(wit_x)^x \bmod m = acc_{\mathcal{X}}$ return 1 and 0 otherwise.

The NIZKAoK scheme allows to prove its knowledge of a witness $w = (pk_i, sk_i, wit_{pk_i})$ for the statement $\phi = (pp, acc_{\mathbf{R}}, \mathsf{H}(msg, nonce), \tilde{y})$ contained in the relation $\mathcal{R} = (pk_i = g^{sk_i}) \wedge (\mathsf{ACC.Verify}_{pp}(acc_{\mathbf{R}}, pk_i, wit_{pk_i})) \wedge (\tilde{y} = \mathsf{H}(msg, nonce)^{sk_i}))$. Intuitively, a verifier can be convinced that the discrete logarithm of $\tilde{y}$ equals a secret key sk_i corresponding to a public key pk_i that is included in the ring $\mathbf{R}$ accumulated by $acc_{\mathbf{R}}$.

Given the public parameters pp of ACC and the common reference string crs of NIZKAoK, each signer $i_s \in \mathbf{SI}$ creates its 1-out-of-N signature in a round $k \in [t]$ as follows:

1. Accumulate $acc_{\mathbf{R}} \leftarrow \mathsf{ACC.Eval}_{pp}(\mathbf{R})$ and compute the witness $wit_{pk_{i_s}} \leftarrow \mathsf{ACC.Wit}_{pp}(acc_{\mathbf{R}}, pk_{i_s})$.
2. Set $nonce = (msg, \mathbf{R})$ and compute $\tilde{y} \leftarrow \mathsf{H}(msg, nonce)^{sk_{i_s}}$.
3. Compute the proof $\pi \leftarrow \mathsf{NIZKAoK.Prove}(crs, \phi, w = (pk_{i_s}, sk_{i_s}, wit_{pk_{i_s}}))$.
4. Output $\sigma_k := (\tilde{y}, \pi)$.

To verify a 1-out-of-N signature σ_k, one needs to recalculate the accumulator $acc_{\mathbf{R}}$ and check if the given proof π verifies: $\mathsf{NIZKAoK.Verify}(crs, \phi, \pi)$.

Given two 1-out-of-N signatures, σ_k, σ_k', the Link algorithm checks whether they are linked if $\tilde{y} = \tilde{y}'$ holds.

Lastly, all 1-out-of-N signatures σ_k can be combined to get a t-out-of-N signature $\sigma := \{\sigma_k\}_{k \in [t]}$, where they all need to be unlinked to each other.

Classification The signing phase of this scheme is non-interactive (NIS) and TA-independent (TAI). The scheme does require a trusted setup to generate a global common reference string for the NIZKAoK, but after the setup, no trusted authority is needed to create a signature. The linkability is individual linkable, non-accusatory, and since it considers the signed message, it is event-oriented.

Anonymity This scheme is anonymous if the non-interactive zero-knowledge argument of knowledge NIZKAoK and the RSA-accumulator ACC are secure and the DDHP is hard in $\mathbb{G}$ under the random oracle model. As in [LWW04] and [TWC+05] discussed in 5.1.1 and 6.1.1, due to the linkability tag $\tilde{y}$ created in Step 2 for ensuring the linkability of this scheme, the anonymity is reduced to the hardness of the DDHP. Additionally, the DLP, which implies the DDHP, also needs to be hard, or otherwise, an adversary could compute any secret key of a public key and create a signature on its own to check if it is linked to a given one. Furthermore, the $\mathsf{NIZKAoK}$ scheme needs to satisfy the zero-knowledge property such that an honest proof should be indistinguishable from a simulated proof created with a trapdoor τ. Since the witness w contains the identity of the signer (and even its secret key), the proof π must not reveal any information about it, except that the signer has knowledge of a witness w and it fulfills the statement ϕ in the relation $\mathcal{R}$. Additionally, the authors require that each ring member's public key needs to be checked, i.e., that it is a prime number, before a signer should continue with the signing process; hence this scheme is secure against adversarially-chosen public keys.

Efficiency and Signature Size The signing process for a t-out-of-N signature requires $2tN$ modular exponentiations and the creation of a $\mathsf{NIZKAoK}$ proof for the specific relation $\mathcal{R}$. This proof should be a rather efficient and compact as it mostly consists of statements about the knowledge of exponents, according to the authors. Verification requires tN modular exponentiations and the verification of the $\mathsf{NIZKAoK}$ proof π.

Lastly, the signature size is independent of the number of the ring members and results to $\mathcal{O}(t)$ due to the common reference string based argument, wherein the statement is not part of the signature.

6.1.3 Based on VRFs

Built on the logarithmic-sized ring signature scheme presented by Backes et al. [BDH+19], Haque et al. [HKSS20] recently proposed a thring signature scheme based on verifiable random functions (VRF) and non-interactive witness-indistinguishable proofs (NIWI) in the plain model. Additional building blocks are a public key encryption scheme (PKE), somewhere perfectly binding hashing (SPB), and a one-way permutation F.

The public- and secret key-pair of each party $i \in \mathcal{U}$ corresponds to $PK_i = (vk_i, pk_i^{(1)}, pk_i^{(2)}, E)$ and $SK_i = (sk_i, sk_i^{(1)}, sk_i^{(2)}, r_E)$ with the following meanings: The key-pair (vk_i, sk_i) belongs to a VRF scheme, $(pk_i^{(1)}, sk_i^{(1)})$ and $(pk_i^{(2)}, sk_i^{(2)})$ are two PKE key-pairs, $r_E \leftarrow\$\ \mathsf{PKE}.R$ is a random value out of the randomness space of PKE, and $E \leftarrow \mathsf{PKE.Enc}(\mathsf{pk_i^{(1)}}, \mathsf{sk_{F,i}}; \mathsf{r_E})$ with $sk_{\mathsf{F},i} \leftarrow\$\ \{0,1\}^{2\lambda}$.

A t-out-of-N thring signature σ consists of t 1-out-of-N ring signatures σ_k, wherein the authors allow that each signer $i_s \in \mathbf{SI}$ may choose an individual threshold t_{i_s}. However, for simplicity, we omit this feature since it does not effect the way the signers are hidden in this scheme.

