# Logarithmic-Size (Linkable) Threshold Ring Signatures in the Plain Model

Abida Haque[1(✉)], Stephan Krenn[2], Daniel Slamanig[2], and Christoph Striecks[2]

[1] North Carolina State University, Raleigh, USA
ahaque3@ncsu.edu
[2] AIT Austrian Institute of Technology, Vienna, Austria
{stephan.krenn,daniel.slamanig,christoph.striecks}@ait.ac.at

**Abstract.** A 1-out-of-$N$ ring signature scheme, introduced by Rivest, Shamir, and Tauman-Kalai (ASIACRYPT '01), allows a signer to sign a message as part of a set of size $N$ (the so-called "ring") which are anonymous to any verifier, including other members of the ring. Threshold ring (or "thring") signatures generalize ring signatures to $t$-out-of-$N$ parties, with $t \geq 1$, who anonymously sign messages and show that they are distinct signers (Bresson et al., CRYPTO'02).

Until recently, there was no construction of ring signatures that both $(i)$ had logarithmic signature size in $N$, and $(ii)$ was secure in the plain model. The work of Backes et al. (EUROCRYPT'19) resolved both these issues. However, threshold ring signatures have their own particular problem: with a threshold $t \geq 1$, signers must often reveal their identities to the other signers as part of the signing process. This is an issue in situations where a ring member has something controversial to sign; he may feel uncomfortable requesting that other members join the threshold, as this reveals his identity.

Building on the Backes et al. template, in this work we present the first construction of a thring signature that is logarithmic-sized in $N$, in the plain model, and does not require signers to interact with each other to produce the thring signature.

We also present a linkable counterpart to our construction, which supports a fine-grained control of linkability. Moreover, our thring signatures can easily be adapted to achieve the recent notions of claimability and repudiability (Park and Sealfon, CRYPTO'19).

## 1 Introduction

Ring signatures, first introduced by Rivest, Shamir, and Tauman-Kalai [30], allow a member of a set (known as the *ring*) to anonymously sign on behalf of the ring. A verifier can check that a signature comes from one of the ring members, but cannot learn who the actual signer is, a property known as *(signer) anonymity*. Bresson, Stern, and Szydlo [6] generalized ring signatures to $t$-out-of-$N$ ring signatures (aka threshold ring signatures or *thring signatures*), in which $t \geq 1$ distinct members of an ad-hoc set participate to produce a signature.


G. Hanaoka et al. (Eds.): PKC 2022, LNCS 13178, pp. 437–467, 2022.
https://doi.org/10.1007/978-3-030-97131-1_15

In a (th)ring signature, the ring should be set-up free, i.e., members can join at will by publishing a public key. Anyone can then sign with respect to a ring assembled from available public keys. Despite this open setting, many ring signature schemes need a trusted setup and rely on heuristic assumptions (e.g.,. the random oracle). However, the most desirable setting for an ad-hoc primitive like ring signatures is the plain model. In the plain model, security is based on standard and falsifiable hardness assumptions and no trusted setup is allowed. Also, most ring signature schemes are linear in the size of the ring, which is an issue when ring sizes are large. Recently, Backes et al. (BDH$^+$ for short) [3] presented an elegant construction of the first logarithmic-sized ring signatures in the plain model.

While the issues of model and signature size appear in ring signatures (see the full version for a discussion on related previous work), thring signatures with $t > 1$ have another issue. In a thring signature, $t$-out-of-$N$ signers compute a signature with the property that any verifier can check that $t$ distinct parties signed the message without revealing exactly which $t$ members signed. While the $t$ signers are anonymous to anyone *outside their set*, these signers may need to interact to create the signature. Thus, signers are not necessarily anonymous *to each other*. Importantly, concatenating $t$ instances of 1-out-of-$N$ ring signatures does not guarantee distinct signers – the same signer may have signed $t$ times in a row.

Two works avoid interaction among the signers but have other drawbacks. First, Okamoto et al. [27] designed a linear-sized scheme in the random oracle model. Here, ring members can create a 1-out-of-$N$ ring signature themselves, while also showing that they are a new signer. Thus, a list of 1-out-of-$N$ ring signatures forms a threshold ring signature. However, their solution requires a fully trusted party who issues short-term keys to all signers. This is a strong assumption for such an ad-hoc distributed primitive. Second, Liu, Wei, and Wong [23] introduced linkable ring signatures, which allow a verifier to publicly check whether two signatures were produced by the same signer. This could be extended to produce threshold ring signatures. With a list of 1-out-of-$N$ *linkable* ring signatures on a message, the signature verification algorithm checks pairwise that no two signatures in the list are linked to the same ring member without learning the identity of the signers. This approach is generic, but only yields a *one-time* thring signature scheme.

### 1.1 Our Contribution

In this work we construct thring signatures which are: *(i)* logarithmic-sized in the number of ring members; *(ii)* in the plain model from standard assumptions; *(iii)* and non-interactive, where specifically signers need not know each other.

In more detail:

– We present and prove the first construction of thring signatures where the signature size is *logarithmic* in the number of ring members and *in the plain model*. Our construction is instantiable from falsifiable standard assumptions

without the need for the random oracle heuristic or trusted setup assumptions. Our construction is inspired by the recent results by BDH$^+$ [3] but requires novel ideas and techniques.
– We create a thring signature scheme in a setting where there is no interaction among the mutually anonymous signers. Signers need not know the other signers that participate in a threshold signing, meaning our scheme achieves *strong inter-signer anonymity*. Every signer locally computes a signature and the thring signature is just the collection of the individual signatures. Additionally, our thring signature scheme allows each signer to select their own threshold. We will discuss our solution in Sect. 1.2.

We also adapt the current model of *linkability* of ring signatures and make this model more flexible and fine-grained by using the concept of a *scope* to support *scoped linkability*. We describe scoped linkability in more detail in Sect. 5. We discuss a potential post-quantum instantiation and future directions in Sect. 6.

### 1.2 Overview of Our Techniques

To give context to our approach and techniques, we describe the approach used by BDH$^+$ [3], which is inspired by the construction of linear-size ring signatures in the plain model due to Bender, Katz, and Morselli [5].

**Outline of BDH$^+$ Approach.** In BDH$^+$, the ring is $P = (VK^1, \ldots, VK^N)$. To join the ring, a user $s \in [N]$ generates key pairs $(vk^s_\sigma, sk^s_\sigma)$ and $(pk^s, sk^s)$ of a signature scheme and a public-key encryption (PKE) scheme, respectively, and sets the verification and signing key to $VK^s := (vk^s_\sigma, pk^s)$ and $SK := (sk^s_\sigma, sk^s)$. To produce a signature for a message $m$ with respect to $R$, a signer $s$ computes a signature $\sigma$ on $m$ using $sk^s_\sigma$ and encrypts $\sigma$ under $pk^s$ resulting in a ciphertext $ct$. The signer samples a random ciphertext $ct'$ (representing another user $i$ of the ring) and generates two hashing keys $hk$ and $hk'$ of a somewhere perfectly binding (SPB) hashing scheme [28] that are *perfectly binding* at position $s$ and $i$ respectively. It computes the hash of the ring $R$ under both $hk$ and $hk'$, obtaining hash values $h$ and $h'$. SPB hashing allows the signer to collapse a ring $R$ of $N$ verification keys into a ring of just two keys and membership witnesses are of size $\mathcal{O}(\log(N))$. Finally, signer $s$ computes a perfectly sound NIWI proof $\pi$ using an OR-statement which proves that either $(hk, h)$ binds to a key $VK^s$ and that $ct$ encrypts a signature of $m$ for $VK^s$ or $(hk', h')$ binds to a key $VK^i$ and that $ct'$ encrypts a signature of $m$ for $VK^i$. A signature has the form $\Sigma = (ct, ct', hk, hk', \pi)$ and verification is straightforward.

For a non-interactive threshold variant, one needs to guarantee that a specific signer cannot contribute more than one signature to a thring signature, but at the same time keep other signatures from the same signer unlinkable. As BDH$^+$ encrypt the conventional signatures (which would identify the actual signer) for anonymity, one signer can sign repeatedly on the same message. While BDH$^+$ do have a linkable version, but as soon as a signer issues two signatures, even on different messages, they can be linked together, which contradicts the anonymity of thring signatures.

**Outline of Our Approach.** To achieve inter-signer anonymous thring signatures, we follow the BDH$^+$ template, but our approach requires novel ideas. First, instead of using a signature scheme, we use a verifiable random function (VRF) [25], inspired by the recent work by Park and Sealfon [29].[1] A VRF is a function which outputs a pseudorandom value $v$ and a proof $p$ so that given the input $m$, the values $(v, p)$ and the corresponding verification key $vk$ everyone can check correctness of the evaluation. However, an output $v$ is still pseudorandom if the proof $p$ is not known. Because the VRF yields a deterministic value $v$, using $v$ in the signature ensures distinctness of the signers. Meanwhile, we encrypt the proof $p$. Our approach now enables non-interactive thring signatures, where the signatures are a collection of single 1-out-of-$N$ ring signatures. A verifier can inspect the VRF values for inequality to determine if the signers are distinct. We need an assumption called *key collision resistance* on the VRF, which requires that if the VRF is evaluated under different (honestly generated) verification keys and the same message, the evaluations will not collide. This is a reasonable assumption. Indeed, VRF candidates such as the Dodis-Yampolskiy VRF [13] satisfy this assumption (where key-collision can be seen to be unconditional).

Suppose the only change we make to BDH$^+$ is replacing a ciphertext with a VRF evaluation $v$ and encrypted proof, and the other one with a random value $\tilde{v}$ in the VRF range with an encryption of a random value. Intuitively, anonymity holds as the values $v$ and $\tilde{v}$ are (pseudo)random and do not leak the signer's identity. Meanwhile, unforgeability is based on the unpredictability of the VRF. However, we cannot use the same proof technique as BDH$^+$. In BDH$^+$, the proof of anonymity goes by hybrid game and indistinguishably hops between two OR-clauses to switch from an encrypted signature from user $s$ to an encryption signature from user $i$. If we use this strategy in our hybrids, we end up at a point where the VRF evaluations of users $s$ and $i$ are at the same time present as values $v$ and $\tilde{v}$ in one signature. Unfortunately, this immediately gives a distinguisher as the adversary can query signatures for the same message and ring both $s$ and $i$ and in the real game one "evaluation" in each signature will be a random string, but here it finds a pair that contains two values from queried signatures at the same time. Such an event is negligible in the real game.

Thus, our second change to change the NIWI to include a third OR clause. This third clause allows us in the anonymity proof to simulate the first two clauses of the OR language and switch the witnesses in the hybrids of the anonymity proof.

Being in the plain model precludes us from using a common reference string (CRS), which would allow us to embed a simulation trapdoor for the anonymity proof. To avoid a CRS, we use the following trick. Each signer $s$ adds an extra secret key $sk_{\mathsf{F}}^s$ into her overall secret key and encrypts it, i.e., $E \leftarrow \mathsf{Enc}(pk^s, sk_{\mathsf{F}}^s; r)$, and the ciphertext is added to the public key $VK^s$. Our

[1] We note that in a concurrent and independent work in [21], Lin and Wang propose a modification of BDH$^+$ that use VRFs instead of signatures to achieve repudiability. We note that their ideas do not extend to thring signatures and thus their approach cannot be directly compared to our work.

third clause in the OR language now proves that for two non-revealed users $s$ and $i$ (i.e., $s$ and $i$ from the first two OR-clauses) in the ring, it holds that $\mathsf{F}(sk_{\mathsf{F}}^{s}) = sk_{\mathsf{F}}^{i}$, where $\mathsf{F}$ is a one-way function (OWF)[2] , i.e., the clause shows that one of the two keys is the image of the other key under $\mathsf{F}$. For honestly generated keys this relation will never be satisfied. However, in the simulation we can now set up user-keys in a way that they satisfy this relationship (without requiring a CRS). We can then use the witness for this clause of the OR proof to switch out the VRF witnesses to random.

**Our Approach to Linkable Thring Signatures.** With scoped linkability, one may control linking in a fine-grained way. An arbitrary string (the scope) used for signing allows one to link multiple signatures issued with respect to the same scope. While using the compiler by Liu et al. [23] on the linkable version of the BDH$^+$ scheme yields linkable thring signatures, it is not clear how to extend this to scoped linkability. One would need to fix the scopes beforehand and make the public keys linear in the number of used scopes. Thus, it would not be possible to support a potential unbounded number of scopes. Besides, the "tagging trick" in BDH$^+$ makes their linkable version rather involved.[3] Our linkable thring signatures support an unbounded number of scopes and are a simple modular extension of our basic thring signatures.

We get scoped linkability by adding another VRF key pair to the user's keys and use the evaluation of the VRF on the scope for linking purposes (and fixing the scope in the scheme yields the conventional notion of linkability). We extend the language of the NIWI used for the OR proof to account for this additional VRF.

We use a variant of the folklore technique of extending the language of the proof system to obtain simulation-sound NIZKs [19,31], but use VRFs instead of PRFs or signatures. The additional VRF "signs" a verification key of a strongly unforgeable one-time signature scheme and the corresponding one-time signing key signs the respective partial signature. The signature and the verification key are attached to the respective 1-out-of-$N$ ring signature.

**Claimability and Repudiability.** Recently, Park and Sealfon in [29] introduced the notions of (un-)repudiability and (un-)claimability for ring signatures and are the first to formalize such definitions. Our constructions satisfy both notions of repudiability and claimability. Details are discussed in the full version.

**Flexibility.** Okamoto et al. [27] introduced the notion of flexibility. Flexibility means ring members can sign a message themselves and add themselves to a previously computed ring signature if they wish to sign on the same message and ring. However, in [27] the new signers must cooperate with a trusted dealer to achieve this. The way we construct our threshold ring signatures also allows

[2] The restriction is that the domain and range of $\mathsf{F}$ is the same.

[3] The evaluation of their $\mathsf{JointVerify}$ algorithm which they need to prove with their NIWI, when unrolled gives 480 clauses, where each clause is a conjunction of 5 verification statements of a commitment scheme.

us to achieve flexibility in that new signers can add themselves to an already-created threshold ring signature at any time and thus the threshold $t$ can be extended dynamically (see the full version for a discussion).

**Applications.** We briefly describe some potential applications for non-interactive thring signatures with inter-signer anonymity and scoped linkability. One interesting practical application for thring signatures is to share cryptocurrency wallets that require no setup and that allow users to have a single key (even if they have multiple wallets) as discussed in [26].

