Repudiable Ring Signatures: Stronger Definitions and Logarithmic-Size

Hao Lin[1,2] , Mingqiang Wang[1,2]

1. School of Mathematics and System Sciences, Shandong University, Jinan, Shandong 250100, PR China;
lhao17@mail.sdu.edu.cn, wangmingqiang@sdu.edu.cn
2. China Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education.

Abstract. Ring signatures allow a person to generate a signature on behalf of an ad hoc group, and can hide the true identity of the signer among the group. Repudiable ring signatures are the more strongly defined ring signatures, which can allow every non-signer to prove to others that the signature was not generated by himself.

This paper has two main areas of focus. First, we propose a new requirement for repudiable ring signatures, which is that no one can forge a valid repudiation for others. Second, as a breakthrough, we present the first logarithmic-size repudiable ring signatures which do not rely on a trusted setup or the random oracle model. Specifically, our scheme can be instantiated from standard assumptions and the size of signatures and repudiations only grows logarithmically in the number of ring members.

Besides, our scheme also provides a new construction of logarithmic-size standard ring signatures.

Keywords: ring signatures, repudiable ring signatures, standard model

1 Introduction

Ring signatures, introduced by [19], are a variant of digital signatures, which can certify that one among a particular set of parties has signed a particular message, without reveal who is the signer. And this particular set is called a 'ring'. More specifically, the signing algorithm of a ring signature scheme takes as additional input a list of verification keys R and outputs a signature. Such a signature can be verified produced by one among R. The interesting feature of ring signatures is that given such a signature, no one can tell which key was used to compute this signature. Ring signatures are useful, for example, to certify that certain leaked information comes from a privileged set of government officials without revealing the identity of the whistleblower, to issue important orders or directives without setting up the signer to be a scapegoat for repercussions, or to enable untraceable transactions in cryptocurrencies (such as Monero [16]). In terms of security two properties are required in ring signatures: unforgeability and anonymity. The first property requires that an efficient adversary should not be able to forge a

signature on behalf of an honest ring of signers. And anonymity requires that signatures do not give away by which member they were created.

The notion of repudiable ring signatures [18] is an extension of the concept of ring signatures which can allow every non-signer to prove to others that the signature was not generated by himself. More specifically, the repudiable ring signature scheme is a ring signature scheme equipped with an additional pair of algorithms (Repudiate, VerRepud), where Repudiate is an algorithm which can create a repudiation ξ of any signature σ for any non-signer, and VerRepud can verify whether ξ is a valid repudiation.

The repudiability of ring signatures is a necessary property in some situation. For example, we can cite an example from [18] to illustrate the importance of repudiability. Let us consider a hypothetical case, wherein two candidates Alice and Bob are running for president in the land of Oz. Oz is notorious for its petty partisan politics and its tendency to prefer whomever appears friendlier in a series of nationally televised grinning contests between the main-party candidates. At the peak of election season, a disgruntled citizen Eve decides to help out her preferred candidate Bob by publishing the following message, which goes viral on the social networks of Bob supporters:

I created a notorious terrorist group and laundered lots of money!

Signed: Alice or Eve or Alice's campaign chairman.

Of course, the virally publicized message does not actually incriminate Alice at all, since any one of the signatories could have produced it. However, perhaps there is nothing that Alice can do to allay the doubt in the minds of her suspicious detractors.

The reason for this is that ring signatures are deliberately designed to allow anyone to attach anyone else's identity to a signature, without the latter's consent. And just like in the example above, these ring signatures provide protection for malicious people who try to damage the reputation of others. Therefore, we need to use repudiable ring signatures in these situations.

1.1 Our Contributions

In this paper, we focus on both definitions and constructions. We summarize our results in each of these areas, and relate them to prior work.

Definitions of security. Prior work on repudiable ring signature scheme provides definitions of security seem (to us) unnaturally weak, in that they do not address what seem to be valid security concerns. One example is that they did not consider the unforgeability of repudiation. Although at first glance, this property seems to be included in anonymity, it actually requires more in some aspects than anonymity. Because repudiation unforgeability requires that the adversary cannot forge a repudiable of signature σ for others even if he knows the signing key corresponding to the signature σ.

This property is useful in many cases. For example, let us consider the following situation. A company is in trouble and its employees are asked to come up with a solution. Bob is an employee of this company, he thinks of a seemingly feasible solution, but he is afraid of being made a scapegoat by his boss if his solution failed. So he uses a repudiable ring signature to sign his plan, and publishes it. In the end, the plan works and Bob wants to be rewarded by the company alone, so he can forge repudiations for everyone others in his ring. In this case, the ring member can only share the risk but not the reward, which is obviously unfair to the ring members.