Each signer $i_s \in \mathbf{SI}$ creates its 1-out-of-N signature in a round $k \in [t]$ as follows:

1. Compute $y \leftarrow \mathsf{VRF.Eval}(sk_{i_s}, msg\|\mathbf{R})$, $p \leftarrow \mathsf{VRF.Prove}(sk_{i_s}, msg\|\mathbf{R})$.
2. Set $r_{ct} \leftarrow\$ \mathsf{PKE}.R$ and compute $ct \leftarrow \mathsf{PKE.Enc}(pk_{i_s}^{(1)}, p; r_{ct})$.
3. Generate $(hk_{i_s}, shk_{i_s}) \leftarrow \mathsf{SPB.KeyGen}(1^\lambda, N, i_s)$ and compute $h_{i_s} \leftarrow \mathsf{SPB.Hash}(hk_{i_s}, \mathbf{R})$ and $wit_{i_s} \leftarrow \mathsf{SPB.Open}(hk_{i_s}, shk_{i_s}, \mathbf{R}, i_s)$.
4. Choose a random ring member $i \leftarrow\$ \mathbf{RI} \setminus \{i_s\}$ and set $r_E \leftarrow\$ \mathsf{PKE}.R$, $(hk_i, shk_i) \leftarrow \mathsf{SPB.KeyGen}(1^\lambda, N, i)$, $h_i \leftarrow \mathsf{SPB.Hash}(hk_i, \mathbf{R})$, and $wit_i \leftarrow \mathsf{SPB.Open}(hk_i, shk_i, \mathbf{R}, i)$.
5. Compute a NIWI proof π that one of the following claims is true:
   (a) $(\phi, w) \in \mathcal{R}_{PK}$ with $\phi = (msg, \mathbf{R}, t, y, ct, hk_{i_s}, h_{i_s})$ and $w = (PK_{i_s}, i_s, p, r_{ct}, wit_{i_s})$ if and only if $\mathsf{SPB.Verify}(hk_{i_s}, h_{i_s}, i_s, PK_{i_s}, wit_{i_s}) = 1 \wedge \mathsf{PKE.Enc}(pk_{i_s}^{(1)}, p; r_{ct}) = ct \wedge \mathsf{VRF.Verify}(vk_{i_s}, msg\|\mathbf{R}, y, p) = 1$.
   (b) $(\phi, w) \in \mathcal{R}_{PK'}$ the same as 5a but for ring member i.
   (c) $(\phi, w) \in \mathcal{R}_{\mathcal{F}}$ with $\phi = (\mathbf{R}, h_{i_s}, h_i, hk_{i_s}, hk_i)$ and $w = (i_s, i, PK_{i_s}, PK_i, wit_{i_s}, wit_i, sk_{\mathsf{F},i_s}, sk_{\mathsf{F},i}, r_{E,i_s}, r_{E,i})$ if and only if $\mathsf{F}(sk_{\mathsf{F},i_s}) = sk_{\mathsf{F},i} \wedge \mathsf{PKE.Enc}(pk_{i_s}^{(2)}, sk_{\mathsf{F},i_s}; r_{E,i_s}) = E_{i_s} \wedge$ $\mathsf{PKE.Enc}(pk_{i_s}^{(2)}, sk_{\mathsf{F},}; r_{E,i}) = E_i \wedge$ $\mathsf{SPB.Verify}(hk_{i_s}, h_{i_s}, i_s, PK_{i_s}, wit_{i_s}) = 1 \wedge$ $\mathsf{SPB.Verify}(hk_i, h_i, i, PK_i, wit_i) = 1$.
6. Output $\sigma_k := (y, ct, hk_{i_s}, hk_i, \pi)$

A 1-out-of-N signature σ_k is verified by recalculating both hashes h_{i_s} and h_i using hk_{i_s} and hk_i, respectively, and by checking if the proof π verifies $\mathsf{NIWI.Verify}(\phi, \pi)$ with $\phi = (msg, \mathbf{R}, y, ct, h_{i_s}, h_i, hk_{i_s}, hk_i)$.

Even though the authors did not explicitly mentioned the linkability of their first version of this scheme, we still consider it as linkable; two signatures, σ_k, σ'_k, are linked if $y = y'$ holds. Note that since $\mathsf{VRF.Eval}$ is deterministic, no signer can create multiple valid signatures on the same message with respect to the same ring without them being linked.

Lastly, a t-out-of-N thring signature corresponds to $\sigma := \{\sigma_k\}_{k\in[t]}$, whereas all 1-out-of-$N$ signatures need to be unlinked to each other.

Classification The signing phase is non-interactive (NIS) and TA-independent (TAI). The linkability is individual linkable, event-oriented, and non-accusatory.

Anonymity First of all, notice that an honest signer $i_s \in \mathbf{SI}$ does not use its second PKE key-pair $(pk_{i_s}^{(2)}, sk_{i_s}^{(2)})$, or E and r_E for creating a signature. These values are only needed for the proof of anonymity to simulate a NIWI proof, where the challenger picks proper secret keys sk_{F} to exploit condition 5c of the proof. Note that such an adaption is necessary due to the missing trapdoor of NIWI proofs, which in turn relies on the fact that NIWI proofs do not need a common reference string. Secondly, the inclusion of another ring member i in Step 4 of the signing phase is also required to prove this scheme's anonymity. Because then it is possible to transform a signature created by a signer i_s to a signature created by another ring member i over a sequence of hybrid arguments. If each change to the next hybrid is indistinguishable, it follows that it is also indistinguishable for the final signature whether it has been created by ring member i_s or i_0.

Essentially, this scheme ensures anonymity if VRF is pseudorandom and has key-privacy, SPB is index-hiding, PKE has key-privacy and CPA-security, and NIWI is witness-indistinguishable.

Since every 1-out-of-N signature σ_k contains the signer's VRF evaluation of $msg\|\mathbf{R}$, y needs to be indistinguishable from random, and it should be impossible to infer to which verification key y belongs to, as long as the corresponding proof p remains secret.

The index-hiding property of SPB ensures that the two hashing keys hk_{i_s}, hk_i do not reveal the indices i_s and i with which they were initially generated and where one of them necessarily belongs to the signer of σ_k.

The key-privacy and CPA-security of PKE is needed so that given the ciphertext ct of a signature σ_k, it should computationally hide the public key $pk_i^{(1)}$ used for the encryption and the encrypted proof p of the VRF evaluation.

Finally, NIWI needs to fulfill witness-indistinguishability, which ensures that it should be indistinguishable for every efficient adversary to decide which witness was used to compute a proof π. Under the assumption that all previously properties hold, this implicates that it should be indistinguishable which ring member is the creator of a signature σ_k.

Since the non-signers' public keys aren't explicitly used during the signing procedure, except that they are included in the ring, which is used as an argument for VRF and SPB, this scheme is secure against adversarially-chosen public keys.

Efficiency and Signature Size Creating a t-out-of-N thring signature requires t VRF evaluations and proofs; t PKE encryptions; $2t$ SPB key-generations, hashing and opening computations; and computing t NIWI proofs for a rather complex language. When considering the feature of individual thresholds, twice as many of the previously mentioned operations are necessary.

Verification requires two $2t$ SPB hashing computations and verifying a

NIWI proof.

Due to SPB hashing, membership witnesses of a ring of size N can be achieved in $\mathcal{O}(\log(N)\mathsf{poly}(\lambda))$ for a polynomial poly. Thus, the signature size of this scheme is in $\mathcal{O}(t\log(N))$.

6.2 Unlinkable Variants

6.2.1 Based on the MDP

Based on a generalization of Stern's zero-knowledge identification protocol [Ste94], Aguilar Melchor et al. [AMCG08] proposed a thring signature that relies on the minimal distance problem (MDP) and on the random oracle model.

The secret key of a party $i \in \mathcal{U}$ equals $sk_i = s_i$, where s_i corresponds to a word of weight w (common for all parties) in the code C_i defined by the $(n-k) \times n$ parity check matrix H_i, which represents the public key $pk_i = H_i$. Note that finding such a non-zero vector s_i with $wt(s_i) \leq w$ for a given matrix H_i so that $H_i s_i^T = 0$ holds, relates to the minimum distance problem, which is assumed to be hard for this scheme.