Secondly, linkable ring signatures are a solution to e-voting [32]. Our scheme features scoped linkability, so signers can use the same verification key to vote for candidates in different offices. For example, votes cast under the scope 'mayor' are linkable, so that nobody can double vote for mayor. Meanwhile, votes cast for different scopes remain unlinkable, such as between scopes 'governor' and 'mayor'. As a result, thrings with scoped linkability might be a valuable tool for e-voting.

Finally, one can consider an extension of the whistleblower example from Rivest et al. [30] to the "parliament's problem". Suppose that a member of a national parliament (an MP) would like to submit a controversial bill for a law. The bill is controversial enough that the MP could lose his standing among his own party. However, if enough other members agree to the bill, it will be submitted for an official law. The MP cannot use a ring signature because another MP, wishing to attach their name, can neither add themselves nor submit a new ring signature while still showing that they are a distinct member. It would not be easy for this MP to discover other interested parties. Otherwise, a thring signature with interaction would do. The solution, then, is for the first MP to publish their bill using a thring signature with strong inter-signer anonymity. Now, he need not interact with other members, and any other MP can add themselves by contributing to the thring signature.

Due to lack of space, we defer a discussion of related work to the full version.

## 2 Preliminaries

We denote the main security parameter by $\lambda$. We write $[N] = \{1, \ldots, N\}$, and $\boldsymbol{a} = (a_1, \ldots, a_N)$. $\sqcup$ denotes disjoint union and $\triangle$ denotes symmetric difference. We denote algorithms by, e.g., $\mathsf{A}$, and write $out \leftarrow \mathsf{A}(in)$ to denote that $out$ is assigned the output of the probabilistic algorithm $\mathsf{A}$ with input $in$; Sometimes we make the used random coins $r$ explicit and write $out \leftarrow \mathsf{A}(in; r)$. A function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}$ is negligible if $\forall k \in \mathbb{N}\ \exists n_0 \in \mathbb{N}\ \forall n > n_0\ :\ \mathsf{negl}(n) \leq n^{-k}$.

For the formal definition properties of the primitives discussed below, we refer the reader to the full version.

**Non-Interactive Witness-Indistinguishable Proof Systems.** Feige and Shamir [15] first introduced witness-indistinguishable proof systems. We recap the basic notions of non-interactive witness-indistinguishable proofs (NIWIs).

Let $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{Y}$ be an *effective* relation, i.e., $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{R}$ are all efficiently computable. For $(x, \mathsf{w}) \in \mathcal{R}$, $x$ is a *statement*, and $\mathsf{w}$ is the *witness*. The language

$\mathcal{L}_{\mathcal{R}}$ is defined as all statements that have a valid witness in $\mathcal{R}$, i.e., $\mathcal{L}_{\mathcal{R}} := \{x \mid \exists \mathsf{w} : (x, \mathsf{w}) \in \mathcal{R}\}$.

**Definition 1 (Non-interactive Proof System).** *Let $\mathcal{R}$ be an effective relation and $\mathcal{L}_{\mathcal{R}}$ be the language accepted by $\mathcal{R}$. A* non-interactive proof system *for $\mathcal{L}_{\mathcal{R}}$ is a pair of algorithms* $(\mathsf{Prove}, \mathsf{Vfy})$ *where:*

- $\pi \leftarrow \mathsf{Prove}(1^\lambda, x, \mathsf{w})$. *On input a statement $x$ and a witness* $\mathsf{w}$*, outputs a proof $\pi$ or $\perp$.*
- $b \leftarrow \mathsf{Vfy}(x, \pi)$. *Given a statement $x$ and a proof $\pi$, outputs a bit $b$.*

NIWIs must satisfy the following three properties. First, *perfect completeness* guarantees that correct statements can always be successfully proven. Second, *perfect soundness* ensures that it is impossible to generate valid proofs for false statements. Finally, *witness indistinguishability* says that, given two valid witnesses for a statement, no efficient adversary can decide which witness was used to compute a proof.

Following BDH$^+$ [3], we only consider NIWIs with bounded *proof-size*. That is, if we require that for any valid proof $\pi$ generated by $\mathsf{Prove}(1^\lambda, x, \mathsf{w})$, it holds that $|\pi| \leq |C_x|\mathsf{poly}(\lambda)$ for a fixed polynomial $\mathsf{poly}(\cdot)$, where $C_x$ is the verification circuit for the statement $x$, i.e., $(x, \mathsf{w}) \in \mathcal{R}$ iff $C_x(\mathsf{w}) = 1$.

**Verifiable Random Functions.** A verifiable random function (VRF) is a pseudo-random function that enables the owner of the secret key to compute a non-interactively verifiable proof for the correctness of its output [25].

**Definition 2 (Verifiable Random Function (VRF)).** *A verifiable random function is 4-tuple* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Prove}, \mathsf{Vfy})$ *where:*

- $(vk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$. *On input the security parameter $\lambda$ in unary, this PPT algorithm outputs a public verification key $vk$ and corresponding secret key $sk$.*
- $v \leftarrow \mathsf{Eval}(sk, x)$. *On input the secret key $sk$ and an input value $x \in \{0,1\}^{a(\lambda)}$, this deterministic algorithm outputs a value $v \in \{0,1\}^{b(\lambda)}$.*
- $p \leftarrow \mathsf{Prove}(sk, x)$. *On input the secret key $sk$ and an input value $x$, this PPT algorithm outputs a proof $p$.*
- $b \leftarrow \mathsf{Vfy}(vk, x, v, p)$. *On input a verification key $vk$, an input value $x$, a value $v$, and a proof $p$, this deterministic algorithm outputs a single bit $b$.*

*Here, $a(\lambda)$ and $b(\lambda)$ are polynomially bounded and efficiently computable functions in $\lambda$.*

VRFs must satisfy the following six properties. First, *complete provability* guarantees that, if an output $v$ and a proof $p$ have been honestly computed on consistent inputs, then $p$ will verify for $v$. Second, *unique provability* ensures that for all inputs $x$, a valid proof can only be computed for a unique output value $v$. Third, *residual pseudorandomness* says that no efficient adversary that sees arbitrarily many VRF evaluations can distinguish outputs on fresh inputs from uniform. Fourth, *residual unpredictability* requires that no efficient adversary

that sees arbitrarily many VRF evaluations can compute a correct input and output pair; this is implied by residual pseudorandomness. Fifth, *key privacy* requires that no efficient adversary, only having access to an output but not the corresponding proof, can decide for which public key the output was computed. Finally, we introduce the notion of *key collision resistance* which guarantees that $\mathsf{Eval}$, on input the same message but two different secret keys, will never return the same output value. We note that all required properties are for instance satisfied by the Dodis-Yampolskiy VRF [13]. Other instantiations in the standard model have been proposed by Lysyanskaya [24] and Hofheinz and Jager [20].

**Somewhere Perfectly Binding Hashing.** Somewhere statistically binding hashes were first introduced by Hubáček and Wichs [28]. Intuitively, such schemes allow one to efficiently commit to a vector (or database). Furthermore, one can generate short openings for individual positions of the vector.

Originally, it was only required that such schemes be *statistically binding* at a single position [28]. BDH$^+$ [3] strengthened this to *perfectly* binding. Furthermore, they introduced private openings to require a secret hashing key to compute a valid opening.

As shown in [3,28] SPB hashes with private local openings in the standard model can be efficiently obtained from any 2-message private information retrieval scheme with fully efficient verifier and perfect correctness. Also, we refer to [3] for DCR and DDH based instantiations of SPB based on [28].

**Definition 3 (Somewhere Perfectly Binding (SPB) Hash).** *A* somewhere perfectly binding hash with private local opening *is a tuple of algorithms* $(\mathsf{Gen}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Vfy})$ *where:*

- $(hk, shk) \leftarrow \mathsf{Gen}(1^\lambda, n, \mathsf{i})$. *On input the security parameter* $\lambda$ *in unary, a maximum database size* $n$*, and an index* $\mathsf{i}$*, this PPT algorithm outputs public hashing key* $hk$ *and corresponding secret hashing key* $shk$*.*
- $h \leftarrow \mathsf{Hash}(hk, db)$. *On input a hashing key* $hk$ *and a database* $db$ *of size* $n$*, this deterministic algorithm outputs a hash value* $h$*.*
- $\tau \leftarrow \mathsf{Open}(hk, shk, db, j)$. *On input a public and private hashing key* $hk$ *and* $shk$*, a database* $db$*, and index* $j$*, this algorithm outputs witness* $\tau$*.*
- $b \leftarrow \mathsf{Vfy}(hk, h, j, x, \tau)$. *On input a hash key* $hk$*, a hash value* $h$*, an index* $j$*, a value* $x$ *and witness* $\tau$*, this algorithm outputs a single bit* $b$*.*

SPBs must satisfy the following three properties. First, *correctness* guarantees that for honestly generated keys, hashes, and openings, verification will allows succeed. Second, *somewhere perfectly binding* ensures that if for a specific index $\mathsf{i}$ and value $x$ verification succeeds, all valid openings on this position must open to $x$. Finally, *index hiding* says that no efficient adversary can infer the index $\mathsf{i}$ from the public hashing key.

**Definition 4 (Public Key Encryption).** *A public key encryption scheme is a triple* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *of algorithms over a message space* $M(\lambda)$*, ciphertext space* $C(\lambda)$*, and randomness space* $Rnd(\lambda)$*:*

- $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$. *On input the security parameter* $\lambda$ *in unary, this PPT algorithm computes a public key* $pk$ *and a corresponding secret key* $sk$.
- $ct \leftarrow \mathsf{Enc}(pk, m)$. *On input a public key* $pk$ *and a message* $m \in M(\lambda)$, *this PPT algorithm outputs a ciphertext* $ct$.
- $m \leftarrow \mathsf{Dec}(sk, ct)$. *On input a secret key* $sk$ *and a ciphertext* $ct$, *this deterministic algorithm outputs a message* $m$.

We require PKE schemes to satisfy the following three properties. First, *perfect correctness* guarantees that for honestly generated keys and ciphertexts, decryption will always yield the original plaintext. Second, *IND-CPA security* ensures that knowing only the public key, it is computationally infeasible to decide which message is contained in a ciphertext. Finally, *key privacy* says that no efficient adversary, not knowing the secret keys, can decide for which public key a ciphertext has been computed.

**Definition 5 (Strong One-Time Signature Scheme).** *A strong one-time signature scheme is a triple* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ *of algorithms over a message space* $M(\lambda)$*:*

- $(vk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$. *On input the security parameter* $\lambda$ *in unary, this PPT algorithm computes a verification key* $vk$ *and a corresponding signing key* $sk$.
- $\varsigma \leftarrow \mathsf{Sign}(sk, m)$. *On input a signing key* $sk$ *and a message* $m \in M(\lambda)$, *this PPT algorithm outputs a signature* $\varsigma$.
- $b \leftarrow \mathsf{Vfy}(vk, m, \varsigma)$. *On input a verification key* $vk$, *a message* $m$ *and a signature* $\varsigma$, *this deterministic algorithm outputs a single bit* $b$.

sOTS schemes must satisfy these two properties: First, *correctness* guarantees that for honestly generated keys and signatures, verification will always succeed. Second, *strong unforgeability* ensures that no efficient adversary that can obtain one signature for a given key can produce another valid signature on any message.

**Definition 6 (One-Way Function).** *A one-way function* $\mathsf{F}$ *is defined such that:*

- $y \leftarrow \mathsf{F}(1^\lambda, x)$. *On input the security parameter* $\lambda$ *in unary and an input value* $x \in \{0,1\}^\lambda$, *this deterministic algorithm computes an output* $y \in \{0,1\}^\lambda$.

OWFs must satisfy the following two properties. First, it must be *efficiently computable*, meaning that there is a polynomial-time algorithm to evaluate the function. Second, it must be *hard to invert*, so that given only an output value $y$, it is computationally infeasible to find a preimage $x^*$ mapping to this output. Note that there can be multiple $x^*$ for which $\mathsf{F}(x^*) = y$, but it is hard to find any such $x^*$. One additional requirement that we put on our OWFs is that the range must be a subset of the domain[4].

We define a concept of a *few fixed points* function. We have not seen this particular property in the literature, however a one-way function $\mathsf{F}$ is naturally

[4] A one-way permutation where the domain and range are equal can be used here.

a few fixed points function. If not, given $x \leftarrow \{0,1\}^\lambda$, it would be likely that $\mathsf{F}(x) = x$ and an adversary could find a pre-image. This property helps clarity in the unforgeability proof.

**Definition 7 (Few fixed points).** *A function $F$ is a few fixed points function if $F : \{0,1\}^* \to \{0,1\}^*$ if* $\Pr[x \leftarrow \{0,1\}^\lambda, F(x) = x] \leq \mathsf{negl}(\lambda)$.

We also introduce a lemma that helps in the unforgeability proof. We provide a lemma that shows that the probability of finding random values that happen to be pre-images in a polynomially-sized list is negligible.

**Lemma 1.** *If $\mathsf{F} : \{0,1\}^* \to \{0,1\}^*$ is a one-way function, then the probability that the process: $s = 1, \ldots, N$ $x_s \leftarrow \{0,1\}^\lambda$ generates a pair $(x_i, x_j)$ such that $\mathsf{F}(x_i) = x_j$ is negligible in $\lambda$ (where $N$ is at most $\mathsf{poly}(\lambda)$).*

*Proof.* Given a list $x_1, \ldots, x_N$, where each $x_s \leftarrow \{0,1\}^\lambda$ (with replacement) we want to find the probability that there exists $i, j \in [N]$ such that $F(x_i) = x_j$.

Because $\mathsf{F}$ is a function, for each value $x_i$, there is one image: $F(x_i) = y$. Thus, given two values $(x_i, x_j)$, which are chosen uniformly at random, the probability that $x_j = y$ is $\frac{1}{2^\lambda}$.

There are $N(N-1)$ pairs where $i \neq j$.

$$\Pr[x_i, x_j \leftarrow D, F(x_i) = x_j] \leq \mathsf{negl}'(\lambda).$$

Where if $x_i = x_j$, from the few fixed points property we know it's less than $\mathsf{negl}(\lambda)$ and if $x_i \neq x_j$ it is $\frac{1}{2^\lambda}$, which is still a negligible function in $\lambda$.

Then we look at the case where $i = j$. There are $N$ such pairs and by definition of few fixed points:

$$\Pr[x_i \leftarrow \{0,1\}^\lambda, F(x_i) = x_j] \leq \mathsf{negl}(\lambda).$$

Adding together, he overall probability of success is $T(\lambda) = \frac{N}{\mathsf{negl}'(\lambda)} + \frac{N(N-1)}{2^\lambda}$. Because $N$ is polynomial in $\lambda$, it is much smaller than $2^\lambda$. So we conclude that $T(\lambda)$ is negligible in $\lambda$ as well.