The reason for this is that repudiable ring signatures do not satisfy repudiation unforgeability. Therefore, we need repudiation unforgeability in these situations. So in this paper, we formalize the property of repudiation-unforgeability, and give the first construction that satisfies this property.

Constructions. In this paper, we present the first construction of logarithmic-size repudiable ring signatures which do not rely on a trusted setup or the random oracle model. Specifically, our scheme can be instantiated from standard assumptions and our scheme has signatures and repudiations of size $\log(n) \cdot \text{poly}(\lambda)$, whereas the size of the signatures and repudiations of construction in [18] is square in the ring size n.

There are two major obstacles in making the size of the signatures and repudiations sublinear in [18] :

1. The signatures and repudiatons contain all witnesses;
2. The witness for the validity of statement is also size linear in n.

Our first modification is that for signature we can just use NIWI to produce a witness, and do not produce witnesses π for every party, and since NIWI has witness indistinguishable, we also have our signature has anonymity. Our second modification is that we first hash the ring R into a succinct digest h, and then use h in the NIWI. Here we use SPB hashing function, which can also prove the membership of VK_i in the ring R. This hash function was first used by [2].

Besides, the size of keys of our construction has been reduced by at least half compared to scheme in [18].

Other Contributions. We find there are some mistakes in [18]. In [18] they use the notation adaptive anonymity against adversarially chosen keys, but we find their construction cannot satisfy this property they proposed, we can find an attack for their construction. The attack algorithm is in the appendix. To rule out such attack, we need to limit the ability of adversary slightly. Our modification is that, we do not allow an adversary to ask its oracle $\text{OR}(\cdot)$ for $(\cdot, m, \text{R}, \cdot)$ after the adversary gives challenge information (j_0, j_1, m, R) to experiment. Their repudiable ring signatures satisfy this modified anonymity, so do our scheme. And we think this limitation is necessary.

Besides, if we only use the first three algorithms of our repudiable ring signature, it is also a secure ring signature scheme and the size of signatures also grows only logarithmically in the number of ring members. And this is also a new construction of standard ring signatures with logarithmic-size signatures .

1.2 Related Work

After the initial work of Rivest, Shamir and Tauman [19], a number of works provided constructions under various computational hardness assumptions. The scheme of Dodis et al. [7] was the first to achieve sublinear size signatures in the ROM. Libert et al. [12] constructed a scheme with logarithmic size ring signature from DDH in the ROM. And recently, Backes et al. [2] provided a standard model construction with signatures of size $\log(n)$.

And since the original proposal of ring signatures, various variant definitions have been proposed. For example, linkable ring signatures [13] allow identification of signatures that were produced by the same signer, without compromising the anonymity of the signer within the ring. Another notion called traceable ring signature [9] considers a setting where signatures are generated with respect to "tags" and each member may sign at most a single message with respect to a particular tag, or else his identity will be revealed. Accountable ring signatures [1,20] allow a signer to assign the power to deanonymize his signature to a specific publicly identified party. And recently, Park and Sealfon proposed four new notations which are repudiable, unrepudiable, claimable and unclaimable ring signature in [18].

2 Preliminaries

Throughout the paper, we let λ denote the security parameter and $\mathrm{negl}(\lambda)$ denote the negligible function. We denote by $y \leftarrow \mathcal{A}(x;r)$ the execution of algorithm that $\mathcal{A}$ output y, on input x and random coins r. We write $y \leftarrow \mathcal{A}(x)$, if the specific random coins used are not important. And we denote by $y = \mathcal{A}(x)$, if the algorithm is deterministic. Let $r \leftarrow S$ denote that r is chosen uniformly at random from the set S. We use $[n]$ to denote the set $\{1, \ldots, n\}$.

Next, we briefly review some building blocks, which will be used in our scheme.

2.1 Non-Interactive Witness-Indistinguishable Proof Systems

Let $\mathcal{R}$ be an efficiently computable binary relation, where for $(x, w) \in \mathcal{R}$ we call x is a statement and w is a witness of x. Moreover, let $\mathcal{L}_{\mathcal{R}}$ denote the language consisting of all statements in $\mathcal{R}$, i.e. $\mathcal{L}_{\mathcal{R}} = \{x \mid \exists w : (x, w) \in \mathcal{L}_{\mathcal{R}}\}$.