First, the signers create a *ring public key* in form of an $N(n-k) \times Nn$ matrix H defined as the direct sum of all ring members' public keys $H = \oplus_{i=1}^{N} H_i$. The matrix H, resulting from the direct sum, has the matrices H_i in the diagonal such that H_i is at position (i,i) and all other entries are 0.

The actual signature scheme results from applying the Fiat-Shamir transformation on a generalized version of Stern's identification protocol, which is a three-move HVZK scheme and works as follows:

1. $\forall i \in \mathbf{RI}$: Set $y_i \leftarrow\$ \mathbb{F}_2^n$, set π_i to a random permutation of $\{1,\ldots,n\}$, and compute the commitments with $s_i \leftarrow 0$ for $i \in \mathbf{NI}$:
   $c_{1,i} \leftarrow \mathsf{H}(\pi_i \| H_i y_i^T)$, $c_{2,i} \leftarrow \mathsf{H}(\pi_i(y_i))$, $c_{3,i} \leftarrow \mathsf{H}(\pi_i(y_i \oplus s_i))$.

2. Set Σ to a random constant n-block permutation on N blocks $\{1,\ldots,N\}$ and compute and send the *master commitments* to the verifier:
   $C_1 \leftarrow \mathsf{H}(\Sigma \| c_{1,1} \| \ldots \| c_{1,N})$, $C_2 \leftarrow \mathsf{H}(\Sigma(c_{2,1} \| \ldots \| c_{2,N})$,
   $C_3 \leftarrow \mathsf{H}(\Sigma(c_{3,1} \| \ldots \| c_{3,N}))$.

3. The verifier sends a challenge $b \leftarrow\$ \{0,1,2\}$ to the signers.

4. The signers set $\Pi \leftarrow \Sigma \circ \pi$ with $\pi \leftarrow (\pi_i)_{i\in\mathbf{RI}}$, $y \leftarrow (y_i)_{i\in\mathbf{RI}}$, and $s \leftarrow (s_i)_{i\in\mathbf{RI}}$, and answer the verifier:

   - If $b = 0$: Send y and Π.
   - If $b = 1$: Send $y \oplus s$ and Π.
   - If $b = 2$: Send $\Pi(y)$ and $\Pi(s)$.

5. The verifier checks that:

- If $b = 0$: Π is a n-block permutation and that C_1 and C_2 were created correctly.
- If $b = 1$: Π is a n-block permutation and that and that C_1 and C_3 were created correctly.
- If $b = 2$: C_2 and C_3 were created correctly, that $\Pi(s)$ has a weight of tw and is formed of N blocks of length n of weigth w or 0.

Notice that this HVZK protocol has a cheating probability or soundness error of 2/3, thus it needs to be repeated until the required security level is reached.

However, the authors did not elaborate on how exactly one would apply the Fiat-Shamir transform on this HVZK scheme to obtain a non-interactive signature scheme. A possible approach would be that the signers have to repeat the first two steps K times (depending on the desired security level), to obtain K rows of master commitments $(C_1^k, C_2^k, C_3^k)_{k=1}^K$. Then, all challenges are computed using a hash function $\mathsf{H}' : \{0,1\}^* \to \{0,1,2\}^K$ as $b_1, \dots, b_K \leftarrow \mathsf{H}'(msg \| (C_1^k, C_2^k, C_3^k)_{k=1}^K))$, where the respective responses have to be included in the signature.

Besides, Cayrel et al. [CLRS10b] later proposed a lattice-based thring signature scheme as a modification of this scheme with a reduced cheating probability of 1/2 per round. This improvement is achieved by relying on the short integer solution (SIS) problem and by applying the CLRS identification protocol [CLRS10a] instead of Stern's protocol. Nevertheless, the way the signers are hidden among the ring is still similar as in the scheme by Aguilar Melchor.

Classification The signing process is interactive (IS) and TA-independent (TAI).

Anonymity This scheme provides unconditional anonymity for the signers. To see this, we have to analyze the three openings the signers reveal depending on the challenge. If $b = 0$, the signers reveal the random vectors y and the n-block permutation Π, where both do not contain any information about the signers' public- or secret keys. If $b = 1$, the signers reveal $y \oplus s$ and Π. Since y is randomly sampled and kept secret, $y \oplus s$ corresponds to a one-time pad of the secret keys s. If $b = 2$, the signers reveal $\Pi(y)$ and $\Pi(s)$. At first, each secret key s_i is permuted using the random permutation π, and those blocks are again permuted using the random constant n-block permutation Σ. Both permutations are kept secret, and only the results of applying the block permutation Π on s and y are published. Furthermore, this list of blocks $\Pi(s)$ hides the signers relying on the fact that every party's secret key has the null common syndrome and all have the same weight w. Hence, the t blocks of weight w only prove that t parties made use of their secret keys, but neither can an adversary reconstruct them nor tell which ring members they belong to.

We can also argue the zero-knowledge property using a simulator Sim who does not know any secret key but can create a valid proof and therefore a valid signature when foretelling/controlling the challenge b. In all cases, Π is set to a random n-block permutation and y to a list of random vectors. If $b = 0$ or $b = 2$, Sim sets s to a list of random vectors, where t of which have a weight of w and all others are equal to 0. Otherwise, if $b = 1$, Sim sets s to 0. It is trivial to see that in all cases the checks for the corresponding commitments verify.

Efficiency and Signature Size In this scheme by Aguilar Melchor, syndrome decoding is the computationally most expensive operation and KN are needed for signing and averagely $2/3KN$ for verifying.

Each public key H_i has a size of $n^2/2$ bits, whereas *double-circulant matrices* would reduce the description of H_i to $n/2$ bits.

In comparison, the lattice-based variant by Cayrel et al. needs KN matrix- vector multiplications for signing and averagely $1/2KN$ for verifying.

Regarding the signature size, for a security level of 2^{-80}, the scheme by Aguilar Melchor et al. requires 140 rounds, while the scheme by Cayrel et al. requires only 80 rounds. Nevertheless, according to Petzoldt [PBB13], when considering a ring size of 100, the signature size results to 1.4 MBytes for the coding-based scheme by Aguilar Melchor et al., but for the lattice-based scheme by Cayrel et al., the size results to 26.7 MBytes.

6.2.2 Based on the MQP

By extending the multivariate identification scheme of Sakumoto et al. [SSH11], Petzoldt et al. [PBB13] proposed a thring signature and identification scheme that is based on the MQ-problem and on a secure commitment scheme Com.

The public- and secret key-pair of a party $i \in \mathcal{U}$ corresponds to $(pk_i, sk_i) = (\mathcal{P}_i, s_i)$, where $\mathcal{P}$ is a random quadratic system $\mathcal{P}_i : \mathbb{F}_2^n \to F_2^m$ such that for the secret vector $s_i \in \mathbb{F}_2^n$, it holds that $\mathcal{P}_i(s_i) = 0$. In order to simulate the the non-signers' parts of the signature without knowing their secret keys, $\mathcal{P}_i$ must not contain any constant terms such that $\mathcal{P}_i(0)$ is also a valid solution.