## 3 Framework and Security Definitions

### 3.1 Syntax

We extend the basic ring signature notation of Bender et al. [5] to a thring signature. The notation is summarized in Table 1. Assuming an ordering of all public keys (e.g., lexicographic), we denote the sequence of all public keys as $P$ as a *ring*. A *subring* is a subsequence $R \subseteq P$. Regardless of which members are part of the subring, we always enumerate the subring as $R = (VK^1, \ldots, VK^N)$. A set of signers is $S \subseteq [N]$, where $R[S] = \{VK^s\}_{s \in S}$. In a thring signature scheme, a set of signers $S \subseteq [N]$ signs a message $\mathsf{msg} \in \mathcal{M}$ with respect to a subring $R$. The secret keys of signers are denoted as $T$.

**Table 1.** Notation used in the algorithms.

| Symbol | Meaning |
|---|---|
| $t^s$ | Individual threshold of signer $s$[§] |
| $\boldsymbol{t}$ | $= (t^{i_1}, \ldots, t^{i_{\lvert S \rvert}})$ |
| $t_V$ | Verification threshold |
| $N$ | Number of members of the ring. Indexed by $s$. |
| $P$ | Ordered list of public keys $P = (VK^1, \ldots, VK^N)$. |
| $R$ | Subring $R \subseteq P$. |
| $S$ | Set of signers where $S \subseteq [N]$. |
| $T$ | Secret keys to signers in $S$, $T = \{sk^s\}_{s \in S}$. |
| $NS$ | Non-signers where $NS \subseteq [N]$ |
| $\mathcal{M}$ | Message space. |
| $\lambda$ | Security parameter. |

[§] By convention, indices used to distinguish between signers are written as superscripts

Each signer $s \in S$ chooses the minimum number of total signers $t^s$ they require for a valid signature, and the verifier can choose a threshold as well. The sequence of all individual signer thresholds is denoted as $\boldsymbol{t}$. We denote the verification threshold by $t_V$, e.g., $t_V \leq |\{s : t^s \leq t_V\}|$ for $t^s \in \boldsymbol{t}$. The different thresholds are there to make the scheme as general as possible. This allows for different levels of signatures in different contexts (e.g., a signer may not want her signature to be used if there's not enough support, a verifier might be a potential signer who wants to see that there's enough support before adding her own signature to the set).

For generality, our syntax also considers system parameters $pp$ generated by a $\mathsf{Setup}$ algorithm (which in our security definitions is always assumed to be honestly executed) allowing one to also model schemes requiring trusted setup in our framework. However, we stress that our instantiations given in Sect. 4 and Sect. 5 *do not require* such a $\mathsf{Setup}$ and are in the plain model.

**Definition 8 (Threshold Ring Signature Scheme).** *A threshold ring signature (thring) scheme is a* 4*-tuple of algorithms* $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$*. A subset of signers* $S$ *from ring* $P$ *signs the message* $\mathsf{msg} \in \mathcal{M}$ *with respect to a subring* $R$ *and thresholds* $\boldsymbol{t}$*.*

- $pp \leftarrow \mathsf{Setup}(1^\lambda)$*. On input the security parameter* $\lambda$ *in unary, this PPT algorithm generates public parameters pp. The public parameters are implicit input to all other algorithms and will be omitted when clear from context.*
- $(VK, SK) \leftarrow \mathsf{KGen}(pp)$*. On input the public parameters pp, this PPT algorithm generates a public verification key* $VK$ *and a corresponding secret key* $SK$ *for a signer.*
- $\sigma \leftarrow \mathsf{Sign}(\mathsf{msg}, T, R, \boldsymbol{t})$*. On input a message* $\mathsf{msg}$*, a set of secret keys* $T$*, a subring* $R$*, and a vector of individual thresholds* $\boldsymbol{t}$*, this potentially interactive PPT procedure outputs a signature* $\sigma$ *on* $\mathsf{msg}$*.*

– $b \leftarrow \mathsf{Vfy}(\mathsf{msg}, R, \sigma, t)$. *On input a message* $\mathsf{msg}$*, a subring* $R$*, a signature* $\sigma$*, and a verification threshold* $t$*, this deterministic algorithm outputs a bit* $b$*.*

### 3.2 Security Definitions

In this section, we define the security properties for a thring signature scheme: correctness, unforgeability with respect to insider corruption, and inter-signer anonymity with respect to adversarial keys. Our paper is the first feasibility result for non-interactive thrings entirely in the plain model. Because of these requirements, it does not seem easy to achieve the strongest notions of unforgeability and anonymity (as shown in [5]). Namely, we avoid malicious users to satisfy unforgeability and we have anonymity for honest users only.

We first describe a set of oracles. In our security definitions, the adversary may access these oracles in arbitrary interleaf during the corresponding experiments. All oracles have access to the following initially empty sequences or sets: $P, P_{corr}, \mathcal{L}_{\text{signers}}$, and $Q$. The first sequence $P$ is the ring, and $P_{corr} \subseteq P$ is the subset of corrupted (or malicious) members in the ring. The sequence $\mathcal{L}_{\text{signers}}$ is the triple of the signer, the public key, and the private key. The set $Q$ is the set of signing queries.

– $\mathsf{OKGen}(s)$. On input a signer $s$, this oracle first checks whether there exists $(s, \cdot, \cdot) \in \mathcal{L}$ and returns $\perp$ if so. Otherwise, it generates a fresh key pair $(VK^s, SK^s) \leftarrow \mathsf{KGen}(pp)$, adds $(s, VK^s, SK^s)$ to $\mathcal{L}_{\text{signers}}$, $VK^s$ to $P$, and returns $VK^s$ to the adversary.
– $\mathsf{OSign}(\mathsf{msg}, S, R, \boldsymbol{t})$. On input a message $\mathsf{msg}$, a list of signers $S$, a subring $R$, and a vector of individual thresholds $\boldsymbol{t}$, this oracle first checks whether $R \subseteq P$ and returns $\perp$ if this is not the case. The oracle then decomposes $S$ to $S = S_{corr} \sqcup S_{hon}$, where $S_{corr}$ denotes corrupted users (i.e., corrupted or registered by $\mathcal{A}$) and $S_{hon}$ denotes honest users. The oracle then engages in an execution of $\mathsf{Sign}(\mathsf{msg}, T, R, \boldsymbol{t})$. The oracle mimics the behavior of honest parties using the secret keys corresponding to $S_{hon}$, and the adversary participates using $S_{corr}$. For all honest signers $s$, the oracle adds $(\mathsf{msg}, R, s, t^s)$ to $Q$.
– $\mathsf{OCorr}(s)$. On input a signer $s$, if there exists $(s, VK^s, sk^s) \in \mathcal{L}_{\text{signers}}$, the oracle returns $SK^s$ to the adversary. The oracle adds $VK^s$ to $P_{corr}$.
– $\mathsf{OReg}(s, VK^s)$. On input a signer $s$ and a public key $VK^s$, the oracle checks if there exists $(s, \cdot, \cdot) \in \mathcal{L}_{\text{signers}}$ and returns $\perp$ if so. Otherwise, it adds $VK^s$ to $P_{corr}$ and $(s, VK^s, \cdot)$ to $\mathcal{L}_{\text{signers}}$.

*Correctness.* Correctness guarantees that a signature generated by sufficiently many honest users will always pass the verification algorithm. In our definition, the verification algorithm will check whether the individual thresholds are less than or equal to the verification threshold. This supports the concept of flexibility (see the full version).

**Definition 9 (Correctness).** *A thring signature scheme is* correct *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\mathsf{msg} \in \mathcal{M}$*, any subring*

**Experiment** $\mathsf{SigForge}^{\mathcal{A}}(\lambda)$
$pp \leftarrow \mathsf{Setup}(1^\lambda)$
$(\mathsf{msg}^*, \sigma^*, R^*, t^*) \leftarrow \mathcal{A}^{\mathsf{OKGen},\mathsf{OCorr},\mathsf{OSign}}(pp)$
return 1 if:
$\mathsf{Vfy}(\mathsf{msg}^*, \sigma^*, R^*, t^*) = 1$ and
$R^* \subseteq P$ and
$|U \cup (R^* \cap P_{corr})| < t^*$
where $U = \{VK^s | \exists (\mathsf{msg}^*, R^*, s, t^s) \in Q : t^s \leq t^*\}$
return 0

**Fig. 1.** Unforgeability

**Experiment** $\mathsf{Anonymity}^{\mathcal{A}}(\lambda)$
$pp \leftarrow \mathsf{Setup}(1^\lambda)$
$(\mathsf{st}, \mathsf{msg}^*, R^*, S_0^*, S_1^*, \vec{t}) \leftarrow \mathcal{A}^{\mathsf{OKGen},\mathsf{OCorr},\mathsf{OSign},\mathsf{OReg}}(pp)$
$b \leftarrow \{0,1\}$
$T^* = \{SK^s\}$ for $s \in S_b^*$
If any $s \in P_{corr}$ $\mathcal{A}$ will cooperate to create $\sigma_b$:
$\sigma_b \leftarrow \mathsf{OSign}(\mathsf{msg}^*, T^*, R^*, \vec{t})$
$b' \leftarrow \mathcal{A}^{\mathsf{OKGen},\mathsf{OCorr},\mathsf{OSign},\mathsf{OReg}}(\mathsf{st}, \sigma_b)$
where $\mathsf{OCorr}$ and $\mathsf{OSign}$ ignore queries on $S_0^* \triangle S_1^*$ on $(\mathsf{msg}^*, R^*)$
return a random bit if:
$|S_0^*| \neq |S_1^*|$, or
$S_0^* \cup S_1^* \not\subseteq R^*$, or
$(S_0^* \cap P_{corr}) \neq (S_1^* \cap P_{corr})$, or
$(\mathsf{msg}^*, R^*)$ has been signed before
return $(b = b')$

**Fig. 2.** Inter-signer anonymity

*and ring such that $R \subseteq P$ (with $|P|$ being polynomially bounded in $\lambda$), any set of signers $S \subseteq R$, any vector of individual thresholds $\boldsymbol{t} = (t^1, \ldots, t^N)$, and any verification threshold $t$ such that $t \leq |\{i : t^i \leq t\}|$, it holds that:*

$$\Pr\left[\begin{array}{l} pp \leftarrow \mathsf{Setup}(1^\lambda) \\ \{(VK^s, SK^s) \leftarrow \mathsf{KGen}(pp)\}_{s \in [|P|]} \\ T = \{SK^s\}_{s \in [|P|]} \\ \sigma \leftarrow \mathsf{Sign}(\mathsf{msg}, T, R, \boldsymbol{t}) \end{array} : \begin{array}{l} R[S] \subseteq P \implies \\ \mathsf{Vfy}(\mathsf{msg}, R, \sigma, t) = 0 \end{array}\right] = \mathsf{negl}(\lambda)$$

*The scheme is called* perfectly correct *iff* $\mathsf{negl}(\lambda) = 0$.

*Unforgeability.* Intuitively, unforgeability guarantees that an adversary who has corrupted up to $t-1$ signers will not be able to generate a valid signature for threshold $t$. More precisely, the adversary can adaptively corrupt an arbitrary number of signers and engage in the signing protocol on arbitrary messages with honest users with respect to any thresholds and subrings. The adversary finally outputs a valid message, signature, subring, and threshold $\mathsf{msg}^*$, $\sigma^*$, $R^*$, and $t^*$. The adversary wins if (1) he did not request $\mathsf{OSign}$ for too many honest parties on $\mathsf{msg}^*$ and $R^*$ for thresholds less than $t^*$, and (2) he corrupted fewer than $t^*$ members in $R$.

Note, we can tolerate corrupted but not *malicious* parties in our scheme. This is since we get inter-signer anonymity by having unique signatures. While this requirement is weaker, it is not unusual among the ring signature definitions (many schemes do not consider malicious parties). The experiment is described in Fig. 1.

**Definition 10 (Unforgeability wrt Insider Corruption).** *A thring signature scheme satisfies* unforgeability wrt insider corruption *if for all PPT adversaries $\mathcal{A}$ there is a negligible function* $\mathsf{negl}(\lambda)$ *such that* $\Pr[\mathsf{SigForge}^{\mathcal{A}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$.

*Anonymity.* Anonymity says that it is infeasible to infer from a valid signature which honest users contributed to the signature generation, or in general to link a signer across different signatures. We protect honest signers' identities even

from other signers (*inter-signer anonymity*). We can tolerate malicious keys, even in the challenge sets, so long as both sets have the same malicious parties.

In the anonymity game, the adversary has access to all the oracles. He then requests a signature on the sets $S_0^*$ or $S_1^*$. He may continue to make $\mathsf{OSign}$ and $\mathsf{OCorr}$ requests, but the oracle will not respond to queries in the set difference between $S_0^*$ and $S_1^*$. The experiment is in Fig. 2.

In our scheme, users signing the same message $\mathsf{msg}$ with respect to the same subring $R$ but potentially different thresholds are linkable among these signatures. We ensure the threshold by preventing signers from signing the same $(\mathsf{msg}, R)$ twice. Thus, in the challenge phase we require new message-ring pairs.

Moreover, a signer who has already signed with respect to a message/ring cannot sign again because the signature needs to be completely *distinct* for new users, which is not possible due to the deterministic part. Thus, if the adversary requests a signature from a specific user in the training phase, he could pinpoint from whom that signature originated. Thus, in the challenge phase we require either that the signature is different from previous signatures or that it verifies for a different message-ring pair.

In some ring signature schemes, an adversary $\mathcal{A}$ cannot identify which user a signature came from *even with knowledge of their secret key.* However, in our scheme, the signature is an output of the VRF, and $\mathcal{A}$ can learn which signer signed a message if he knows all the keys. Thus, we do not achieve this more robust definition.

We also note that the anonymity security definition does not hide the set sizes, but this is usually the case with threshold ring signatures. Unlike in other threshold ring signatures (where after the signature is created, it is not possible to *remove* signers), here the signature can be modified for a lower threshold by removing signatures from the total concatenation.

Due to how we use the VRF in our construction, we cannot achieve the strongest notion of anonymity from Bender et al. [5] (i.e., anonymity against attribution attacks/full key exposure), where the adversary sees all the random coins for generating all the honest keys. However, we do achieve anonymity with respect to adversarially chosen keys [5], which still allows an adversary to join the ring with maliciously generated keys and corrupt users. Although an adversary can de-anonymize corrupt or malicious users, our definition allows us to protect honest users' identities.