Definition 1 (NIWI). *Let $\mathcal{R}$ be an efficiently computable witness relation and $\mathcal{L}_{\mathcal{R}}$ be the language generated by $\mathcal{R}$. A non-interactive witness-indistinguishable proof* NIWI *for language $\mathcal{L}_{\mathcal{R}}$ is a pair of probabilistic polynomial-time algorithms* $(\mathrm{Prove}, \mathrm{Verify})$ *satisfying the three properties of Perfect Completeness, Perfect Soundness, and Witness-Indistinguishability. The syntax of* NIWI *follows:*

Prove $(1^\lambda, x, w)$**:** *Takes as input a security parameter 1^λ, a statement x and a witness w, outputs either a proof π or $\perp$.*
Verify (x, π)**:** *Takes as input a statement x and a proof π, outputs either 0 or 1.*

We require the following properties of the NIWI :

Perfect Completeness: *For any security parameter* $\lambda \in \mathbb{N}$*, any statement* $x \in \mathcal{L}_{\mathcal{R}}$ *and any witness* w*, that if* $\mathcal{R}(x, w) = 1$ *and* $\pi \leftarrow \text{Prove}(1^\lambda, x, w)$*, then it holds that* $\text{Verify}(x, \pi) = 1$.

Perfect Soundness: *For any security parameter* $\lambda \in \mathbb{N}$*, any statement* $x \notin \mathcal{L}_{\mathcal{R}}$ *and any proof* π*, it holds that* $\text{Verify}(x, \pi) = 0$.

Witness-Indistinguishability: *For any* PPT *adversary* $\mathcal{A}$*, there is a negligible function* negl*, such that for any* $\lambda \in \mathbb{N}$*, adversary* $\mathcal{A}$ *has at most* $\text{negl}(\lambda)$ *advantage in the following experiment:*

$\text{Exp}_{\text{WI}}(\mathcal{A})$:

- *The adversary* $\mathcal{A}$ *is given input* 1^λ*, and output a triple* (x, w_0, w_1) *, with* $\mathcal{R}(x, w_0) = 1$ *and* $\mathcal{R}(x, w_1) = 1$.
- *The experiment chooses a random bit* $b \leftarrow \{0, 1\}$*, and computes* $\pi \leftarrow \text{Prove}(1^\lambda, x, w_b)$*, then gives* π *to* $\mathcal{A}$.
- *The adversary* $\mathcal{A}$ *outputs a guess* b'.
- *The output of the experiment is defined to be* 1 *if* $b' = b$*, and* 0 *otherwise.*

The advantage of $\mathcal{A}$ *is defined by*

$$\text{Adv}_{\text{WI}}(\mathcal{A}) = |\Pr[\text{Exp}_{\text{WI}}(\mathcal{A}) = 1] - \frac{1}{2}|.$$

Proof-size: *For* $\pi \leftarrow \text{Prove}(1^\lambda, x, w)$ *it holds that* $|\pi| = |C_x| \cdot \text{poly}(\lambda)$*, where* C_x *is a verificaiton circuit for the statement* x.

In order not to cause confusion, we write NIWI.Prove, NIWI.Verify to denote the Prove and Verify algorithms belonging to NIWI.

Non-interactive witness-indistinguishable proofs can be constructed from NIZK proofs derandomization assumptions [3,5], from bilinear pairings [10], and indistinguishablity obfuscation [4].

2.2 Verifiable Random Function

Let $a : \mathbb{N} \to \mathbb{N} \cup \{*\}$ and $b,\ s : \mathbb{N} \to \mathbb{N}$ be any three functions such that $a(\lambda), b(\lambda), s(\lambda)$ are all computable in time $\text{poly}(\lambda)$, and $a(\lambda), b(\lambda), s(\lambda)$ are both bounded by a polynomial in λ (expect when $a(\lambda)$ takes on the value $*$).[1]

Definition 2 (VRF). *A family of functions*

$$\mathcal{F} = \{f_k : \{0,1\}^{a(\lambda)} \to \{0,1\}^{b(\lambda)}\}_{k \in \{0,1\}^{s(\lambda)}}$$

is a family of verifiable random functions with security parameter λ *if there is a tuple of algorithms* $(\text{Gen}, \text{Eval}, \text{Prove}, \text{Verify})$ *such that: the key-generation algorithm* $\text{Gen}(1^\lambda)$ *is a* PPT *algorithm that outputs a pair of keys* (pk, sk)*; the function-evaluator algorithm* $\text{Eval}(sk, x)$ *is a deterministic polynomial-time algorithm that outputs* $f_{sk}(x)$*; the prover algorithm* $\text{Prove}(sk, x)$ *is a deterministic polynomial-time algorithm that outputs a proof of correctness* π*; and the verifier algorithm* $\text{Verify}(pk, x, y, \pi)$ *is a* PPT *algorithm that outputs either* 0 *or* 1.