For a multivariate system $\mathcal{P}$, its *polar form* $\mathcal{G}$ is defined as $\mathcal{G}(x_1, x_2) = \mathcal{P}(x_1 + x_2) - \mathcal{P}(x_1) - \mathcal{P}(x_2)$. The bilinearity of $\mathcal{G}$ enables the creation of zero-knowledge proofs about the secret vector s. When dividing s into $s = r_0 + r_1$, it holds that $0 = \mathcal{P}(r_0) + \mathcal{P}(r_1) + \mathcal{G}(r_0, r_1)$. Since this terms contain both, r_0 and r_1, r_0 is further divided into $r_0 = t_0 + t_1$, and $\mathcal{P}(r_0)$ into $\mathcal{P}(r_0) = e_0 + e_1$. Then, one can rewrite this equation to $0 = (\mathcal{G}(t_0, r_1) + e_0) + (\mathcal{P}(r_1) + \mathcal{G}(t_1, r_1) + e_1)$. Hence, knowing such a tuple $(r_0, r_1, t_0, t_1, e_0, e_1)$

proves the knowledge of the secret key s, whereas this proof is extended to the threshold scenario in a three-move HVZK protocol.

Similarly as in [AMCG08] discussed in 6.2.1, a signature is created by applying the Fiat-Shamir transform on an interactive HVZK identification protocol, which follows the above observation and works as follows:

1. $\forall i \in \mathbf{RI}$: Set $r_0^{(i)}, t_0^{(i)} \leftarrow\!\$\, \mathbb{F}_2^n$, $e_0^{(i)} \leftarrow\!\$\, \mathbb{F}_2^m$ and compute $r_1^{(i)} \leftarrow s_i - r_0^{(i)}$, $t_1^{(i)} \leftarrow r_0^{(i)} - t_0^{(i)}$, and $e_1^{(i)} \leftarrow \mathcal{P}_i(r_0^{(i)}) - e_0^{(i)}$ with $s_i \leftarrow 0$ for $i \in \mathbf{NI}$.

2. $\forall i \in \mathbf{RI}$: Compute the commitments $c_0^{(i)} \leftarrow \mathsf{Com}(r_1^{(i)}, \mathcal{G}(t_0, r_1) + e_0^{(i)})$, $c_1^{(i)} \leftarrow \mathsf{Com}(t_0^{(i)}, e_0^{(i)})$, $c_2^{(i)} \leftarrow \mathsf{Com}(t_1^{(i)}, e_1^{(i)})$, $c_3^{(i)} \leftarrow \mathsf{Com}(r_0^{(i)})$, and $c_4^{(i)} \leftarrow \mathsf{Com}(r_1^{(i)})$.

3. Set Σ to a random permutation of $\{1, \dots, N\}$ and compute and send the master commitments to the verifier: $C_0 = \mathsf{Com}(c_0^{(1)}, \dots, c_0^{(N)})$, $C_1 = \mathsf{Com}(\Sigma, c_1^{(1)}, \dots, c_1^{(N)})$, $C_2 = \mathsf{Com}(c_2^{(1)}, \dots, c_2^{(N)})$, $C_3 = \mathsf{Com}(\Sigma(c_3^{(1)}, \dots, c_3^{(N)}))$, $C_4 = \mathsf{Com}(\Sigma(c_4^{(1)}, \dots, c_4^{(N)}))$.

4. The verifier sends a challenge $b \leftarrow\!\$\, \{0, 1, 2, 3\}$ to the signers.

5. The signers set $r_0 \leftarrow (r_0^{(i)})_{i \in \mathbf{RI}}$, $r_1 \leftarrow (r_1^{(i)})_{i \in \mathbf{RI}}$, $t_0 \leftarrow (t_0^{(i)})_{i \in \mathbf{RI}}$, $t_1 \leftarrow (t_1^{(i)})_{i \in \mathbf{RI}}$, $e_0 \leftarrow (e_0^{(i)})_{i \in \mathbf{RI}}$, and $e_1 \leftarrow (e_1^{(i)})_{i \in \mathbf{RI}}$ and answer the verifier:
   - If $b = 0$: Send (r_0, t_1, e_1).
   - If $b = 1$: Send (r_1, t_1, e_1).
   - If $b = 2$: Send (r_1, t_0, e_0).
   - If $b = 3$: Send $(\Sigma(c_3^{(1)}, \dots, c_3^{(N)}), \Sigma(c_4^{(1)}, \dots, c_4^{(N)}))$.

6. The verifier checks the commitments:
   - If $b = 0$: $\forall i \in \mathbf{RI}$: Compute $\tilde{c}_1^{(i)} \leftarrow \mathsf{Com}(r_0^{(i)} - t_1^{(i)}, \mathcal{P}_i(r_0^{(i)}) - e_1^{(i)})$, $\tilde{c}_2^{(i)} \leftarrow \mathsf{Com}(t_1^{(i)}, e_1^{(i)})$, $\tilde{c}_3^{(i)} \leftarrow \mathsf{Com}(r_0^{(i)})$ and check the correctness of C_1, C_2, and C_3.
   - If $b = 1$: $\forall i \in \mathbf{RI}$: Compute $\tilde{c}_0^{(i)} \leftarrow \mathsf{Com}(r_1^{(i)}, -\mathcal{P}_i(r_1^{(i)}) - \mathcal{G}_i(t_1^{(i)}, r_1^{(i)}) - e_1^{(i)})$, $\tilde{c}_2^{(i)} \leftarrow \mathsf{Com}(t_1^{(i)}, e_1^{(i)})$, and check the correctness of C_0 and C_2.
   - If $b = 2$: $\forall i \in \mathbf{RI}$: Compute $\tilde{c}_0^{(i)} \leftarrow \mathsf{Com}(r_1^{(i)}, \mathcal{G}_i(t_0^{(i)}, r_1^{(i)}) + e_0^{(i)})$, $\tilde{c}_1^{(i)} \leftarrow \mathsf{Com}(t_0^{(i)}, e_0^{(i)})$, $\tilde{c}_4^{(i)} \leftarrow \mathsf{Com}(r_1^{(i)})$ and check the correctness of C_0, C_1, and C_4.
   - If $b = 3$: Check the correctness of C_3 and C_4, and that there are at least t indices in $i \in \mathbf{RI}$ s.t. $c_3^{(\Sigma(i))} \neq c_4^{(\Sigma(i))}$.

This HVZK has a cheating probability of 3/4 and therefore needs to repeated K times depending on the the expected security level. The Fiat-Shamir transform is analogously performed as in the approach shown for 6.2.1.

Classification The resulted signature scheme has an interactive signing phase (IS) and is TA-independent (TAI).