**Definition 11 (Anonymity wrt adversarial Keys).** *A threshold ring signature scheme satisfies* inter-signer anonymity with respect to adversarial keys *if for every PPT adversary $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that*

$$\left|\Pr[\mathsf{Anonymity}^{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)\,.$$

## 4 Our Construction of Threshold Ring Signatures

In this section, we provide an overview of our construction. The formal description of our threshold ring signature scheme $\mathsf{TRS}$ is in Fig. 3.

*Signing.* Suppose that $t$ members of a ring $R = (VK^1, \ldots, VK^N)$ wish to sign a message $\mathsf{msg}$. We identify a signer index $s \in [N]$. Then each signer $s$ locally evaluates the VRF using her private key on the inputs $\mathsf{msg}||R$ and $t^s||\mathsf{msg}||R$. The latter is needed because we allow each signer to choose its own threshold. The signer then encrypts the proofs of these VRF evaluations in $ct$ and $ct'$. Next, it samples two $SPB$ hashing keys $hk^s$ and $hk^i$ for $i \in [N]$ where $i \neq s$, binding at positions $s$ and $i$ respectively. Next, it calculates $h^i = \mathsf{Hash}_{SPB}(hk^i, R)$ and $h^s = \mathsf{Hash}_{SPB}(hk^s, R)$. Then $(hk^s, h^s)$ and $(hk^i, h^i)$ are commitments to $VK^s$ and $VK^i$ respectively. Finally, the signer computes a NIWI proof (discussed below). Signer $s$ then outputs its signature $\sigma^s$ as a tuple containing the VRF evaluations, the ciphertexts, hashing keys, the NIWI proof, and its individual threshold. A threshold signature is now a plain concatenation of individual signatures, i.e., $\sigma = (\sigma^1, \ldots, \sigma^t)$.

*Verification.* To verify a signature for a target threshold $t$, the verifier on $\sigma = (\sigma^1, \ldots, \sigma^t)$ checks each $\sigma^i$ for each $1 \leq i \leq t$. It checks to see if the VRF value is different than all previously verified signatures. Then the verifier will check if the NIWI verifies and whether the threshold is less than or equal to his threshold $t$. The verifier will keep track of how many valid signatures it sees in a list $\mathcal{L}_V$. At the end, if $\mathcal{L}_V$ contains at least $t$ signatures, the verifier will accept.

*NIWI.* The signer needs to show that one of the following claims is true:

*(i)* The computations are correct for signer $s$, i.e., $hk^s$ is binding at position $s$ and commits to $R$, $VK^s$ and the corresponding secret key was used to evaluate the VRF on $\mathsf{msg}||R$ and $t^s||\mathsf{msg}||R$ resulting in $v$ and $v'$, and the corresponding proofs have been encrypted as $ct$ and $ct'$ under $pk^s_\dagger$. This is the branch for which an honest signer has all necessary keys; OR

*(ii)* the same computations have been performed correctly for signer $i$; OR

*(iii)* the secret keys $sk_\mathsf{F}$ of signer $s$ and $i$ satisfy $\mathsf{F}(sk^s_\mathsf{F}) = sk^i_\mathsf{F}$, that $hk^s$ and $hk^i$ have been computed for positions $s$ and $i$, and that the $sk_\mathsf{F}$ are those corresponding to the public keys of $s$ and $i$. As discussed in Sect. 1.2 this is needed in the anonymity proof (as publishing the VRF evaluations in the plain does no longer work with the technique of [3]) but will never be satisfied for honest keys.

More formally, we denote this language as: $\mathcal{L}' := \mathcal{L}_{\mathcal{R}|_{VK}} \vee \mathcal{L}_{\mathcal{R}|_{VK'}} \vee \mathcal{L}_F$, where $\mathcal{R}|_{VK}$ indicates the following relation $\mathcal{R}$ for a specific key $VK^s = (vk^s, pk^s_\dagger, pk^s_\ddagger, E_s)$. Statements and witnesses have the form:

$$\begin{aligned} \mathcal{R}|_{VK} : x &= (\mathsf{msg}, R, t^s, v, v', ct, ct', hk^s, h^s) \\ \mathsf{w} &= (VK^s, s, p, p', r_{ct}, r_{ct'}, \tau) \\ \mathcal{R}_F : x &= (R, h^s, h^i, hk^s, hk^i) \\ \mathsf{w} &= (s, i, VK^s, VK^i, \tau^s, \tau^i, sk^s_\mathsf{F}, sk^i_\mathsf{F}, r_{E_s}, r_{E_i}) \end{aligned}$$

The relations are then defined as follows:

$(x, \mathsf{w}) \in \mathcal{R}|_{VK}$ if and only if:

$\mathsf{Vfy}_{SPB}(hk^s, h^s, s, VK^s, \tau) = 1 \wedge$
$\mathsf{Enc}_{PKE}(pk^s_\dagger, p; r_{ct}) = ct \wedge$
$\mathsf{Enc}_{PKE}(pk^s_\dagger, p'; r_{ct'}) = ct' \wedge$
$\mathsf{Vfy}_{VRF}(vk^s, \mathsf{msg}||R, v, p) = 1 \wedge$
$\mathsf{Vfy}_{VRF}(vk^s, t^s||\mathsf{msg}||R, v', p') = 1$

$(x, \mathsf{w}) \in \mathcal{R}_F$ if and only if:

$\mathsf{F}(sk^s_\mathsf{F}) = sk^i_\mathsf{F} \wedge$
$\mathsf{Enc}_{PKE}(pk^s_\ddagger, sk^s_\mathsf{F}; r_{E_s}) = E_s \wedge$
$\mathsf{Enc}_{PKE}(pk^i_\ddagger, sk^i_\mathsf{F}; r_{E_i}) = E_i \wedge$
$\mathsf{Vfy}_{SPB}(hk^s, h^s, s, VK^s, \tau^s) = 1 \wedge$
$\mathsf{Vfy}_{SPB}(hk^i, h^i, i, VK^i, \tau^i) = 1$

Note that an honest signer does not use $(pk_\ddagger, sk_\ddagger)$ for the NIWI proof. As mentioned above, our final language $\mathcal{L}'$ is the OR of two $\mathcal{L}_{\mathcal{R}|_{VK}}$ and $\mathcal{L}_F$. Statements and witnesses for $\mathcal{L}'$ are of the form:

$$x = \begin{pmatrix} \mathsf{msg} & R & v & v' & ct & ct' \\ t & h^s & h^i & hk^s & hk^i & \end{pmatrix} \qquad \mathsf{w} = \begin{pmatrix} VK^s & VK^i & s & i & \tau^s & \tau^i \\ p & p' & sk^s_\mathsf{F} & sk^i_\mathsf{F} & & \\ r_{ct} & r_{ct'} & r_{E_s} & r_{E_i} & & \end{pmatrix}$$

As our instantiation does not rely on any trusted setup, there is no need for $\mathsf{Setup}$ to generate joint parameters. The verifier must know a description of the ring to check the signature. The input of the VRF includes the description of the ring. Thus, it seems that the ring signature must always be linear in the size of the ring. However, if the verifier knows the ring beforehand, the signature need not include the ring description. Alternatively, we can change the domain of the VRF to compute on the hash of the ring instead (this was also noted by Park and Sealfon [29]). For simplicity, we include the ring as input everywhere in our scheme.

### 4.1 Security of Our Construction

In this section, we provide the formal proofs of each property correctness, unforgeability, and anonymity for the construction $\mathsf{TRS}$.

**Theorem 1 (Correctness).** *If the underlying NIWI, VRF, PKE, and SPB schemes are correct, and the VRF is key collision free,* $\mathsf{TRS}$ *is correct.*

*Proof.* By construction, all individual signatures are valid. Also, we require that $\sigma^i.v \neq \sigma^j.v$ for all $i \neq j$, which follows directly as otherwise we would have that $\mathsf{Eval}_{VRF}(sk^i, \mathsf{msg}||R) = \mathsf{Eval}_{VRF}(sk^j, \mathsf{msg}||R)$ contradicting to the assumed key collision freeness. □

**Theorem 2 (Unforgeability).** *If* $\mathsf{F}$ *is a one-way function,* VRF *has residual unpredictability and unique provability,* NIWI *has perfect soundness, SPB is somewhere perfectly binding and PKE is perfectly correct, then* $\mathsf{TRS}$ *is unforgeable.*

To prove unforgeability, we need to show that a forger $\mathcal{F}$ who knows up to $t-1$ secret keys cannot forge a signature that verifies for $t$ signers. At a high level, because $\mathcal{F}$ needs to provide a valid NIWI proof in the signature, and the NIWI is perfectly sound, the claimed statement must indeed be true. We do not give $\mathcal{F}$ access to the $\mathsf{OReg}$ oracle so it is not possible for it to produce keys

**Key Generation** $\mathsf{Gen}(1^\lambda)$:

$(vk, sk) \leftarrow \mathsf{Gen}_{VRF}(1^\lambda)$;
$(pk_\dagger, sk_\dagger) \leftarrow \mathsf{Gen}_{PKE}(1^\lambda)$;
$(pk_\ddagger, sk_\ddagger) \leftarrow \mathsf{Gen}_{PKE}(1^\lambda)$;
$sk_{\mathsf{F}} \leftarrow \{0,1\}^\lambda$;
$r_E \leftarrow Rnd_{PKE}$;
$E \leftarrow \mathsf{Enc}_{PKE}(pk_\ddagger, sk_{\mathsf{F}}; r_E)$;
$VK := (vk, pk_\dagger, pk_\ddagger, E)$;
$SK := (sk, sk_\dagger, sk_\ddagger, r_E, VK)$;
**return** $(VK, SK)$.

**Verification** $\mathsf{Vfy}(\mathsf{msg}, R, \sigma, t_V)$:

// Parse each signature in the list;
$\sigma = ((v, v', ct, ct', hk^s, hk^i, \pi, t^i))_{i=1}^t$;
Sort list by $t^i$;
**for** $i \in [t]$
  $h' := \mathsf{Hash}_{SPB}(hk^s, R)$;
  $h'' := \mathsf{Hash}_{SPB}(hk^i, R)$;
  $x := (\mathsf{msg}, R, v, v', ct, ct', h', h'', hk^s, hk^i)$;
  $b' \leftarrow \mathsf{Vfy}_{NIWI}(x, \pi)$;
  **if** $b' = 1 \ \wedge \ \sigma^i.v \neq \sigma^k.v \ \forall k \in [i-1]$
    $\mathcal{L}_V.\mathsf{append}(t^i)$;
**endfor**
**if** $\exists i \geq t_V : \ \mathcal{L}_V[i] \leq i$ **return** 1;
**return** 0.

**Threshold Signing** $\mathsf{Sign}(\mathsf{msg}, T, R, \vec{t})$:

// Note $T = \{SK^s\}$
// Every signer $s$ signs by themselves
// Look at algorithm per each signer $s$.
$v \leftarrow \mathsf{Eval}_{VRF}(sk^s, \mathsf{msg}||R)$;
$p \leftarrow \mathsf{Prove}_{VRF}(sk^s, \mathsf{msg}||R)$;
$v' \leftarrow \mathsf{Eval}_{VRF}(sk^s, t^s||\mathsf{msg}||R)$;
$p' \leftarrow \mathsf{Prove}_{VRF}(sk^s, t^s||\mathsf{msg}||R)$;
$r_{ct}, r_{ct'} \leftarrow Rnd_{PKE}$;
$ct \leftarrow \mathsf{Enc}_{PKE}(pk^s_\dagger, p; r_{ct})$;
$ct' \leftarrow \mathsf{Enc}_{PKE}(pk^s_\dagger, p'; r_{ct'})$;
$(hk^s, shk^s) \leftarrow \mathsf{Gen}_{SPB}(1^\lambda, N, s)$;
$h^s \leftarrow \mathsf{Hash}_{SPB}(hk^s, R)$;
$\tau^s \leftarrow \mathsf{Open}_{SPB}(hk^s, shk^s, R, s)$;
// Pick other ring member $i \neq s$
$i \in [N] \setminus s$;
$r_{E_s}, r_{E_i} \leftarrow Rnd_{PKE}$;
$(hk^i, shk^i) \leftarrow \mathsf{Gen}_{SPB}(1^\lambda, N, i)$;
$h^i := \mathsf{Hash}_{SPB}(hk^i, R)$;
$\tau^i \leftarrow \mathsf{Open}_{SPB}(hk^i, shk^i, R, i)$;
// Call on the NIWI for language $\mathcal{L}'$
$\pi \leftarrow \mathsf{Prove}_{NIWI}(x, \mathsf{w})$
$\sigma^s := (v, v', ct, ct', hk^s, hk^i, \pi, t^s)$;
// Every signer $s$ broadcasts the signature
**broadcast** $\sigma^s$;
// Final threshold ring signature under set $T$
**return** $\sigma = \{\sigma^j\}_{j=1}^t$.

**Fig. 3.** Our threshold ring signature scheme. For notation refer to Table 1.

which satisfy $\mathcal{R}_{\mathsf{F}}$. $\mathcal{F}$ can, however, corrupt honest users. One possible winning strategy is for $\mathcal{F}$ to find a pair $sk^i_{\mathsf{F}}, sk^j_{\mathsf{F}}$ such that $\mathsf{F}(sk^i_{\mathsf{F}}) = sk^j_{\mathsf{F}}$. Because $\mathsf{F}$ is one-way, such a pair will exist with only negligible probability in the set of users (by Lemma 1).

Then for the other strategy, as $\mathcal{F}$ needs to hold a witness to either $\mathcal{R}|_{VK}$ (or $\mathcal{R}|_{VK'}$) due to the somewhere perfect binding property of the SPB, the forgery must have used the identity of a signer who is a member of the ring. Due to the perfect correctness of the PKE scheme, $ct$ (or $ct'$) contains a valid proof for the respective VRF. Due to the unique provability of the VRF we know that an uncorrupted signer must have generated such a value. Finally, we can reduce unforgeability to the residual unpredictability of the VRF.

*Proof.* We prove unforgeability via hybrid arguments and a reduction to residual unforgeability.