We require the following properties for the VRF*:*

[1] When $a(\lambda)$ takes the value of $*$, it means the VRF is defined for inputs of all length.

Perfect Completeness: *For any* $\lambda \in \mathbb{N}$*, any* $(pk, sk) \leftarrow \mathrm{Gen}(1^\lambda)$ *and any input* x*, that if* $y = \mathrm{Eval}(sk, x)$*,* $\pi = \mathrm{Prove}(sk, x)$*, then it holds that*

$$\mathrm{Verify}(pk, x, y, \pi) = 1.$$

Uniqueness: *For every* pk, x, y_0, y_1, π_0*, and* π_1 *such that* $y_0 \neq y_1$*, the following holds for either* $i = 0$ *or* $i = 1$*:*

$$\Pr[\mathrm{Verify}(pk, x, y_i, \pi_i) = 1] < \mathrm{negl}(\lambda).$$

Where the probability is taken over the random coins of Verify*.*

Pseudorandomness: *For any* PPT *adversary* $\mathcal{A}$*, there is a negligible function* negl*, such that for any* $\lambda \in \mathbb{N}$*, adversary* $\mathcal{A}$ *has at most* $\mathrm{negl}(\lambda)$ *advantage in the following experiment:*

$\mathrm{Exp}_{\mathrm{VRF}}(\mathcal{A})$*:*

- *A pair of keys* (pk, sk) *are generated by running* $\mathrm{Gen}(1^\lambda)$*.*
- *The adversary* $\mathcal{A}$ *is given input* $1^\lambda, pk$ *and oracle access to* $\mathrm{Eval}(sk, \cdot)$ *and* $\mathrm{Prove}(sk, \cdot)$*, and outputs an* x*.*
- *The experiment chooses a random bit* $b \leftarrow \{0,1\}$*, if* $b = 0$ *computes* $y = \mathrm{Eval}(sk, x)$*, otherwise* $y \leftarrow \{0,1\}^{b(\lambda)}$*, then gives* y *to* $\mathcal{A}$*.*
- *The adversary* $\mathcal{A}$ *continues to have oracle access to* $\mathrm{Eval}(sk, \cdot)$ *and* Prove $(sk, \cdot)$*. Eventually, adversary* $\mathcal{A}$ *outputs a guess* b'*. Let* $\mathcal{Q}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle.*
- *The output of the experiment is defined as follow:*
  1. *If* $x \in \mathcal{Q}$*, the experiment outputs a random bit;*
  2. *If* $x \notin \mathcal{Q}$ *and* $b' = b$*, the experiment outputs* 1*;*
  3. *Otherwise, the experiment outputs* 0*.*

The advantage of $\mathcal{A}$ *is defined by*

$$\mathrm{Adv}_{\mathrm{VRF}}(\mathcal{A}) = |\Pr[\mathrm{Exp}_{\mathrm{VRF}}(\mathcal{A}) = 1] - \frac{1}{2}|.$$

In order not to cause confusion, we write VRF.Gen, VRF.Eval, VRF.Prove, VRF.Verify to denote the Gen, Eval, Prove and Verify algorithms belonging to VRF.

The notion of a verifiable random function (VRF) was introduced by Micali, Rabin, and Vadhan [15]. Known constructions of VRFs are due to [15] based on strong RSA, [14] based on a strong version of the Diffie-Hellman assumption in bilinear groups, [6] based on the sum-free generalized DDH assumption, and [8] based on the bilinear Diffie-Hellman inversion assumption.

2.3 Somewhere Perfectly Binding Hash Function

The notion of Somewhere Perfectly Binding Hash Function(SPB)[2] was introduced by [2], which can be used to create a short digest $h = \mathrm{H}_{\mathrm{hk}}(x)$ of some long input $x = (x[1], \ldots, x[L]) \in \Sigma^L$, where Σ is some alphabet. The hashing

[2] This is a stronger notion, compare with SSB in [11].

key $(\mathrm{hk}, \mathrm{shk}) \leftarrow \mathrm{Gen}(i)$ can be generated by providing a special "binding index" i and this ensures that the hash $h = \mathrm{H}_{\mathrm{hk}}(x)$ is perfectly binding for the i'th symbol. In other words, even though h has many other preimages x' such that $\mathrm{H}_{hk}(x') = h$, all of these preimages agree in the i'th symbol $x'[i] = x[i]$. Moreover, we will be interested in SPB hash function with a 'private local opening' property that allow us to prove that i'th symbol of x takes on some particular value $x[i] = u$ by providing a short opening π. The formal definition is below.