Anonymity The signers of a thring signature are perfectly hidden since this transformed identification protocol is zero-knowledge assuming that the commitment scheme Com is statistically hiding. First of all, every signers' secret key s_i is divided into $r_0^{(i)}$ and $r_1^{(i)}$, while $r_0^{(i)}$ is again divided into $t_0^{(i)}$ and $t_1^{(i)}$. Since no response contains both $r_0^{(i)}$ and $r_1^{(i)}$ or $t_0^{(i)}$, $t_1^{(i)}$ and $r_1^{(i)}$, it is not possible to recalculate the secret key s_i. Also, given a triple corresponding to one of the first three responses, it is impossible to tell if it belongs to a signer or a non-signer. This is due to the fact that every $r_1^{(i)}$ vector is a random element of $\mathbb{F}_2^n$ regardless of whether s_i is set to 0 or is an actual secret key, which is also just a random element of $\mathbb{F}_2^n$; both are subtracted with the randomly sampled $r_0^{(i)}$ vector. Also, $t_1^{(i)}$ and $e_1^{(i)}$ are randomly distributed over $\mathbb{F}_2^n$ for every ring signer due to $r_0^{(i)}$, $t_0^{(i)}$, and $e_0^{(i)}$. For $b = 3$, the commitments $c_3^{(i)}$ and $c_4^{(i)}$ are different for every signer and equal for every non-signer because their s_i vector is set to 0 so that $r_0^{(i)}$ and $r_1^{(i)}$ are equal, too. Therefore, the random permutation Σ ensures that the signers' indices remain unknown among the ring; the verifier can only check that there are at least t indices, where $c_3^{(\Sigma(1))}$ and $c_4^{(\Sigma(1))}$ are unequal and belong to a signer in the ring.

Similarly to [AMCG08] analyzed in 6.2.1, one can further prove the zero-knowledge property of this protocol by constructing a simulator Sim that can produce a valid HVZK transcript and hence a valid signature without knowing any secret key. It only needs know beforehand which challenge b will not be chosen in each repetition of the protocol. $e_1^{(i)} \leftarrow \mathcal{P}_i(r_0^{(i)}) - \mathcal{G}(t_0, r_1) - e_0^{(i)}$ Regarding the signature scheme, Sim needs to be able to control the output of the random oracle used for the Fiat-Shamir transformation.

Efficiency and Signature Size Here, evaluating quadratic- multivariate polynomials for each $\mathcal{P}_i$ system is the operation with the highest computational costs. The authors claim that multivariate cryptosystems tend to be faster than classical public-key schemes such as RSA. For signing, $4KN$ system evaluations of m equations in N variables are required and approximately $7/4KN$ for verifying.

The signature size is in $\mathcal{O}(KN)$ and specifically results to $160 + K(464 + 266N)$ bits using a slightly optimized version of the protocol, according to the authors. For instance, for a security level of 2^{-80}, 193 rounds are required, and with a ring size of 100, the signature size is 0.64 MBytes.

7 Conclusion

In the last chapter we summarize the results of this work and discuss some observable trends for thring signatures and possible future work .

7.1 Results

The results of our analysis can be found in Table 1. Note that the columns regarding the classification represent the most favorable properties in terms of anonymity, as we discussed in Section 3.4.1.

We identified three different major construction types of thring signatures, where the principle of how the signers are hidden among the ring is similar for the schemes of each type. Of course, the precise way of ensuring anonymity may still vary depending on the specific scheme; hence we also examined various approaches for each construction type.

Secret Sharing: For the first construction type using Shamir's secret sharing, each ring member's share of the signature lies on a polynomial. While the non-signers' shares have been used to interpolate the polynomial, the signers' shares have been computed using the polynomial evaluation with the aid of the signers' secret keys. The principle idea of why the signers are hidden is that it is impossible to tell which shares have been used for the interpolation- or have been obtained by evaluating the randomly generated polynomial.

Two exemplary schemes that employ Shamir's secret sharing using trapdoor permutations are the first scheme by Bresson et al. [BSS02] and the code-based scheme by Dallot and Vergnaud [DV09]. The scheme [BSS02]-1 uses RSA-trapdoor permutations, thus the computationally most expensive operations are modular exponentiations. The scheme [DV09] uses trapdoor permutations based on the syndrome decoding problem (SDP) that is believed to be post-quantum secure [CM10], thus syndrome computations and syndrome decodings (for the signing phase) are required.

A different approach has been chosen by Chow et al. [CHY05] and Haque et al. [HS20]; in both cases, all shares are included in a hash digest for the initial point of the polynomial and after its interpolation, the signers adjust other values that are connected to their fixed shares. In the case of the identity based scheme [CHY05], the correctness of the signature is verified using bilinear pairings, whereas their evaluation also corresponds to the most expensive operation. Even though their scheme provides unconditional anonymity, the required trusted authority can cause a risk for anonymity, from which every party needs to request their secret key before being able to sign. For example, if the trusted authority logs every party that requested its secret key and this log gets revealed, an adversary can exclude signer candidates of a ring if they never requested their secret keys.

The post-quantum secure scheme [HS20], is based on trapdoor commitments, whereas the signers use their trapdoor to adapt their openings. Therefore its anonymity is reduced to the trapdoor indistinguishability of the instantiated trapdoor commitment scheme.

Ring Hashing: For the second construction type using ring hashing, all ring members' shares of the signature contribute to the fulfillment of a cyclic ring equation that is constructed using hash functions. To create a signature, each signer must compute a hash digest in advance, hash along the ring, and eventually use its secret key to close the ring. However, this closing step is not recognizable as an outsider; one can only verify the signature's validness, i.e., that the ring equation verifies.

While extending this construction type to the threshold scenario is straightforward for linkable schemes, the fact that a verifier can only check the correctness of a ring equation demands additional techniques to ensure that a signature has been created by at least t distinct signers while simultaneously providing unlinkability.

For instance, the thring signature by Liu et al. [LWW04] consists of t linkable 1-out-of-N signatures such that a t-out-of-N signature is only valid if all t 1-out-of-N signatures are valid and unlinked to each other. A benefit of this method is that the signing phase is non-interactive, so the t signers do not have to communicate with one another to create a signature. Besides the hardness of the discrete logarithm problem (DLP), which is the assumption needed for its underlying public-key cryptosystem, the anonymity of each 1-out-of-N signature is reduced to a stronger assumption, the decisional Diffie-Hellman problem (DDHP). Modular exponentiations are the most expensive operations of this scheme.

Okamoto et al. [OTYO18] achieved a non-interactive signing phase in a similar way but by remaining unlinkable. However, their approach requires a trusted authority that distributes and saves short-time keys for each ring-message combination, which further allows previously non-signers to extend an existing t-out-of-N thring signature to a $t+1$-out-of-N signature. If those short-time keys are revealed for a specific ring- message combination, the signers can be identified for this signature but only for this signature since their long-time keys are not connected to their short-time keys. Also for this scheme, the most expensive operations are modular exponentiations.

A different approach using message block sharing, which also requires a trusted authority, can be found in the lattice-based scheme proposed by Chen et al. [CHGL19]. In their scheme, the signers cannot sign any arbitrary messages, but a trusted authority must first preprocess and distribute chosen messages to all users in the system. This technique ensures that only coalitions of more or equal than t users can reconstruct the original message and thus are able to create a thring signature on this message. The actual signing phase is based on a family of hash functions proposed by Lyubashevsky [Lyu08], whereas their collision resistance can be reduced to the shortest vector problem (SVP), which is believed to be secure against quantum attacks [CCKK15]. Here, calculating polynomial vector products is the computationally most expensive operation.