**$\mathcal{H}_0$ to $\mathcal{H}_1$:** $\mathcal{H}_0$ is the unforgeability experiment $\mathsf{SigForge}$ from Fig. 1. In $\mathcal{H}_1$, the challenger will pick one index $i^*$ ahead of time. We abort on an $\mathsf{OCorr}$ request of $i^*$. When the adversary provides a forgery, he must have used honest keys

(created via $\mathsf{OKGen}$ queries) in his chosen ring. Since $i^*$ was picked at random, $i^*$ is in the forgery's ring with probability at least $\frac{1}{q_{KG}}$ with $q_{KG}$ the number of users generated by $\mathsf{OKGen}$. Suppose there is an adversary $\mathcal{F}$ who forges in $\mathcal{H}_0$ with some probability. Then $\mathcal{F}$ can forge in $\mathcal{H}_1$ with the same probability (except with a loss of $\frac{1}{q_{KG}}$).

**$\mathcal{H}_1$ to $\mathcal{H}_2$:** All keys $sk_{\mathsf{F}}$ generated via $\mathsf{OKGen}$ are chosen in a way that for none of the pairs $(sk_{\mathsf{F}}^j, sk_{\mathsf{F}}^k)$ it holds that either $sk_{\mathsf{F}}^j = \mathsf{F}(sk_{\mathsf{F}}^k)$ or $sk_{\mathsf{F}}^k = \mathsf{F}(sk_{\mathsf{F}}^j)$.

Such a pair existed in $\mathcal{H}_1$ with only negligible probability (Lemma 1), thus the probability of distinguishing between $\mathcal{H}_1$ and $\mathcal{H}_2$ is negligible as well. As the $sk_{\mathsf{F}}^i$ are chosen uniformly at random in the original game, here $\mathsf{OKGen}$ need only re-sample at random if a collision is found.

Now $\mathcal{R}_{\mathsf{F}}$ can never be satisfied among all the honest keys. In $\mathcal{H}_2$, the forgery $\sigma^* = \{(v, v', ct, ct', hk^s, hk^i, \pi, t^i)\}_{i=1}^t$ needs to use a witness for $\mathcal{R}|_{VK}$ or $\mathcal{R}|_{VK'}$. Due to symmetry of these both cases let us w.l.o.g. assume that $\mathcal{F}$ uses a witness for $\mathcal{R}|_{VK}$. Now by the perfect soundness of the NIWI we know that

$$(\mathsf{msg}, R, t, v, v', ct, ct', hk^{i_0}, h^{i_0}) \in \mathcal{L}|_{VK}.$$

As the SPB is somewhere perfectly binding, we have that $h^{i_0} = \mathsf{Hash}(hk^{i_0}, R)$ and $\mathsf{Vfy}_{SPB}(hk^{i_0}, h^{i_0}, i_0, VK^{s_0}, \tau^{i_0}) = 1$ implies that $R[i_0] = VK^{i_0}$. If we have $i_0 = i^*$, due to the perfect correctness of $PKE$ we have that $(pk_\dagger^{i^*}, sk_\dagger^{i*})$ are correct for all messages. Then for $p := \mathsf{Dec}(sk_\dagger^{i*}, ct)$ and $p' := \mathsf{Dec}(sk_\dagger^{i*}, ct')$ the VRF verifications $\mathsf{Vfy}_{VRF}(vk^{i^*}, \mathsf{msg}^*||R^*, v, p) = 1$ and $\mathsf{Vfy}_{VRF}(vk^{i^*}, t^*||\mathsf{msg}^*||R^*, v', p') = 1$. Finally, due to the unique provability of the VRF we know that the values $(v, p)$ and $(v', p')$ are the unique pairs under $vk^{i^*}$ corresponding to inputs $\mathsf{msg}^*||R^*$ and $t^*||\mathsf{msg}^*||R^*$.

**Reduction to Residual Unpredictability.**

We present a reduction $\mathcal{A}$ to the residual unpredictability of the VRF, which uses $\mathcal{F}$ (as in $\mathcal{H}_2$) as a subroutine.

- $\mathcal{A}$ from challenger $\mathcal{C}_{VRF}$ receives $vk$ which we embed into $VK^{i^*}$.
- On each request from $\mathcal{F}$: $\mathsf{OKGen}$, $\mathsf{OCorr}$, $\mathsf{OSign}$ are as in $\mathcal{H}_2$. But for each VRF evaluation at $i^*$ $\mathcal{A}$ queries $\mathcal{C}_{VRF}^{\mathsf{OEval}(sk,\cdot)}$. Remember that if $\mathcal{F}$ requests $\mathsf{OCorr}$ on either $i^*$ then Abort.
- From a valid forgery $\sigma^*$ of $\mathcal{F}$, we obtain $p$ and $p'$ for inputs $\mathsf{msg}^*||R^*$ and $t^*||\mathsf{msg}^*||R^*$.
- Output one of $(\mathsf{msg}^*||R^*, v)$ and $(t^*||\mathsf{msg}^*||R^*, v')$ as forgery to $\mathcal{C}_{VRF}$.

If $\mathcal{F}$ made a valid forgery, then with probability $\frac{1}{q_{KG}}$ he picked the index $i^*$. Because the relationship for $\mathsf{F}$ does not hold, the NIWI has perfect soudness, the SPB is somewhere perfectly binding, and the PKE is perfectly correct, $\mathcal{F}$ can make a valid forgery by violating the residual unpredictability of the VRF. Consequently, we can just forward all VRF evaluations for this key in calls to the $\mathsf{OSign}$ oracle to the challenger of the VRF and given that the winning condition for the forgery output by the thrings forger are valid, we need to have a fresh evaluation of the VRF, which allows $\mathcal{A}$ to break residual unpredictability.

As $\mathcal{A}$ can break residual unpredictability with at most negligible probability, we see that $\mathcal{F}$ can win in $\mathcal{H}_2$ with at most negligible probability as well. By hybrid argument, we see that $\mathcal{F}$ cannot win in $\mathcal{H}_0$ either except with negligible probability.

□

**Theorem 3 (Anonymity).** *If SPB is index hiding, PKE has key-privacy and CPA-security,* NIWI *is computationally witness-indistinguishable, and* VRF *has residual pseudorandomness and key-privacy then* TRS *is anonymous.*

Recall the anonymity experiment in Fig. 2. In the training phase, the adversary $\mathcal{A}_{anon}$ queries on OKGen, OSign, OCorr, and OReg. Then in the challenge phase, $\mathcal{A}_{anon}$ submits a message msg, a subring $R$, and two signing sets $S_0, S_1 \subset R$. The challenger picks one of the signing sets $S_b$ and computes a signature $\sigma$. On $\sigma$, $\mathcal{A}_{anon}$ guesses which of $S_0, S_1$ signed the message.

For us, a $t$-out-of-$N$ thring signature is a collection of $t$ ring signatures, and signatures are independent of each other. Thus, it suffices to show anonymity for a single signer and one can use a hybrid argument to show anonymity for larger thresholds. The probability of distinguishing between two sets of signatures is negligible if distinguishing between two signatures is negligible. Then for the challenge phase, $\mathcal{A}_{anon}$ will produce two indices $s_0, s_1$, message msg, and ring $R$.

Over a sequence of hybrids, we transform the signature element by element from one under $s_0$ to a signature under $s_1$. By showing that each hybrid is computationally indistinguishable from its predecessor, we see that signatures under $s_0$ and $s_1$ are indistinguishable to $\mathcal{A}_{anon}$.

We make changes over the hybrids and justify them in the proofs by using the following properties: *(i)* Changes to $hk$: the SPB is index-hiding. *(ii)* Changes to $ct$: the PKE has key-privacy and CPA-security. *(iii)* Changes to the witness used for $\pi$: the NIWI is computationally witness-indistinguishable. *(iv)* Changes to the value $v$: the VRF has residual pseudorandomness.

*Proof.* Consider the following hybrids:

**$\mathcal{H}_0$ to $\mathcal{H}_1$:** $\mathcal{H}_0$ is the anonymity experiment in Fig. 2 with challenge bit $b = 0$. The challenger knows ahead of time $q_{KG}$, the number of queries $\mathcal{A}_{anon}$ will make to OKGen and picks two indices $\mathsf{i}_0, \mathsf{i}_1 \leftarrow [q_{KG}]$ ($\mathsf{i}_0 \neq \mathsf{i}_1$). If on either $\mathsf{i}_0, \mathsf{i}_1$, $\mathcal{A}_{anon}$ requests OCorr (or chooses these for OReg) then ABORT. Finally, we require that $\mathcal{A}_{anon}$ picks indices $\mathsf{i}_0, \mathsf{i}_1$ equal to the two indices the challenger picked ahead of time. Because $\mathsf{i}_0, \mathsf{i}_1$ were picked randomly, with $\frac{1}{q_{KG}}$ probability these will be the right two indices. An adversary playing in $\mathcal{H}_1$ wins with the same probability as in $\mathcal{H}_0$, except for a multiplicative loss of $\frac{1}{(q_{KG})^2}$.

**$\mathcal{H}_1$ to $\mathcal{H}_2$:** In this step, the challenger always chooses $\mathsf{i}_1$ as the 'other index' when computing the final challenge signature. As $\mathsf{i}_1$ was uniformly random, this is indistinguishable.

**$\mathcal{H}_2$ to $\mathcal{H}_3$:** In this step, for OKGen on $\mathsf{i}_0, \mathsf{i}_1$ make sure the secret keys $sk_{\mathsf{F}}^{\mathsf{i}_0}$ and $sk_{\mathsf{F}}^{\mathsf{i}_1}$ are such that $\mathsf{F}(sk_{\mathsf{F}}^{\mathsf{i}_0}) = sk_{\mathsf{F}}^{\mathsf{i}_1}$ holds. This change affects only $sk_{\mathsf{F}}^{\mathsf{i}_0}$ and $sk_{\mathsf{F}}^{\mathsf{i}_1}$, which are hidden in $E_{\mathsf{i}_0}$ and $E_{\mathsf{i}_1}$ and are never revealed.

$\mathcal{H}_3$ **to** $\mathcal{H}_4$**:** Calculate $(v^1, p^1) \leftarrow (\mathsf{Eval}_{VRF}(sk^{\mathsf{i}_1}, \mathsf{msg}||R), \mathsf{Prove}_{VRF}(sk^{\mathsf{i}_1}, \mathsf{msg}||R))$ and $(v'^1, p'^1) \leftarrow (\mathsf{Eval}_{VRF}(sk^{\mathsf{i}_1}, t||\mathsf{msg}||R), \mathsf{Prove}_{VRF}(sk^{\mathsf{i}_1}, t||\mathsf{msg}||R))$,
and $\tau^{\mathsf{i}_1} = \mathsf{Open}_{SPB}(hk^{\mathsf{i}_1}, shk^{\mathsf{i}_1}, R, \mathsf{i}_1)$. Then change the witness $\mathsf{w}$:

$$\mathcal{H}_3 \quad \mathsf{w}^0 = (VK^{\mathsf{i}_0}, VK^{\mathsf{i}_1}, \mathsf{i}_0, \mathsf{i}_1, \tau^{\mathsf{i}_0}, \tau', p, p', sk_{\mathsf{F}}^{\mathsf{i}_0}, sk_{\mathsf{F}}', r_{ct}, r_{ct'}, r_{E_0}, r_{E_1})$$

$$\mathcal{H}_4 \ \widehat{\mathsf{w}}^0 = (VK^{\mathsf{i}_0}, VK^{\mathsf{i}_1}, \mathsf{i}_0, \mathsf{i}_1, \tau^{\mathsf{i}_0}, \tau^{\mathsf{i}_1}, p^1, p'^1, sk_{\mathsf{F}}^{\mathsf{i}_0}, sk_{\mathsf{F}}^{\mathsf{i}_1}, r_{ct}, r_{ct'}, r_{E_0}, r_{E_1})$$

Note that this only makes changes in the *witness* of the NIWI. The values in the signature are only changed in subsequent games. Since the NIWI is witness indistinguishable, these changes are indistinguishable to any adversary. We construct $\mathcal{A}_{WI}$ which uses $\mathcal{A}_{anon}$.

1. $\mathcal{A}_{WI}$ activates $\mathcal{A}_{anon}$. He chooses $\mathsf{i}_0, \mathsf{i}_1$.
2. For each query, he answers as the challenger would.
3. On a challenge $(s_0, s_1, \mathsf{msg}, R)$, if $s_0 = \mathsf{i}_0$ and $s_1 = \mathsf{i}_1$, he calculates $\mathsf{w}^0, \mathsf{w}^0$ as above and sends to the challenger. He gets back $\pi^*$.
4. $\mathcal{A}_{WI}$ forwards $\pi^*$ as part of the signature to $\mathcal{A}_{anon}$.
5. $\mathcal{A}_{WI}$ outputs the same as $\mathcal{A}_{anon}$.

In $\mathcal{H}_3$, the witness is $\mathsf{w}_0$, and in $\mathcal{H}_4$, it is $\widehat{\mathsf{w}}^0$. Then if $\mathcal{A}_{anon}$ wins $\mathcal{H}_3$ and $\mathcal{H}_4$ with different probabilities, then $\mathcal{A}_{WI}$ can win the witness-indistinguishability game with the same probability. Thus, $\mathcal{H}_3$ and $\mathcal{H}_4$ are indistinguishable.

$\mathcal{H}_4$ **to** $\mathcal{H}_5$**:** $ct := \mathsf{Enc}_{PKE}(pk_{\dagger}^{\mathsf{i}_0}, p; r_{ct}) \rightarrow ct^1 := \mathsf{Enc}_{PKE}(pk_{\dagger}^{\mathsf{i}_1}, p; r_{ct})$

To show that this change is indistinguishable, we construct an adversary to PKE key privacy $\mathcal{A}_{KP}^{PKE}$.

1. $\mathcal{A}_{KP}^{PKE}$ receives two public keys $pk^0, pk^1$ from his challenger.
2. $\mathcal{A}_{KP}^{PKE}$ activates $\mathcal{A}_{anon}$. He picks $\mathsf{i}_0, \mathsf{i}_1$. $\mathcal{A}_{KP}^{PKE}$ answers every query as the challenger would have done, except for $\mathsf{KGen}$ at $\mathsf{i}_0$ and $\mathsf{i}_1$, where he gives $pk^{\mathsf{i}_0} = pk^0$ and $pk^{\mathsf{i}_1} = pk^1$.
3. Finally, $\mathcal{A}_{anon}$ will request a signature on $\mathsf{msg}, R$.
4. $\mathcal{A}_{KP}^{PKE}$ computes using $sk^{\mathsf{i}_0}$, $p \leftarrow \mathsf{Prove}_{NIWI}(sk^{\mathsf{i}_0}, \mathsf{msg}||R)$. He sends $p$ to his challenger.
5. The challenger will pick $b \leftarrow \{0, 1\}$. If $b = 0$, returns $ct^* = ct$ and if $b = 1$, returns $ct^* = ct^1$.
6. $\mathcal{A}_{KP}^{PKE}$ uses $ct^*$ for the signature he gives $\mathcal{A}_{anon}$.
7. Output the same as $\mathcal{A}_{anon}$.