Definition 3 (SPB). *A somewhere perfectly binding hash family with private local opening* SPB *is given by a tuple of algorithms* $(\mathrm{Gen}, \mathrm{Hash}, \mathrm{Open}, \mathrm{Verify})$ *with the following syntax:*

Gen$(1^\lambda, n, i)$**:** [3] *Takes as input a security parameter* 1^λ*, a database size* n *and an index* i*, and outputs a hashing key* hk *and a private key* shk*.*

Hash(hk, x)**:** *Takes as input a hashing key* hk *and a database* x *and outputs a digest* h*.*

Open$(\mathrm{hk}, \mathrm{shk}, x, j)$**:** *Takes as input a hashing key* hk*, a private key* shk*, a database* x *and an index* j *and outputs a witness* π*.*

Verify$(\mathrm{hk}, h, j, u, \pi)$ *Takes as input a hashing key* hk*, a digest* h*, an index* j*, an alphabet* u *and a witness* π*, and outputs either* 0 *or* 1*.*

We require the following properties for the SPB*:*

Perfect Correctness: *For any security parameter* $\lambda \in \mathbb{N}$*, any* $n = \mathrm{poly}(\lambda)$*, any database* x *of size* n *and all index* $i \in [n]$*, that if* $(\mathrm{hk}, \mathrm{shk}) \leftarrow \mathrm{Gen}(1^\lambda, n, i)$*,* $h = \mathrm{Hash}(\mathrm{hk}, x)$ *and* $\pi \leftarrow \mathrm{Open}(\mathrm{hk}, \mathrm{shk}, x, i)$*, then it holds that*

$$\Pr[\mathrm{Verify}(\mathrm{hk}, h, i, x[i], \pi) = 1] = 1.$$

Somewhere Perfectly Binding: *For any security parameter* $\lambda \in \mathbb{N}$*, any* $n = \mathrm{poly}(\lambda)$*, any database* x *of size* n*, any index* $i \in [n]$*, any alphabet value* u *and any witness* π*, that if* $h = \mathrm{Hash}(\mathrm{hk}, x)$ *and* $\mathrm{Verify}(\mathrm{hk}, h, i, u, \pi) = 1$*, then it holds that* $i = \mathrm{ind}$*, and* $u = x[\mathrm{ind}]$*, where* ind *is the index for generating* hk*.*

Index Hiding: *For any* PPT *adversary* $\mathcal{A}$*, there is a negligible function* negl*, such that for any* $\lambda \in \mathbb{N}$*, adversary* $\mathcal{A}$ *has at most* $\mathrm{negl}(\lambda)$ *advantage in the following experiment:*

$\mathrm{Exp}_{\mathrm{IH}}$ $(\mathcal{A})$*:*

- *The adversary* $\mathcal{A}$ *is given input* 1^λ*, and output a triple* (n, i_0, i_1)*, then gives it to experiment.*
- *The experiment chooses a random bit* $b \leftarrow \{0, 1\}$*, and then computes* $(\mathrm{hk}, \mathrm{shk}) \leftarrow \mathrm{Gen}(1^\lambda, n, i_b)$*, then give* hk *to* $\mathcal{A}$*.*
- *The adversary* $\mathcal{A}$ *outputs a guess* b'*.*
- *The output of the experiment is defined to be* 1 *if* $b' = b$*, and* 0 *otherwise.*

The advantage of $\mathcal{A}$ *is defined by*

$$\mathrm{Adv}_{\mathrm{IH}}(\mathcal{A}) = |\Pr[\mathrm{Exp}_{\mathrm{IH}}(\mathcal{A}) = 1] - \frac{1}{2}|.$$

[3] Where we need $i \in [n]$, the same thing has to be true for the following j.

Efficiency: *The hashing keys* hk *generated by* $\mathrm{Gen}(1^\lambda, n, i)$ *and the witness* τ *generated by* $\mathrm{Open}(\mathrm{hk}, \mathrm{shk}, x, j)$ *are of size* $\log(n) \cdot \mathrm{poly}(\lambda)$*. Moreover,* $\mathrm{Verify}(\mathrm{hk}, \mathrm{shk}, i, x, \tau)$ *can be computed by a circuit of size* $\log(n) \cdot \mathrm{poly}(\lambda)$*.*

In order not to cause confusion, we write SPB.Gen, SPB.Hash, SPB.Open, SPB.Verify to denote the Gen, Hash, Open and Verify algorithms belonging to SPB.

Remark 1. To simplify notation, we will not provide the block size of databases as an input to SPB.Gen but rather assume that the block size for the specific application context is hardwired.

Remark 2. We can input every $j \in [n]$ into Open algorithm, but the only j that was used to generate hashing key can produce a valid witness.