Finally, an unlinkable thring signature scheme, which does not require any trusted authority, has been proposed with the second scheme by Bresson et al. [BSS02]-2 using fair partitions. The signature consists of $2^t \log N$ partitions, each of which having t sub-rings, such that every combination of t possible signers has a fair partition, i.e., that there exists a signer in each sub-ring. The actual signers can then close a so-called super-ring using their own signer partition. If this super-ring equation holds, a verifier can be convinced that at least t signers have participated in the signing procedure. However, since there is a fair partition for all possible signer subsets, all those subsets are equally likely to be the true signer group. Due to RSA trapdoor permutations, modular exponentiations are the most expensive operations. A disadvantage of this approach is the large signature size, which results in $\mathcal{O}\big(2^t N \log N\big)$.

Zero-Knowledge Proofs: For the third construction type using zero-knowledge proofs, the signers employ a zero-knowledge proof about the thring signature's correctness, whereas the zero-knowledge property is used to ensure the anonymity of the signers among the ring.

Two exemplary linkable thring signature schemes, which both applied zero-knowledge proofs of knowledge (PoK) for creating the signature, were proposed by Tsang et al. [TWC$^+$05] and Fujisaki et al. [FS07]. In both cases, the signers prove that for at least t values contained in the signature, the discrete logarithm equals the secret key of the corresponding ring member. This type of joint zero-knowledge requires interaction between the signers. Due to the provided linkability, the anonymity of both schemes is reduced to the DDHP and the assumption of their public-key cryptosystem, strong RSA for [TWC$^+$05] and the DLP for [FS07]. Furthermore, both schemes require modular exponentiations, and the signature size grows linear with the ring size.

More recently, Munch-Hansen et al. [MHOY20] and Haque et al. [HKSS20] presented two linkable thring signatures with a signature size that is, for the first time, sub-linear to the ring size. In both schemes, a t-out-of-N signatures consists of t 1-out-of-N signatures, resulting in a non-interactive signing phase. While in [MHOY20], constant-sized membership witnesses of the ring are realized with accumulators, [HKSS20] uses somewhere perfectly binding hashing to achieve logarithmic-sized witnesses. Furthermore, in [MHOY20], every signer needs to use its secret key as the exponent of a hash digest during the signing phase, which ensures linkability and reduces the anonymity to the DDHP. In contrast, [HKSS20] uses verifiable random functions and a public key encryption scheme, reducing the anonymity to the security properties of these instantiated primitives. Lastly, [MHOY20] utilizes a non-interactive zero-knowledge argument of knowledge, whereas its language consists of statements about the knowledge of discrete logarithms. However, such proof requires a common-reference-string, which has to be generated in a trusted setup; the non-interactive witness indistinguishable proof used in [HKSS20], in contrast, does not require any trusted setup but operates on a more complex language.

Regarding unlinkable thring signature schemes under this construction type, we examined three schemes, where all of which are based on quantum secure problems. Furthermore, all schemes modified an honest-verifier zero-knowledge identification protocol to the threshold scenario and transformed it into a signature scheme using the Fiat-Shamir heuristic. The first code-based scheme by Aguilar Melchor [AMCG08] essentially hides the signers using one-time-pads and random permutations; it relies on the minimal distance problem (MDP) and requires syndrome decodings. The lattice-based scheme by Cayrel et al. [CLRS10b] can be considered as an improvement of the [AMCG08] scheme with a reduced soundness error per round and by relying on the short-integer solution problem (SIS) while having matrix- vector multiplications as the most expensive operation. Lastly, the multivariate-based scheme by Petzoldt et al. [PBB13] hides the signers by using the multivariate system's polar-form and a statistically hiding commitment scheme. It is further based on the MQ-problem and requires evaluating quadratic-multivariate polynomials.

A disadvantage of this common approach is that the underlying identification protocol need many repetitions to ensure an adequate security level, which affects both, the amount of data the signers must exchange, as well as the signature size since all rounds (commits, challenges, and responses) have to be included in the final signature.

Pairings: Another currently rather rare construction type we have observed for only two thring signature schemes so far, namely [YLA$^+$11] and [YLA$^+$13], both by Yuen et al., is based on pairings. These thring signatures are verified by checking multiple equations using bilinear pairings that prove their correctness. This type of verification has already been seen in the examined scheme by Chow et al. [CHY05]; however, the crucial method for constructing their thring signature is Shamir's secret sharing. As standard pairing-based signature schemes such as [BB04, YLTZ09, SOO09] and group signature schemes [BW06, ACHdM05] were proposed that do not require the random oracle model, thereupon pairing-based ring signatures schemes [SW07, CLWY05, Fuj11, GW18] and thring signature schemes [YLA$^+$11, YLA$^+$13] without random oracles were also presented. As a downside, both thring signature schemes under this construction type require a common reference string generated during a trusted setup. Apart from a trusted setup, both schemes do not need a trusted authority (TAI).

The thring signature by Yuen et al. [YLA$^+$13] is constructed using t linkable 1-out-of-N signatures (LA) based on the linkable ring signature scheme by Fujisaki [Fuj11] resulting in a non-interactive signing phase (NIS). Using the signature scheme by Boneh and Boyen [BB04], each signer signs both an event-id and a verification key of a generic one-time signature scheme with which the actual message is then signed with. The anonymity of each 1-out-of-N signature is reduced to a variant of the DDHP, the so-called Q-decisional Diffie-Hellman inversion problem. It further achieves a sub-linear signature size in $\mathcal{O}\left(t\sqrt{N}\right)$.

The unlinkable thring signature scheme by Yuen et al. [YLA$^+$11] (ULA) is an extension of the ring signature scheme by Shacham and Waters [SW07]. A signature contains a commitment for every ring member, which either equals the encryption of a signer's public key or the identity element in the case of a non-signer. Since these values belong to different (sub-)groups, ensuring anonymity requires that the subgroup decision problem is hard. The scheme's signing phase is interactive (IS), and the resulting signature size is in $\mathcal{O}(N)$.

7.2 Trends and Future Work

One observable trend is that more recent schemes are leveraging multiple building blocks to enable compact ring membership witnesses resulting in a reduced signature size independent of- or sub-linear in the ring size N. While this may not seem directly connected to the scheme's anonymity, in practice, larger signature sizes that are at least linear in N lead to the choice of smaller ring sizes, simultaneously reducing the signers' anonymity to this smaller set. Especially in ad-hoc or distributed networks, smaller signature sizes are a significant aspect. For instance, the cryptocurrency Monero, which uses a ring signature scheme based on the work of [FS07] to hide the actual sender of a transaction, has fixed the ring size to only 11 members [4].

Here, unlinkable thring signature schemes with a signature size sub-linear in N that provide unconditional anonymity would be promising, whereas such signature sizes are most likely only possible to achieve under the construction type using zero-knowledge proofs. Besides, the implementation, analysis, and possible optimization regarding the efficiency and concrete signature size of various thring signature schemes would also be interesting for future research.

Furthermore, linkable (threshold) ring signatures have gained increasing attention due to their more practical applicability in many areas such as e-voting, cryptocurrencies, or e-cash in general. They provide anonymity

[4] `https://www.getmonero.org/resources/moneropedia/ring-size.html`

for honest users, but it can be detected if parties vote more than once or try to spend their same funds multiple times. However, a downside of linkable thring signatures is that they can only provide computational anonymity, i.e., the anonymity is based on complexity assumptions. This follows from their inherent culpability addressed in Section 3.4.1; on the one hand, non-signers can prove that they are not part of the signer group by creating another signature that appears to be unlinked to the signature in question. On the other hand, the signers can prove that they have created this signature. Hence, as soon as an adversary has calculated the secret key of a signer or signer subset, the adversary can create a signature on its own and check whether this signer (subset) is indeed the creator of a given signature by checking if those signatures are linked.