If the challenger picks $pk^0$, this is the anonymity game as in $\mathcal{H}_4$, but if he picks $pk^1$ then this is the game as in $\mathcal{H}_5$. Thus, if $\mathcal{A}_{anon}$ wins $\mathcal{H}_4$ and $\mathcal{H}_5$ with different probabilities, then this is the advantage of $\mathcal{A}_{KP}^{PKE}$ winning the PKE key privacy experiment.

$\mathcal{H}_5$ **to** $\mathcal{H}_6$**:** $ct' := \mathsf{Enc}_{PKE}(pk_{\dagger}^{\mathsf{i}_0}, p'; r_{ct'}) \rightarrow ct'^1 := \mathsf{Enc}_{PKE}(pk_{\dagger}^{\mathsf{i}_1}, p'; r_{ct'})$

The argument is identical to the transition from $\mathcal{H}_4$ to $\mathcal{H}_5$ with a reduction to PKE key privacy. The only difference is that $p' \leftarrow \mathsf{Prove}_{VRF}(sk^{\mathsf{i}_0}, t||\mathsf{msg}||R)$ and thus we omit details.

$\mathcal{H}_6$ **to** $\mathcal{H}_7$**:** $ct^1 := \mathsf{Enc}_{PKE}(pk_\dagger^{\mathsf{i}_1}, p; r_{ct}) \rightarrow \widehat{ct}^1 := \mathsf{Enc}_{PKE}(pk_\dagger^{\mathsf{i}_1}, p^1; r_{ct})$, where $p^1 \leftarrow \mathsf{Prove}_{VRF}(sk^{\mathsf{i}_1}, \mathsf{msg}||R)$.

We construct $\mathcal{A}_{CPA}$ which uses $\mathcal{A}_{anon}$ as a subroutine to break CPA security.

1. $\mathcal{A}_{CPA}$ receives $pk$. $\mathcal{A}_{CPA}$ picks $\mathsf{i}_0, \mathsf{i}_1$, activates $\mathcal{A}_{anon}$.
2. For each query by $\mathcal{A}_{anon}$, $\mathcal{A}_{CPA}$ answers as a challenger would, except for $\mathsf{OKGen}$ at $\mathsf{i}_0$, where he gives $pk_\dagger^{\mathsf{i}_0} = pk$.
3. When $\mathcal{A}_{anon}$ queries on $(s_0, s_1, \mathsf{msg}, R)$ then $\mathcal{A}_{CPA}$ calculates both $p \leftarrow \mathsf{Eval}_{VRF}(sk^{\mathsf{i}_0}, \mathsf{msg}||R)$ and $p^1 \leftarrow \mathsf{Eval}_{VRF}(sk^{\mathsf{i}_1}, \mathsf{msg}||R)$. He gives $p, p^1$ to his challenger.
4. Challenger flips $b \leftarrow \{0, 1\}$. If $b = 0$ he encrypts $p$, if $b = 1$ he encrypts $p^1$. He returns $ct^*$ to $\mathcal{A}_{CPA}$.
5. $\mathcal{A}_{CPA}$ uses $ct^*$ for the signature he gives to $\mathcal{A}_{anon}$.
6. $\mathcal{A}_{CPA}$ outputs the same as $\mathcal{A}_{anon}$.

If the challenger picks $p$, we are in $\mathcal{H}_6$, if $p^1$ then $\mathcal{H}_7$. Thus, $\mathcal{A}_{CPA}$ wins the CPA-security experiment with the same advantage as the difference of $\mathcal{A}_{anon}$ winning in $\mathcal{H}_6$ versus winning in $\mathcal{H}_7$.

$\mathcal{H}_7$ **to** $\mathcal{H}_8$**:** $ct'^1 := \mathsf{Enc}_{PKE}(pk_\dagger^{\mathsf{i}_1}, p'; r_{ct}) \rightarrow \widehat{ct}'^1 := \mathsf{Enc}_{PKE}(pk_\dagger^{\mathsf{i}_1}, p'^1; r_{ct})$, where $p'^1 \leftarrow \mathsf{Prove}_{VRF}(sk^{\mathsf{i}_1}, t||\mathsf{msg}||R)$. The argument is identical to the transition from $\mathcal{H}_6$ to $\mathcal{H}_7$ and thus we omit details.

$$\mathcal{H}_8 \textbf{ to } \mathcal{H}_9\text{: } \sigma = (v, v', \widehat{ct}^1, \widehat{ct}'^1, hk^{\mathsf{i}_0}, hk^{\mathsf{i}_1}, \pi, t) \rightarrow$$
$$\sigma = (v^1, v', \widehat{ct}^1, \widehat{ct}'^1, hk^{\mathsf{i}_0}, hk^{\mathsf{i}_1}, \pi, t)$$

where $v^1 \leftarrow \mathsf{Eval}_{VRF}(sk^{\mathsf{i}_1}, \mathsf{msg}||P)$.

We show that the change between $\mathcal{H}_8$ and $\mathcal{H}_9$ is indistinguishable using the following reduction to the VRF key privacy.

1. $\mathcal{A}_{KP}^{VRF}$ gets a $vk^0, vk^1$ from his challenger.
2. $\mathcal{A}_{KP}^{VRF}$ picks $\mathsf{i}_0, \mathsf{i}_1$ and activates $\mathcal{A}_{anon}$.
3. $\mathcal{A}_{KP}^{VRF}$ answers every query from $\mathcal{A}_{anon}$. At index $\mathsf{i}_0$ he sets the VRF $vk^{\mathsf{i}_0} = vk^0$ and at $\mathsf{i}_1$ he sets $vk^{\mathsf{i}_1} = vk^1$.
4. On an $\mathsf{OSign}$ query for $\mathsf{msg}_i, R_i$, at $\mathsf{i}_0$: $\mathcal{A}_{KP}^{VRF}$ asks the challenger to return $v \leftarrow \mathsf{Eval}_{VRF}(sk^b, \mathsf{msg}_i||R_i)$
5. When $\mathcal{A}_{anon}$ makes his challenge, $(s_0, s_1, \mathsf{msg}, R)$, then $\mathcal{A}_{KP}^{VRF}$ submits $\mathsf{msg}||R$ to the challenger as his challenge and gets back $v^*$. He uses this in the signature $\sigma = (v^*, ct, hk, h, \pi)$.

If $b = 0$, then the $\mathcal{A}_{KP}^{VRF}$ uses $vk^{\mathsf{i}_0}$ to answer queries. If $b = 1$, then $\mathcal{A}_{KP}^{VRF}$ uses $vk^{\mathsf{i}_1}$. By the VRF's key privacy property, $\mathcal{A}_{KP}^{VRF}$ cannot distinguish between a $v$ from $vk^{\mathsf{i}_0}$ and $vk^{\mathsf{i}_1}$. Thus, $\mathcal{H}_8$ and $\mathcal{H}_9$ are indistinguishable.

$\mathcal{H}_9$ **to** $\mathcal{H}_{10}$**:** $\sigma = (v^1, v', \widehat{ct}^1, \widehat{ct}'^1, hk^{\mathsf{i}_0}, hk^{\mathsf{i}_1}, \pi, t) \rightarrow (v^1, v'^1, \widehat{ct}^1, \widehat{ct}'^1, hk^{\mathsf{i}_0}, hk^{\mathsf{i}_1}, \pi, t)$. The challenger replaces $v^1$ by $v'^1 \leftarrow \mathsf{Eval}_{PRF}(sk^{\mathsf{i}_1}, t||\mathsf{msg}||P)$. The argument is as in $\mathcal{H}_8$ to $\mathcal{H}_9$.

$\mathcal{H}_{10}$ **to** $\mathcal{H}_{11}$**:** $(hk, shk) \leftarrow \mathsf{Gen}_{SPB}(1^\lambda, N, \mathsf{i}_0) \rightarrow hk^1, shk^1 \leftarrow \mathsf{Gen}_{SPB}(1^\lambda, N, \mathsf{i}_1)$.

Because of the SPB's index-hiding property we can next change the index for which $hk$ is generated from $\mathsf{i}_0$ to $\mathsf{i}_1$. We construct $\mathcal{A}_{IH}$ as an adversary against SPB index hiding which uses $\mathcal{A}_{anon}$ as a subroutine.

1. $\mathcal{A}_{IH}$ picks $(N, \mathsf{i}_0, \mathsf{i}_1, \emptyset)$ (where $N$ is the maximum ring size).
2. $\mathcal{A}_{IH}$ activates $\mathcal{A}_{anon}$ as a subroutine. On each query $\mathcal{A}_{IH}$ answers as the challenger would.
3. Eventually, $\mathcal{A}_{anon}$ requests a signature on $\mathsf{i}_0$, $\mathsf{i}_1$.
4. $\mathcal{A}_{IH}$ produces the signature as described in $\mathcal{H}_{10}$, except for he gives $(N, \mathsf{i}_0, \mathsf{i}_1)$ to the challenger. He uses $(hk, shk)$ from the challenger to create the signature for $\mathcal{A}_{anon}$.
5. $\mathcal{A}_{IH}$ outputs same as $\mathcal{A}_{anon}$.

If $\mathcal{A}_{anon}$ wins with non-negligibly different probabilities in $\mathcal{H}_{10}$ and $\mathcal{H}_{11}$, then $\mathcal{A}_{IH}$ could win the index hiding experiment. We see then that $\mathcal{H}_{10}$ and $\mathcal{H}_{11}$ must be indistinguishable.

**$\mathcal{H}_{11}$ to $\mathcal{H}_{12}$:** Using $\tau^1 \leftarrow \mathsf{Open}_{SPB}(hk^1, shk^1, R, \mathsf{i}_1)$, select $sk_{\mathsf{F}}^{\mathsf{i}_0}, sk_{\mathsf{F}}^{\mathsf{i}_1}$ randomly when requested for $\mathsf{OKGen}$ and change the witness:

$$\mathcal{H}_{11}\ \widehat{\mathsf{w}}^0 = (VK^{\mathsf{i}_0}, VK^{\mathsf{i}_1}, \mathsf{i}_0, \mathsf{i}_1, \tau^{\mathsf{i}_0}, \tau^{\mathsf{i}_1}, p^1, p'^1, sk_{\mathsf{F}}^{\mathsf{i}_0}, sk_{\mathsf{F}}^{\mathsf{i}_1}, r_{ct_1}, r_{ct'_1}, , r_{E_0}, r_{E_1})$$

$$\mathcal{H}_{12}\ \mathsf{w}^1 = (VK^{\mathsf{i}_1}, VK^{\mathsf{i}_0}, \mathsf{i}_1, \mathsf{i}_0, \tau^{\mathsf{i}_1}, \tau^{\mathsf{i}_0}, p^1, p'^1, sk_{\mathsf{F}}^{\mathsf{i}_1}, sk_{\mathsf{F}}^{\mathsf{i}_0}, r_{ct_1}, r_{ct'_1}, r_{E_0}, r_{E_1})$$

and use to compute $\pi^1 \leftarrow \mathsf{Prove}_{NIWI}(x, \mathsf{w}^1)$. This change is indistinguishable because NIWI has witness indistinguishability.

In $\mathcal{H}_{12}$, the challenger is returning a signature for $\mathsf{i}_1$. Because each hybrid is computationally indistinguishable from its predecessor, we see that signatures under $s_0$ and $s_1$ are indistinguishable to $\mathcal{A}_{anon}$. □

## 5 (Scoped) Linkable Thring Signatures

We extend the techniques in TRS to create a *linkable* threshold ring signature scheme LTRS. Linkability [23] means that given two thring signatures for any two messages, one can verify whether (at least one of) the same signers contributed to both signatures. The verification is done via a Link algorithm that takes as input two thring signatures and outputs a bit indicating whether the two signatures are linked.

The security framework and construction presented in the following support *scoped* linkability, where two signatures link if they have been produced by related sets of signers for the same scope (e.g., context information)[5]. Scoped linkability is more fine-grained than linkability: two signatures are linkable if they have been produced w.r.t the same scope, but across different scopes signatures cannot be linked. Scope can be an arbitrary string. Using a scope string fixed in

[5] We note that this concept is not new and has previously been used within anonymous credential systems (cf. [9]), direct anonymous attestation [7] and also in context of traceable ring signatures [16].

the scheme yields the conventional notion of linking. Like BDH$^+$ [3] recently did for ring signatures, we present the first construction of a *linkable* thring signature scheme in the plain model by building upon our thring signature scheme.

The standard security requirements for (scope-)linkable threshold ring signatures are correctness, unforgeability, *scoped linkability*, *linkable anonymity*, and *non-frameability*. Scoped linkability requires that even maliciously generated signatures need to link. With linkable anonymity, while it is possible to see that two signatures come from the same signer, it is not possible to determine which signer it is. Finally, non-frameability requires that an adversary, even after seeing many messages and signatures, cannot generate fresh signatures which will link to signatures that have been generated by honest parties.

**Syntax.** A linkable threshold ring signature scheme is a 5-tuple of algorithms $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy}, \mathsf{Link})$ and an extension of threshold ring signatures, where $\mathsf{Sign}$ and $\mathsf{Vfy}$ take an extra input $\mathsf{sc}$ (the scope). Thus, we do not detail the first four interfaces here. Finally, $\mathsf{Link}$ is defined as follows:

– $b \leftarrow \mathsf{Link}(\sigma_1, \sigma_2)$. On input two valid threshold ring signatures for the same scope, this deterministic algorithm outputs a single bit $b$.

### 5.1 Properties and Definitions

We now formally define the above mentioned properties.
*Scoped Linkability.* Intuitively, scoped linkability guarantees that signatures from non-disjoint sets of signers for the same scope will link. This is captured by giving the adversary access to honestly generated keys for a ring of size $q$ (for any $q$) and have the adversary output $q + 1$ valid signatures for the same scope. The adversary wins if none of them link to each other. The definition of linkability is reminiscent of unforgeability (Def. 10): the adversary with only knowledge of $q$ keys cannot create $q+1$ signatures. We have a similar limitation with linkability as we had with unforgeability: our construction has us exclude malicious keys due to our use of $\mathcal{R}_{\mathcal{F}}$. Thus, we miss the situation where the adversary forges signatures using malicious keys, which could be trivially made to link. As before, this is inherent in our scheme due to the use of the VRF and the OWF which allows us to be in the plain model. The formal experiment is provided in Fig. 4.