The notion of somewhere perfectly binding hash family with private local opening (SPB) was introduced by [2]. In that work, they give a simple black-box transformation from any SPB hash family to a SPB with private local opening. They also show that the DDH-based SSB construction of [17] can be proofed to be SPB hash family.

3 Repudiable Ring Signatures

In this section, we provide the definitions related to repudiable ring signatures. The notion of repudiable ring signatures was introduced by [18].

Definition 4 (RRS). *A repudiable ring signature scheme is a tuple of* PPT *algorithms* $\mathrm{RRS} = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Verify}, \mathrm{Repudiate}, \mathrm{VerRepud})$*, satisfying the five properties of correctness* (Definition 5)*, anonymity* (Definition 6)*, unforgeability* (Definition 7)*, repudiability* (Definition 8) *and repudiation unforgeability* (Definition 9)*. The syntax of* RRS *follows:*

Gen(1^λ)**:** *Takes as input a security parameter* 1^λ*, and outputs a pair* $(\mathrm{VK}, \mathrm{SK})$ *of verification and signing keys.*

Sign$(\mathrm{SK}, m, \mathrm{R})$**:** *Takes as input a signing key* SK*, a message* m*, and a set of verification keys* $\mathrm{R} = (\mathrm{VK}_{i_1}, \ldots, \mathrm{VK}_{i_n})$*, and outputs a signature* σ*. The set* R *is also known as a 'ring'.*

Verify(m, R, σ)**:** *Takes as input a message* m*, a set* $\mathrm{R} = (\mathrm{VK}_{i_1}, \ldots, \mathrm{VK}_{i_n})$ *of verification key, and a signature* σ*, and outputs either* 0 *or* 1*.*

Repudiate$(\mathrm{SK}, m, \mathrm{R}, \sigma)$**:** *Takes as input a signing key* SK*, a message* m*, a set* $\mathrm{R} = (\mathrm{VK}_{i_1}, \ldots, \mathrm{VK}_{i_n})$ *of verification key, and a signature* σ*, and outputs a repudiation* ξ*.*

VerRepud$(\mathrm{VK}, m, \mathrm{R}, \sigma, \xi)$**:** *Takes as input an identity* VK*, a message* m*, a set* $\mathrm{R} = (\mathrm{VK}_{i_1}, \ldots, \mathrm{VK}_{i_n})$ *of verification key, a signature* σ*,and a repudiation* ξ*, and outputs either* 0 *or* 1*.*

Definition 5 (Correctness). *We say that a repudiable ring signature scheme* $\mathrm{RRS} = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Verify}, \mathrm{Repudiate}, \mathrm{VerRepud})$ *satisfies correctness, if there is a negligible function* $\mathrm{negl}(\lambda)$ *such that for any* $\lambda \in \mathbb{N}$*, any* $n = \mathrm{poly}(\lambda)$*, any* n *key pairs* $(\mathrm{VK}_{i_1}, \mathrm{SK}_{i_1}), \ldots, (\mathrm{VK}_{i_n}, \mathrm{SK}_{i_n}) \leftarrow \mathrm{Gen}(1^\lambda)$*, any* $j_0, j_1 \in [n]$*, where* $j_0 \neq j_1$*, and any message* m*, we have*

$$\Pr[\mathrm{Verify}(m, \mathrm{R}, \sigma) = 1] \geq 1 - \mathrm{negl}(\lambda),$$

and

$$\Pr[\mathrm{VerRepud}(\mathrm{VK}_{i_{j_1}}, m, \mathrm{R}, \sigma, \xi) = 1] \geq 1 - \mathrm{negl}(\lambda),$$

where $\mathrm{R} = (\mathrm{VK}_{i_1}, \ldots, \mathrm{VK}_{i_n})$*,* $\sigma = \mathrm{Sign}(\mathrm{SK}_{i_{j_0}}, m, \mathrm{R})$*, and* $\xi = \mathrm{Repudiate}(\mathrm{SK}_{i_{j_1}}, m, \mathrm{R}, \sigma)$.