As the anonymity of all linkable thring signatures based on non-generic building blocks examined in this work is reduced to the decisional Diffie-Hellman problem, it may also be suggestive to design linkable thring signatures where the anonymity is reduced to a weaker assumption. The scheme by Haque et al. [HKSS20] with concrete instantiations of the required building blocks may be such a solution.

In contrast, for the initial use-case envisaged by Rivest et al. [RST01] or Bresson et al.[BSS02], leaking secrets or whistleblowing, unlinkable (threshold) ring signatures that provide unconditional anonymity are more suitable. With unconditional anonymous thring signatures, no ring member can prove that it was not part of this exposure, and also, the signers themselves cannot readily attest to be the creators of this signature. Even if the underlying public-key cryptosystem is no longer secure such that the secret keys can be computed efficiently, e.g., through breakthroughs in quantum computing, such schemes still perfectly hide the signers.

Additionally, for thring signature schemes that are based on quantum-resistant complexity assumption while requiring the standard random oracle model, a transformation to obtain their security provable in the quantum random oracle model would be promising for the post-quantum age.

References

[ACHdM05] Giuseppe Ateniese, Jan Camenisch, Susan Hohenberger, and Breno de Medeiros. Practical group signatures without random oracles. *IACR Cryptology ePrint Archive*, 2005:385, 01 2005.

[AMCG08] Carlos Aguilar Melchor, Pierre-Louis Cayrel, and Philippe Gaborit. A new efficient threshold ring signature scheme based on coding theory. In Johannes Buchmann and Jintai Ding, editors, *Post-Quantum Cryptography*, pages 1–16, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan L. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, pages 56–73, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[BDH$^+$19] Michael Backes, Nico Döttling, Lucjan Hanzlik, Kamil Kluczniak, and Jonas Schneider. Ring signatures: Logarithmic-size, no setup—from standard assumptions. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, pages 281–311, Cham, 2019. Springer International Publishing.

[BdM94] Josh Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital signatures. In Tor Helleseth, editor, *Advances in Cryptology — EUROCRYPT '93*, pages 274–285, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

[BKM06] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, pages 60–79, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[Bon98] Dan Boneh. The decision diffie-hellman problem. In *IN THIRD ALGORITHMIC NUMBER THEORY SYMPOSIUM, LNCS 1423*, pages 48–63. Springer-Verlag, 1998.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, page 62–73, New York, NY, USA, 1993. Association for Computing Machinery.

[BS20] Dan Boneh and Victor Shoup. A graduate course in applied cryptography. *Draft 0.5*, 2020.

[BSS02] Emmanuel Bresson, Jacques Stern, and Michael Szydlo. Threshold ring signatures and applications to ad-hoc groups. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, pages 465–480, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[BW06] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 427–444. Springer, 2006.

[CCKK15] Dong Pyo Chi, Jeong Woon Choi, Jeong San Kim, and Taewan Kim. Lattice based cryptography for beginners. *IACR Cryptol. ePrint Arch.*, 2015:938, 2015.

[CDMP05] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-damgård revisited: How to construct a hash function. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, pages 430–448, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[CHGL19] Jiangshan Chen, Yupu Hu, Wen Gao, and Hong-Liang Li. Lattice-based threshold ring signature with message block sharing. *KSII Trans. Internet Inf. Syst.*, 13(2):1003–1019, 2019.

[CHY05] Sherman S. M. Chow, Lucas C. K. Hui, and S. M. Yiu. Identity based threshold ring signature. In Choon-sik Park and Seongtaek Chee, editors, *Information Security and Cryptology – ICISC 2004*, pages 218–232, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[CLRS10a] Pierre-Louis Cayrel, Richard Lindner, Markus Rückert, and Rosemberg Silva. Improved zero-knowledge identification with lattices. In Swee-Huay Heng and Kaoru Kurosawa, editors, *Provable Security*, pages 1–17, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[CLRS10b] Pierre-Louis Cayrel, Richard Lindner, Markus Rückert, and Rosemberg Silva. A lattice-based threshold ring signature scheme. In Michel Abdalla and Paulo S. L. M. Barreto, editors, *Progress in Cryptology – LATINCRYPT 2010*, pages 255–272, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[CLWY05] Sherman S. M. Chow, Joseph K. Liu, Victor K. Wei, and Tsz Hon Yuen. Ring signatures without random oracles. *IACR Cryptol. ePrint Arch.*, 2005:317, 2005.

[CM10] Pierre-Louis Cayrel and Mohammed Meziani. Post-quantum cryptography: Code-based signatures. In Tai-hoon Kim and Hojjat Adeli, editors, *Advances in Computer Science and Information Technology*, pages 82–99, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[CPS08] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. The random oracle model and the ideal cipher model are equivalent. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, pages 1–20, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[CvH91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *Advances in Cryptology — EUROCRYPT '91*, pages 257–265, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.

[DHS15] David Derler, Christian Hanser, and Daniel Slamanig. Revisiting cryptographic accumulators, additional properties and relations to other primitives. Cryptology ePrint Archive, Report 2015/087, 2015. `https://eprint.iacr.org/2015/087`.

[DS16] Yuanxi Dai and John Steinberger. Indifferentiability of 8-round feistel networks. In *Proceedings, Part I, of the 36th Annual International Cryptology Conference on Advances in Cryptology — CRYPTO 2016 - Volume 9814*, page 95–120, Berlin, Heidelberg, 2016. Springer-Verlag.

[DSKT16] Dana Dachman-Soled, Jonathan Katz, and Aishwarya Thiruvengadam. 10-round feistel is indifferentiable from an ideal cipher. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016*, pages 649–678, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[DV09] Léonard Dallot and Damien Vergnaud. Provably secure code-based threshold ring signatures. In Matthew G. Parker, editor, *Cryptography and Coding*, pages 222–235, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing,* *May 13-17, 1990, Baltimore, Maryland, USA*, pages 416–426. ACM, 1990.

[FS07] Eiichiro Fujisaki and Koutarou Suzuki. Traceable ring signature. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography – PKC 2007*, pages 181–200, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[Fuj11] Eiichiro Fujisaki. Sub-linear size traceable ring signatures without random oracles. In Aggelos Kiayias, editor, *Topics in Cryptology – CT-RSA 2011*, pages 393–415, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[GM17] Jens Groth and Mary Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable snarks. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 581–612, Cham, 2017. Springer International Publishing.

[GW18] Ke Gu and Na Wu. Constant size traceable ring signature scheme without random oracles. *IACR Cryptol. ePrint Arch.*, 2018:288, 2018.

[HIL99] Johan Håstad, Russell Impagliazzo, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28, 02 1999.

[HKSS20] Abida Haque, Stephan Krenn, Daniel Slamanig, and Christoph Striecks. Logarithmic-size (linkable) threshold ring signatures in the plain model. *IACR Cryptol. ePrint Arch*, 683:2020, 2020.