We note that the proposed signatures will be linkable for different scopes if the same message/ring is signed.

**Definition 12 (Scoped Linkability).** *A linkable threshold ring signature scheme satisfies* scoped linkability *if for every PPT adversary $\mathcal{A}$ and every $q$ polynomially bounded in $\lambda$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that*

$$\Pr[\mathsf{ScopedLinkability}^{\mathcal{A}}(\lambda, q) = 1] \leq \mathsf{negl}(\lambda)\,.$$

*Non-Frameability.* Non-frameability guarantees that no adversary can generate fresh signatures which will link to signatures that have been generated by honest parties. The adversary has access to $\mathsf{OCorr}$ and $\mathsf{OSign}$ and can receive arbitrarily

**Experiment** $\mathsf{ScopedLinkability}^{\mathcal{A}}(\lambda, q)$
$pp \leftarrow \mathsf{Setup}(1^\lambda)$
$(vk^s, sk^s) \leftarrow \mathsf{KGen}(pp)$ for $s \in [q]$
$(\{(\mathsf{msg}_i, R_i, \sigma_i, t_i)\}_{i\in[q+1]}, \mathsf{sc}) \leftarrow \mathcal{A}(\{(vk^s, sk^s)\}_{s\in[q]})$
return 1 if:
$\mathsf{Vfy}(\mathsf{msg}_i, R_i, \sigma_i, t_i, \mathsf{sc}) = 1$ for $i \in [q+1]$
$R_i \subseteq \{vk^1, \ldots, vk^s\}$ for $i \in [q+1]$
$\forall i \neq j \in [q+1] \ : \ \mathsf{Link}(\sigma^i, \sigma^j) = 0$
return 0

**Fig. 4.** Scoped linkability

**Experiment** $\mathsf{Frameability}^{\mathcal{A}}(\lambda, q)$
$pp \leftarrow \mathsf{Setup}(1^\lambda)$
$(vk^s, sk^s) \leftarrow \mathsf{KGen}(pp)$ for $s \in [q]$
$(\mathsf{st}, \mathsf{msg}^*, R^*, \sigma^*, t^*, \mathsf{sc}) \leftarrow \mathcal{A}^{\mathsf{OCorr},\mathsf{OSign}}(\{vk^s\}_{s\in[q]})$
where $\sigma^* = (\sigma_1^*, \ldots, \sigma_{n^*}^*)$
$(\mathsf{msg}^\dagger, R^\dagger, \sigma^\dagger, t^\dagger) \leftarrow \mathcal{A}(\mathsf{st}, \{sk^s\}_{s\in[q]})$
where $\sigma^\dagger = (\sigma_1^\dagger, \ldots, \sigma_{n^\dagger}^\dagger)$
return 1 if:
$\mathsf{Vfy}(\mathsf{msg}^*, R^*, \sigma^*, t^*, \mathsf{sc}) = 1$
$\mathsf{Vfy}(\mathsf{msg}^\dagger, R^\dagger, \sigma^\dagger, t^\dagger, \mathsf{sc}) = 1$
$R^* \cup R^\dagger \subseteq P$
$|R^* \cap P_{corr}| < t^*$
$\exists i$ such that $\sigma_i^*$ was not obtained for $(\mathsf{msg}^*, R^*)$ from $\mathsf{OSign}$
$R^* \cap R^\dagger \cap P_{corr} = \emptyset$
$\mathsf{Link}(\sigma^*, \sigma^\dagger) = 1$
return 0

**Fig. 5.** Non-frameability

many signatures, and finally outputs a strong forgery, i.e., a fresh signature to a new message and subring. The adversary then learns all secret keys and wins if it can generate another signature which links to the former one, if no corrupted user was in the subring for both signatures (as this would allow for trivial attacks).

**Definition 13 (Non-Frameability).** *A linkable threshold ring signature scheme satisfies* non-frameability *if for every PPT adversary $\mathcal{A}$ and every $q$ polynomially bounded in $\lambda$, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that*

$$\Pr[\mathsf{Frameability}^{\mathcal{A}}(\lambda, q) = 1] \leq \mathsf{negl}(\lambda)\,.$$

In our definition (see Fig. 5 for the formal experiment), we only consider signature schemes where the threshold signature consists of a list of individual signatures (as is also the case in the construction). This is because the same message can be signed for the same subring and scope, but by two disjoint sets of signers. Because of non-interactivity and inter-signer anonymity, the individual contributions of the signature cannot depend on the set of signers, and thus an adversary could create a fresh overall signature by combining parts from both signatures. The adversary could then to trivially win the frameability experiment, if the winning condition only excluded that the overall challenge signature $\sigma^*$ has not been generated by $\mathsf{OSign}$, while not having a real-world impact. To overcome this problem, we either must drop inter-signer anonymity or require an interactive process.

*Linkable Anonymity.* For linkable anonymity, it is not possible to decide from which signer in the ring the signatures came from, only what signatures are linked together. We capture this concept formally in the experiment in Fig. 6.

The adversary picks two signing sets $S_0, S_1$ such that $|S_0| = |S_1|$. We can assume that $S_0 \cap S_1 = \emptyset$ without loss of generality. By ordering members of each set, we have a correspondence from user $i_k$ and $j_k$ in $S_0, S_1$ respectively. We say a key $VK_0^s \in S_0$ is matched with a key $VK_1^s \in S_1$.

Then in one case, all signature queries with signer $i_k$ are signed with $i_k$. Otherwise, all signature queries of $i_k$ are signed with $j_k$ (and vice versa). Then $\mathcal{A}$ can use $\mathsf{Link}$ for any signature gotten from the challenger. If $\mathcal{A}$ requested two signatures for $i_k$ then these two signatures will always link.

Depending on the bit $b$ the challenger generates, the requested signatures on the sets $S_0, S_1$ have this form: If $b = 0$ then the signers are the ones $\mathcal{A}$ asks for. If $b = 1$ then each signature on $i_k \in S_0$ is replaced with $j_k \in S_1$ (and vice versa). In the end, $\mathcal{A}$ must decide whether the signatures were signed according to what he requested, or whether they were all flipped.

We note a weakness in our scheme: neither $S_0$ nor $S_1$ can have a corrupted member in the set. That is, if user $i_k$ is corrupted, then it will be easy for $\mathcal{A}$ to learn whether $i_k$'s key was used to create a signature. Therefore, we have $\mathsf{OCorr}$ ignores any calls to users in $S_0 \cup S_1$. This is a rather weak definition in the context of ring signatures, but seems unavoidable when using a deterministic function such as VRFs.

**Experiment** $\mathsf{LinkableAnonymity}^{\mathcal{A}}(\lambda, q)$
- $pp \leftarrow \mathsf{Setup}(1^\lambda)$
- $(vk^s, sk^s) \leftarrow \mathsf{KGen}(pp)$ for $s \in [q]$
- $b \leftarrow \{0,1\}$
- $(S_0, S_1, \mathsf{st}) \leftarrow \mathcal{A}(\{vk^s\}_{s\in[q]})$
  - let $S_0 = \{vk^{i_1}, \ldots, vk^{i_m}\}$ and $S_1 = \{vk^{j_1}, \ldots, vk^{j_m}\}$
- $(S_0^*, S_1^*, \mathsf{msg}^*, R^*, \vec{t}^*, \mathsf{sc}^*, \mathsf{st}) \leftarrow \mathcal{A}^{\mathsf{OSign},\mathsf{OSign}',\mathsf{OCorr}}(\mathsf{st})$
  - where $\mathsf{OCorr}$ ignores any calls to users in $S_0 \cup S_1$
  - where $\mathsf{OSign}'$ engages in a signing protocol with $\mathcal{A}$ on the given inputs mimicking all uncorrupted users
  - where $\mathsf{OSign}$ ignores calls where $\exists m$ with $vk^{i_m} \in S$ or $vk^{j_m} \in S$ but $\{vk^{i_m}, vk^{j_m}\} \not\subseteq S$, and otherwise computes $S'$ by replacing all signers from $S_0$ with $S_1$ and vice versa, i.e., with $S' = S \setminus (\{vk^{i_m}\}_{i_m \in S} \cup \{vk^{j_m}\}_{j_m \in S}) \cup (\{vk^{j_m}\}_{i_m \in S} \cup \{vk^{i_m}\}_{j_m \in S})$ and then engages in signing protocols with $\mathcal{A}$ for sets $S$ and $S'$ ($b = 0$) or $S'$ and $S$ ($b = 1$).
- $\sigma^* \leftarrow \mathsf{Sign}(\mathsf{msg}^*, T_b^*, R^*, \vec{t}^*, \mathsf{sc}^*)$
  - where $T_b^* = \{SK^s\}$ for $s \in S$ if $b = 0$ and $s \in S'$ if $b = 1$.
- $b' \leftarrow \mathcal{A}(\mathsf{st}, \sigma^*)$

return a random bit if:
- $(\mathsf{msg}^*, R^*)$ was queried before, or
- $(\mathsf{sc}^*, S_j')$ has been queried before for some $j \in \{0,1\}$, or
- $S_0 \cap S_1 \neq \emptyset$

return $b = b'$

**Fig. 6.** Linkable anonymity.

**Definition 14 (Linkable Anonymity).** *A linkable threshold ring signature scheme satisfies* scope-exclusive linkable anonymity *if for every PPT adversary $\mathcal{A}$ and every $q$ polynomially bounded in $\lambda$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that*

$$\left|\Pr[\mathsf{LinkableAnonymity}^{\mathcal{A}}(\lambda, q) = 1] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)\,.$$

### 5.2 Our Construction

Our threshold ring signature TRS is modular in the sense that we need only to add a few elements to TRS to turn it into a linkable thring signature scheme including the concept of a scope. The full LTRS construction is presented in Fig. 7.

**Key Generation** $\mathsf{Gen}(1^\lambda)$:

$(vk, sk) \leftarrow \mathsf{Gen}_{VRF}(1^\lambda)$;
$(vk_L, sk_L) \leftarrow \mathsf{Gen}_{VRF}(1^\lambda)$;
$(vk_{mal}, sk_{mal}) \leftarrow \mathsf{Gen}_{VRF}(1^\lambda)$;
$(pk_\dagger, sk_\dagger) \leftarrow \mathsf{Gen}_{PKE}(1^\lambda)$;
$(pk_\ddagger, sk_\ddagger) \leftarrow \mathsf{Gen}_{PKE}(1^\lambda)$;
$sk_\mathsf{F} \leftarrow \{0,1\}^\lambda$;
$r_E \leftarrow Rnd_{PKE}$;
$E \leftarrow \mathsf{Enc}_{PKE}(pk_\ddagger, sk_\mathsf{F}; r_E)$;
$VK := (vk, vk_L, vk_{mal}, pk_\dagger, pk_\ddagger, E)$;
$SK := (sk, sk_L, sk_{mal}, sk_\dagger, sk_\ddagger, r_E, VK)$;
**return** $(VK, SK)$.

**Verification** $\mathsf{Vfy}(\mathsf{msg}, R, \sigma, t_V, \mathsf{sc})$:

// Parse signature;
$\sigma = (\rho, \varsigma, \mathsf{vk}_{sOTS})_{j=1}^{t}$;
$\rho = (v, v', v_L, v_{mal}, ct, ct', ct_L, ct_{mal}, hk^s, hk^i, \pi, t^j, \mathsf{sc}^j)$;
Sort list by $t^i$;
**for** $j \in [t]$
  $h' := \mathsf{Hash}_{SPB}(hk^s, R)$;
  $h'' := \mathsf{Hash}_{SPB}(hk^i, R)$;
  $x := (\mathsf{msg}, R, v, v', ct, ct', ct_L, ct_{mal}, h', h'', hk^s, hk^i, \mathsf{sc})$;
  $b \leftarrow \mathsf{Vfy}_{NIWI}(x, \pi)$;
  $b \leftarrow b \ \wedge \ \mathsf{Vfy}_{sOTS}(\mathsf{vk}_{sOTS}, \rho, \varsigma)$;
  **if** $b = 1 \ \wedge \ \sigma^j.v \neq \sigma^k.v \ \forall k \in [j-1]$
    $\mathcal{L}_V.\mathsf{append}(t^j)$;
**endfor**
**if** $\exists i \geq t_V : \ \mathcal{L}_V[i] \leq i$ **return** 1;
**return** 0

**Link** $\mathsf{Link}(\sigma_1, \sigma_2)$ :

*Let* $t_i := |\sigma_i|, i \in \{1,2\}$
**for** $(j,k) \in [t_1] \times [t_2]$
  **if** $\sigma_1^j.v_L = \sigma_2^k.v_L$ **return** 1
**endfor**
**return** 0

**Threshold Signing** $\mathsf{Sign}(\mathsf{msg}, T, R, \vec{t}, \mathsf{sc})$:

// Every signer $s \in S$, $|S| \geq t$
$v \leftarrow \mathsf{Eval}_{VRF}(sk^s, \mathsf{msg}||R)$;
$p \leftarrow \mathsf{Prove}_{VRF}(sk^s, \mathsf{msg}||R)$;
$v' \leftarrow \mathsf{Eval}_{VRF}(sk^s, t^s||\mathsf{msg}||R)$;
$p' \leftarrow \mathsf{Prove}_{VRF}(sk^s, t^s||\mathsf{msg}||R)$;
$v_L \leftarrow \mathsf{Eval}_{VRF}(sk_L^s, \mathsf{sc})$;
$p_L \leftarrow \mathsf{Prove}_{VRF}(sk_L^s, \mathsf{sc})$;
$(vk_{sOTS}, sk_{sOTS}) \leftarrow \mathsf{Gen}_{sOTS}(1^\lambda)$;
$v_{mal} \leftarrow \mathsf{Eval}_{VRF}(sk_{mal}^s, vk_{sOTS})$;
$p_{mal} \leftarrow \mathsf{Prove}_{VRF}(sk_{mal}^s, vk_{sOTS})$;
$r_{ct}, r_{ct'}, r_L, r_{mal} \leftarrow Rnd_{PKE}$;
$ct \leftarrow \mathsf{Enc}_{PKE}(pk_\dagger^s, p; r_{ct})$;
$ct' \leftarrow \mathsf{Enc}_{PKE}(pk_\dagger^s, p'; r_{ct'})$;
$ct_L \leftarrow \mathsf{Enc}_{PKE}(pk_\dagger^s, p_L; r_L)$;
$ct_{mal} \leftarrow \mathsf{Enc}_{PKE}(pk_\dagger^s, p_{mal}; r_{mal})$;
$(hk^s, shk^s) \leftarrow \mathsf{Gen}_{SPB}(1^\lambda, N, s)$;
$h^s \leftarrow \mathsf{Hash}_{SPB}(hk^s, R)$;
$\tau^s \leftarrow \mathsf{Open}_{SPB}(hk^s, shk^s, R, s)$;
// Pick other ring member $i \neq s$
$i \leftarrow [N] \setminus s$;
$r_{E_0}, r_{E_1} \leftarrow Rnd_{PKE}$
$(hk^i, shk^{i_1}) \leftarrow \mathsf{Gen}_{SPB}(1^\lambda, N, i)$;
$h^i := \mathsf{Hash}_{SPB}(hk^i, R)$;
$\tau^i \leftarrow \mathsf{Open}_{SPB}(hk^{i_1}, shk^{i_1}, R, i)$;
// Call on the NIWI for language $\mathcal{L}'_L$
$\pi \leftarrow \mathsf{Prove}_{NIWI}(x, \mathsf{w})$
$\rho := (v, v', v_L, v_{mal}, ct, ct', ct_L, ct_{mal}, hk^s, hk^i, \pi, t^s, \mathsf{sc})$;
$\varsigma \leftarrow \mathsf{Sign}_{sOTS}(\mathsf{sk}_{sOTS}, \rho)$;
$\sigma^s := (\rho, \varsigma, \mathsf{vk}_{sOTS})$;
// Every signer $s$ broadcasts the signature
**broadcast** $\sigma^s$;
// Final threshold ring signature
**return** $\sigma = \{\sigma^i\}_{i=1}^{t}$.