To give the formal definition of other properties, we need to introduce three oracles first:

- `Corruption oracle`: For a repudiable ring signature scheme RRS, the oracle $\mathrm{OC}_{(\mathrm{VK}_1, \mathrm{SK}_1), \ldots, (\mathrm{VK}_l, \mathrm{SK}_l)}$ is defined to take as input $i \in [l]$, and outputs $(\mathrm{VK}_i, \mathrm{SK}_i)$.
- `Signing oracle`: For a repudiable ring signature scheme RRS, the oracle $\mathrm{OS}_{(\mathrm{VK}_1, \mathrm{SK}_1), \ldots, (\mathrm{VK}_l, \mathrm{SK}_l)}$ is defined to take as input $i \in [l]$, a message m, and a ring R[4], and output $\mathrm{Sign}(\mathrm{SK}_i, m, \mathrm{R})$.
- `Repudiation oracle`: For a repudiable ring signature scheme RRS, the oracle $\mathrm{OR}_{(\mathrm{VK}_1, \mathrm{SK}_1), \ldots, (\mathrm{VK}_l, \mathrm{SK}_l)}$ is defined to take as input $j \in [l]$, a message m, a ring R, and a signature σ, and output $\mathrm{Repudiate}(\mathrm{SK}_j, m, \mathrm{R}, \sigma)$.

In this paper, we refer to adaptive anonymity against adversarially chosen keys, which is slightly different from the previous one in [18].

Definition 6 (Anonymity). *We say that a repudiable ring signature scheme* $\mathrm{RRS} = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Verify}, \mathrm{Repudiate}, \mathrm{VerRepud})$ *satisfies adaptive anonymity against adversarially chosen keys, if for all* PPT *adversary* $\mathcal{A}$*, there is a negligible function* negl *such that, for all* $\lambda \in \mathbb{N}$*, all* $l = \mathrm{poly}(\lambda)$*, it holds that* $\mathcal{A}$ *has at most negligible advantage in the following experiment.*
$\mathrm{Exp}_{\mathrm{Ano}}(\mathcal{A})$*:*

- *For each* $i \in [l]$*, the experiment generates the key pairs* $(\mathrm{VK}_i, \mathrm{SK}_i) \leftarrow \mathrm{Gen}(\lambda)$.
- *The adversary* $\mathcal{A}$ *is given input* 1^λ*,* $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$*, and oracle access to* $\mathrm{OC}(\cdot)$*,* $\mathrm{OS}(\cdot)$*,* $\mathrm{OR}(\cdot)$.
- *The adversary provides a tuple* $(m, \mathrm{R}, j_0, j_1)$ *to experiment, with* $j_0, j_1 \in [l]$*, and* $\mathrm{VK}_{j_0}, \mathrm{VK}_{j_1} \in \mathrm{R}$.
- *The experiment chooses a random bit* $b \leftarrow \{0, 1\}$*, and computes* $\sigma \leftarrow \mathrm{Sign}(\mathrm{SK}_{j_b}, m, \mathrm{R})$*, then gives* σ *to adversary* $\mathcal{A}$.

[4] We allow that the ring R may contain maliciously chosen verification keys that were not included in $\{\mathrm{VK}_1, \ldots, \mathrm{VK}_l\}$. The same thing holds for the ring in the OR.

- *The adversary continues to have oracle access to* $\mathrm{OC}(\cdot)$, $\mathrm{OS}(\cdot)$ *and* $\mathrm{OR}(\cdot)$ *except that* $\mathcal{A}$ *can not query* $\mathrm{OR}(\cdot)$ *with* $(\cdot, m, \mathrm{R}, \cdot)$. *Let* $\mathcal{Q}_{\mathrm{OC}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OC.
- *The adversary* $\mathcal{A}$ *outputs a guess* b'.
- *The output of the experiment is defined as follow :*
  1. *If* $j_0 \in \mathcal{Q}_{\mathrm{OC}}$ *or* $j_1 \in \mathcal{Q}_{\mathrm{OC}}$, *the experiment outputs a random bit.*
  2. *If* $j_0, j_1 \notin \mathcal{Q}_{\mathrm{OC}}$ *and* $b' = b$, *the experiment outputs* 1.
  3. *Otherwise, the experiment outputs* 0.

The advantage of $\mathcal{A}$ *is defined by*

$$\mathrm{Adv}_{\mathrm{Ano}}(\mathcal{A}) = |\Pr[\mathrm{Exp}_{\mathrm{Ano}}(\mathcal{A}) = 1] - \frac{1}{2}|.$$

Remark 3. We allow that ring R chosen by adversary $\mathcal{A}$ in step 3 may contain maliciously chosen verification keys that were not generated by challenger.