[HS20] Abida Haque and Alessandra Scafuro. Threshold ring signatures: New definitions and post-quantum security. In *IACR International Conference on Public-Key Cryptography*, pages 423–452. Springer, 2020.

[HW15] Pavel Hubáček and Daniel Wichs. On the communication complexity of secure function evaluation with long output. pages 163–172, 01 2015.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition.* Chapman & Hall/CRC, 2nd edition, 2014.

[LWW04] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups. In

Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *Information Security and Privacy*, pages 325–335, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[Lyu08] Vadim Lyubashevsky. *Towards practical lattice-based cryptography.* PhD thesis, UC San Diego, 2008.

[MBB^{+}13] Carlos Aguilar Melchor, Slim Bettaieb, Xavier Boyen, Laurent Fousse, and Philippe Gaborit. Adapting lyubashevsky's signature schemes to the ring signature setting. In *International Conference on Cryptology in Africa*, pages 1–25. Springer, 2013.

[McC90] Kevin S McCurley. The discrete logarithm problem. In *Proc. of Symp. in Applied Math*, volume 42, pages 49–74. USA, 1990.

[Men09] Alfred Menezes. An introduction to pairing-based cryptography. *Recent trends in cryptography*, 477:47–65, 2009.

[MHOY20] Alexander Munch-Hansen, C. Orlandi, and Sophia Yakoubov. Stronger notions and a more efficient construction of threshold ring signatures. *IACR Cryptol. ePrint Arch.*, 2020:678, 2020.

[MVR99] Silvio Micali, Salil Vadhan, and Michael Rabin. Verifiable random functions. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, FOCS '99, page 120, USA, 1999. IEEE Computer Society.

[OTYO18] Takeshi Okamoto, Raylin Tso, Michitomo Yamaguchi, and Eiji Okamoto. A k-out-of-n ring signature with flexible participation for signers. *IACR Cryptol. ePrint Arch.*, 2018:728, 2018.

[PBB13] Albrecht Petzoldt, Stanislav Bulygin, and Johannes Buchmann. A multivariate based threshold ring signature scheme. *Applicable Algebra in Engineering, Communication and Computing*, 24, 08 2013.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *Advances in Cryptology — ASIACRYPT 2001*, pages 552–565, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

[SOO09] Chifumi Sato, Takeshi Okamoto, and Eiji Okamoto. Strongly unforgeable id-based signatures without random oracles. In

Feng Bao, Hui Li, and Guilin Wang, editors, *Information Security Practice and Experience*, pages 35–46, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. Public-key identification schemes based on multivariate quadratic polynomials. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, pages 706–723, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[Ste94] Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *Advances in Cryptology — CRYPTO' 93*, pages 13–21, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

[SW07] Hovav Shacham and Brent Waters. Efficient ring signatures without random oracles. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography – PKC 2007*, pages 166–180, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[TWC$^+$05] Patrick P. Tsang, Victor K. Wei, Tony K. Chan, Man Ho Au, Joseph K. Liu, and Duncan S. Wong. Separable linkable threshold ring signatures. In Anne Canteaut and Kapaleeswaran Viswanathan, editors, *Progress in Cryptology - INDOCRYPT 2004*, pages 384–398, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, pages 755–784, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[vS13] Nicolas van Saberhagen. Cryptonote v 2.0. 2013.

[YLA$^+$11] Tsz Hon Yuen, Joseph K. Liu, Man Ho Au, Willy Susilo, and Jianying Zhou. Threshold ring signature without random oracles. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '11, page 261–267, New York, NY, USA, 2011. Association for Computing Machinery.

[YLA$^+$13] Tsz Hon Yuen, Joseph K. Liu, Man Ho Au, Willy Susilo, and Jianying Zhou. Efficient linkable and/or threshold ring signature without random oracles. *Comput. J.*, 56(4):407–421, 2013.

[YLTZ09] Yumin Yuan, Da Li, Liwen Tian, and Haishan Zhu. Certificateless signature scheme without random oracles. In Jong Hyuk Park, Hsiao-Hwa Chen, Mohammed Atiquzzaman, Changhoon Lee, Tai-Hoon Kim, and Sang-Soo Yeo, editors, *Advances in Information Security and Assurance, Third International Conference and Workshops, ISA 2009, Seoul, Korea, June 25-27, 2009. Proceedings*, volume 5576 of *Lecture Notes in Computer Science*, pages 31–40. Springer, 2009.

Table 1: Comparison of analyzed thring signature schemes

Scheme	NIS	TAI	ULA	Assumptions	Building Blocks	Anonymity	Size
Construction with Secret Sharing							
[BSS02]-1	No	**Yes**	**Yes**	ICM, ROM, RSA	TP	Unconditional	$\mathcal{O}(N^2)$
[CHY05]	No	No	**Yes**	ICM, ROM, CDHP	Bilinear Pairings	TA, Unconditional	$\mathcal{O}(N)$
[DV09]	No	**Yes**	**Yes**	ROM, SDP †	TP	Unconditional	$\mathcal{O}(N)$
[HS20]	No	**Yes**	**Yes**	QROM	TC‡, F‡	Trapdoor indistinguishability, F hiding	$\mathcal{O}(KmN)*$
Construction with Ring Hashing							
[BSS02]-2	No	**Yes**	**Yes**	ROM, RSA	TP, Fair Partitions	Unconditional	$\mathcal{O}(2^t N \log N)$
[LWW04]	**Yes**	**Yes**	No	ROM, DLP		DDHP	$\mathcal{O}(tN)$
[OTYO18]	**Yes**	No	**Yes**	ROM, DLP	Short-Time Keys	TA, Unconditional	$\mathcal{O}(tN)$
[CHGL19]	No	No	**Yes**	ROM, SVP †	Message Block Sharing	Unconditional	$\mathcal{O}(N)$
Construction with Zero-Knowledge Proofs							
[TWC$^+$05]	No	**Yes**	No	ROM, DDHP, Strong RSA	HVZK PoK	Strong RSA, DDHP	$\mathcal{O}(N)$
[FS07]	No	**Yes**	No	ROM, DLP	NIZK‡	DDHP, NIZK zero-knowledge	$\mathcal{O}(N)$
[AMCG08]	No	**Yes**	**Yes**	ROM, MDP †	HVZK ID-protocol	Unconditional	$\mathcal{O}(KN)*$
[CLRS10b]	No	**Yes**	**Yes**	ROM, SIS †	HVZK ID-protocol	Unconditional	$\mathcal{O}(KN)*$
[PBB13]	No	**Yes**	**Yes**	ROM, MQP †	HVZK ID-protocol, Com‡	Com statistically hiding	$\mathcal{O}(KN)*$
[MHOY20]	**Yes**	**Yes**	No	ROM, DLP, RSA, CRS	NIZKAoK‡, ACC	DDHP, NIZKAoK zero-knowledge	$\mathcal{O}(t)$
[HKSS20]	**Yes**	**Yes**	No		NIWI‡, VRF‡, SPB‡, PKE‡, F‡	VRF pseudorandom, key-privacy; SPB index-hiding; PKE key-privacy, CPA-secure; NIWI witness-indistinguishability	$\mathcal{O}(t \log(N))$

* K and m are statistical security parameters.
† Post-quantum secure problem.
‡ Generic building block.