**Fig. 7.** Linkable threshold ring signature scheme LTRS (changes to TRS highlighted).

Besides TRS keys, we include two VRF keys $(vk_L, sk_L)$ (for linking) and $(vk_{mal}, sk_{mal})$ (for achieving non-malleability). For signing, a signer additionally evaluates the first VRF on the scope $\mathsf{sc}$ to get $(v_L, p_L)$ and encrypts $p_L$ into $ct_L$. The evaluation on scope is necessary to allow for scoped linkability. Then, it creates a key-pair $(vk_{sOTS}, sk_{sOTS})$ for a strong one-time signature (sOTS) scheme. The signer evaluates another VRF using $sk_{mal}$ on $vk_{sOTS}$ (i.e., "signs" the verification key) and encrypts $p_{mal}$ into $ct_{mal}$. The purpose of the second VRF is for non-malleability, i.e., $sk_{sOTS}$ is used to sign the partial signature and the final signature also includes the sOTS signature. As before, the signer evaluates a NIWI:

*NIWI.* The NIWI consists of the OR of three different languages:

$$\mathcal{L}'_L := (\mathcal{L}_{\mathcal{R}|_{VK}} \wedge \mathcal{L}_{\mathcal{R}_{Link}}) \vee (\mathcal{L}_{\mathcal{R}|_{VK'}} \wedge \mathcal{L}_{\mathcal{R}_{Link'}}) \vee \mathcal{L}_{\mathcal{R}_F}$$

The relation $\mathcal{R}_F$ is identical to the one in TRS and $\mathcal{R}|_{VK}$ is a straightforward adaption of the one in TRS. We added a new language for the relationship $\mathcal{R}_{Link}$, which allows a signer to maintain anonymity and non-frameability: $\mathcal{R}_{Link}(x, \mathsf{w}) \iff$

$$ct_L = \mathsf{Enc}_{PKE}(pk_\dagger, p_L; r_L) \ \wedge \ ct_{mal} = \mathsf{Enc}_{PKE}(pk_\dagger, p_{mal}; r_{mal}) \ \wedge$$
$$\mathsf{Vfy}_{VRF}(vk_L, \mathsf{sc}, v_L, p_L) = 1 \ \wedge \ \mathsf{Vfy}_{VRF}(vk_L, vk_{ots}, v_{mal}, p_{mal}) = 1$$

Statements $x$ and witnesses $\mathsf{w}$ for $\mathcal{L}'_L$ are of the form:

$$x = \begin{pmatrix} \mathsf{msg} & R & v & v_{mal} & v_L \\ h^s & h^i & hk^s & hk^i & vk_{ots} \\ ct & ct_{mal} & \mathsf{sc} & ct_L & \varsigma_{ots} \end{pmatrix} \qquad \mathsf{w} = \begin{pmatrix} VK^s & VK^{s_1} & s & s_1 & \tau^s & \tau^i \\ p & p_{mal} & p_L & sk_{\mathsf{F}}^{i_0} & sk_{\mathsf{F}}^{i_1} & \\ r_{ct} & r_{ct'} & r_{E_0} & r_{E_1} & r_L & r_{mal} \end{pmatrix}$$

### 5.3 Security of Our Construction

In the following we state the security claims for our (scope) linkable thring signature scheme LTRS. The proofs are along the same lines as those for TRS and therefore omitted; proof sketches can be found inthe full version.

**Theorem 4.** *If* F *is a OWF,* VRF *has residual unpredictability and unique provability,* NIWI *has perfect soundness, SPB is somewhere perfectly binding and PKE is perfectly correct, SPB is index hiding then* LTRS *is unforgeable.*

**Theorem 5.** *If* F *is a one-way function, the NIWI has perfect soundness, PKE is perfectly correct, SPB is somewhere perfectly binding, VRF has residual unpredictability and key collision resistance, and sOTS is strongly unforgeable then* LTRS *is non-frameable.*

**Theorem 6.** *If the NIWI is computationally witness-indistinguishable, PKE has key-privacy and CPA-security, and VRF has key privacy and residual pseudorandomness, then* LTRS *has linkable anonymity.*

**Theorem 7.** *If the NIWI has perfect soundness, SPB is somewhere perfectly binding, the VRF has unique provability, and PKE is perfectly correct, then* LTRS *is linkable.*

## 6 Instantiations and Future Directions

Our construction is generic but we have made some choices for convenience, i.e., the use of PKE instead of commitments as key-privacy is a natural well studied notion for PKEs. Thus, for a concrete instantiation there may be a number of choices and possible optimizations, which are outside the scope of this paper. Also, for asymptotics, the general algebraic circuit for the NIWI will have a polynomial expansion in the size of its input. The input is logarithmic in the number of users of the ring, and therefore the overall size of the proof is *polylogarithmic* in the size of the input. This is a natural limitation when relying on general building blocks, an issue that is also present in BDH$^+$ [3]. We discuss later how this could be circumvented.

**Towards Post-Quantum Instantiations.** If the instantiations of underlying primitives are post-quantum secure, then our thring signature scheme is also post-quantum secure. Post-quantum VRFs that rely on LWE [17] exist[6] and so do key-private PKE schemes (e.g., based on LWE [22]). Many PKE schemes have key privacy, as this property immediately holds if the ciphertexts are pseudorandom. SPBs can be constructed from somewhere statistically binding hashing (SSBs). BDH$^+$ show in Appendix A.2 of [2] how to turn two-to-one SSBs into SPBs. There are SSBs that rely only on the existence of a lossy/injective functions. One natural lossy/injective function is one built from the LWE problem [1]. The other building blocks we require are a post-quantum strong one-time signature scheme and OWFs. One example for the former is the Winternitz scheme [8] and for latter there are multiple candidates (e.g., from assumptions such as LWE or SIS or based on symmetric primitives as in Picnic [10] and related signature schemes).

The last concern is whether one can construct NIWIs that are post-quantum secure. While NIWIs in the post-quantum setting are not known, as discussed in a recent work by Chatterjee et al. [11] and based on an observation in [5], the NIWI in the BDH$^+$ approach (and also ours) can be replaced with a two-message public coin argument systems (ZAPs [14]). This can be done by extending verification keys with the first message of the ZAP (cf. [11]). While ZAPs are known under the LWE assumption [4,18], one requires to rely on subexponential hardness. To achieve standard polynomial hardness, it though might be possible to adapt the recent approach in [11], which uses the BDH$^+$ approach along with a novel ZAPs for a limited class of languages to achieve compact ring signatures in the plain model from LWE.

**Potential Trade-offs.** The most challenging part of our approach is proving that a verification key belongs to the ring of verification keys. Like BDH$^+$ [3], our thring signature is asymptotically logarithmic but due to the insistence of being in the plain model, there are technical sticking points that guide our choice of building blocks. In particular, when we want to use NIWIs we require perfect

[6] Though key privacy and key collision resistance seem natural in this approach, a formal treatment is missing.

soundness and thus like BDH$^+$ [3] rely on SPBs. Clearly, if we move to knowledge sound NIZKs and thus allow a trusted setup (CRS), then we can move to computationally sound versions and in particular accumulators, e.g., Merkle-tree accumulators with log-sized membership witnesses or even ones with constant size. For the latter ones, the accumulators rely on a trusted setup. Using accumulators was already shown to be useful to get compact ring signatures [12] and in concurrent and independent work also more compact thring signatures [26]. However, the latter requires accepting trusted setup and the random oracle heuristic. We expect that our core idea could also be combined with these primitives when accepting these additional assumptions. Another direction to reduce the signature size would be replace the NIWI with a zk-SNARG or zk-SNARK. However, this would again require a trusted setup or the random oracle heuristic and additionally non-falsifiable assumptions in the latter case.

**Acknowledgments.** We thank anonymous reviewers for their comments. This work was in part funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 830929 (CyberSec4Europe) and by the Austrian Science Fund (FWF) and netidee SCIENCE under grant agreement P31621-N38 (PROFET).

## References

1. Alwen, J., Krenn, S., Pietrzak, K., Wichs, D.: Learning with rounding, revisited. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 57–74. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_4
2. Backes, M., Döttling, N., Hanzlik, L., Kluczniak, K., Schneider, J.: Ring signatures: Logarithmic-size, no setup – from standard assumptions. Cryptology ePrint Archive, Report 2019/196 (2019)
3. Backes, M., Döttling, N., Hanzlik, L., Kluczniak, K., Schneider, J.: Ring signatures: logarithmic-size, no setup—from standard assumptions. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 281–311. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_10
4. Badrinarayanan, S., Fernando, R., Jain, A., Khurana, D., Sahai, A.: Statistical ZAP arguments. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 642–667. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_22
5. Bender, A., Katz, J., Morselli, R.: Ring signatures: stronger definitions, and constructions without random oracles. J. Cryptol. **22**(1), 114–138 (2009)
6. Bresson, E., Stern, J., Szydlo, M.: Threshold ring signatures and applications to ad-hoc groups. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 465–480. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_30
7. Brickell, E.F., Camenisch, J., Chen, L.: Direct anonymous attestation. In: ACM CCS (2004)
8. Buchmann, J., Dahmen, E., Ereth, S., Hülsing, A., Rückert, M.: On the security of the winternitz one-time signature scheme. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 363–378. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21969-6_23

9. Camenisch, J., Krenn, S., Lehmann, A., Mikkelsen, G.L., Neven, G., Pedersen, M.Ø.: Formal treatment of privacy-enhancing credential systems. In: SAC (2015)
10. Chase, M., et al.: Post-quantum zero-knowledge and signatures from symmetric-key primitives. In: ACM CCS (2017)
11. Chatterjee, R., et al.: Compact ring signatures from learning with errors. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 282–312. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_11
12. Dodis, Y., Kiayias, A., Nicolosi, A., Shoup, V.: Anonymous identification in *ad hoc* groups. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 609–626. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_36
13. Dodis, Yevgeniy, Yampolskiy, Aleksandr: A verifiable random function with short proofs and keys. In: Vaudenay, Serge (ed.) PKC 2005. LNCS, vol. 3386, pp. 416–431. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30580-4_28
14. Dwork, C., Naor, M.: Zaps and their applications. In: 41st FOCS (2000)
15. Feige, U., Shamir, A.: Witness indistinguishable and witness hiding protocols. In: STOC (1990)
16. Fujisaki, Eiichiro, Suzuki, Koutarou: Traceable ring signature. In: Okamoto, Tatsuaki, Wang, Xiaoyun (eds.) PKC 2007. LNCS, vol. 4450, pp. 181–200. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71677-8_13
17. Goyal, R., Hohenberger, S., Koppula, V., Waters, B.: A generic approach to constructing and proving verifiable random functions. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10678, pp. 537–566. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_18
18. Goyal, V., Jain, A., Jin, Z., Malavolta, G.: Statistical zaps and new oblivious transfer protocols. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 668–699. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_23
19. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (2006). https://doi.org/10.1007/11935230_29
20. Hofheinz, D., Jager, T.: Verifiable random functions from standard assumptions. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016. LNCS, vol. 9562, pp. 336–362. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49096-9_14
21. Lin, H., Wang, M.: Repudiable ring signature: Stronger security and logarithmic-size. Cryptology ePrint Archive, Report 2019/1269 (2019)
22. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_21
23. Liu, J.K., Wei, V.K., Wong, D.S.: Linkable spontaneous anonymous group signature for ad hoc groups. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 325–335. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27800-9_28
24. Lysyanskaya, A.: Unique signatures and verifiable random functions from the DH-DDH separation. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 597–612. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_38
25. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: 40th FOCS (1999)

26. Munch-Hansen, A., Orlandi, C., Yakoubov, S.: Stronger notions and a more efficient construction of threshold ring signatures. In: Longa, P., Ràfols, C. (eds.) LATINCRYPT 2021. LNCS, vol. 12912, pp. 363–381. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88238-9_18
27. Okamoto, T., Tso, R., Yamaguchi, M., Okamoto, E.: A $k$-out-of-$n$ ring signature with flexible participation for signers. Cryptology ePrint Archive, Report 2018/728 (2018)
28. Okamoto, T., Pietrzak, K., Waters, B., Wichs, D.: New realizations of somewhere statistically binding hashing and positional accumulators. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 121–145. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_6
29. Park, S., Sealfon, A.: It wasn't me! - repudiability and claimability of ring signatures. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11694, pp. 159–190. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26954-8_6
30. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_32
31. Sahai, A.: Simulation-sound non-interactive zero knowledge. Technical report, IBM RESEARCH REPORT RZ 3076 (2000)
32. Tsang, P.P., Wei, V.K.: Short linkable ring signatures for e-voting, e-cash and attestation. In: Deng, R.H., Bao, F., Pang, H.H., Zhou, J. (eds.) ISPEC 2005. LNCS, vol. 3439, pp. 48–60. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31979-5_5