Definition 7 (Unforgeability). *We say that a repudiable ring signature construction* $\mathrm{RRS} = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Verify}, \mathrm{Repudiate}, \mathrm{VerRepud})$ *is unforgeable with respect to insider corruption, if for all* PPT *adversary* $\mathcal{A}$, *there is a negligible function* negl *such that , for all* $\lambda \in \mathbb{N}$, *all* $l = \mathrm{poly}(\lambda)$, *it holds that* $\mathcal{A}$ *has at most* $\mathrm{negl}(\lambda)$ *advantage in the following experiment.*
$\mathrm{Exp}_{\mathrm{Unf}}(\mathcal{A})$*:*

- *For each* $i \in [l]$, *the experiment generates the key pairs* $(\mathrm{VK}_i, \mathrm{SK}_i) \leftarrow \mathrm{Gen}(\lambda)$.
- *The adversary* $\mathcal{A}$ *is given input* 1^λ, $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$, *and oracle access to* $\mathrm{OC}(\cdot)$, $\mathrm{OS}(\cdot)$, $\mathrm{OR}(\cdot)$. *The adversary* $\mathcal{A}$ *outputs* (m, R, σ). *Let* $\mathcal{Q}_{\mathrm{OC}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OC, *and let* $\mathcal{Q}_{\mathrm{OS}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OS.
- *The output of the experiment is defined to be* 1 *if it satisfies the following conditions:*
  1. $\mathrm{Verify}(m, \mathrm{R}, \sigma) = 1$,
  2. $\mathrm{R} \subset \{\mathrm{VK}_1, \ldots, \mathrm{VK}_l\} \setminus \mathcal{Q}_{\mathrm{OC}}$[5],
  3. $(\cdot, m, \mathrm{R}) \notin \mathcal{Q}_{\mathrm{OS}}$.

  Otherwise, the experiment is defined by 0.

The advantage of $\mathcal{A}$ *is defined by*

$$\mathrm{Adv}_{\mathrm{Unf}}(\mathcal{A}) = \Pr[\mathrm{Exp}_{\mathrm{Unf}}(\mathcal{A}) = 1].$$

Definition 8 (Repudiability). *We say that a repudiable ring signature construction* $\mathrm{RRS} = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Verify}, \mathrm{Repudiate}, \mathrm{VerRepud})$ *is repudiable with respect to insider corruption, if for all* PPT *adversary* $\mathcal{A}$, *there is a negligible function* negl *such that, for all* $\lambda \in \mathbb{N}$, *all* $l = \mathrm{poly}(\lambda)$, *it holds that* $\mathcal{A}$ *has at most* $\mathrm{negl}(\lambda)$ *advantage in the following experiments.*
1. $\mathrm{Exp}_{\mathrm{Rep1}}(\mathcal{A})$ *(Non-signer can repudiate experiment):*

[5] With a slight abuse of notation, we sometimes use $\mathcal{Q}_{\mathrm{OC}}$ to denote the set of verification keys $\{\mathrm{VK}_{j_1}, \ldots, \mathrm{VK}_{j_s}\}$ where $\{j_k | k = 1, \ldots, s\}$ were all the queries that $\mathcal{A}$ asked its oracle OC.

- *For each* $i \in [l]$*, the experiment generates the key pairs* $(\mathrm{VK}_i, \mathrm{SK}_i) \leftarrow \mathrm{Gen}(\lambda)$.
- *The adversary* $\mathcal{A}$ *is given input* 1^λ*,* $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$*, and oracle access to* $\mathrm{OC}(\cdot)$*,* $\mathrm{OS}(\cdot)$*,* $\mathrm{OR}(\cdot)$*. The adversary* $\mathcal{A}$ *outputs* (m, R, σ) *with* $\mathrm{R} \subset \{\mathrm{VK}_1, \ldots, \mathrm{VK}_l\}$*. Let* $\mathcal{Q}_{\mathrm{OC}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OC*, let* $\mathcal{Q}_{\mathrm{OS}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OS*, and let* $\mathcal{Q}_{\mathrm{OR}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OR.
- *The experiment compute:*

$$\xi_{i_k} = \mathrm{Repudiate}(\mathrm{SK}_{i_k}, m, \mathrm{R}, \sigma), \text{ and } b_{i_k} = \mathrm{VerRepud}(\mathrm{VK}_{i_k}, m, \mathrm{R}, \sigma, \xi_{i_k}),$$

  for all $\mathrm{VK}_{i_k} \in \mathrm{R} \setminus \mathcal{Q}_{\mathrm{OC}}$.
- *Finally, the output of the experiment is defined to be* 1 *if it satisfies the following condition:*
  1. $(\cdot, m, \mathrm{R}) \notin \mathcal{Q}_{\mathrm{OS}}$ *and* $(\cdot, m, \mathrm{R}, \cdot) \notin \mathcal{Q}_{\mathrm{OR}}$*,*
  2. $\mathrm{Verify}(m, \mathrm{R}, \sigma) = 1$*,*
  3. $\mathrm{R} \setminus \mathcal{Q}_{\mathrm{OC}} \neq \emptyset$*,*
  4. $\bigwedge_{\mathrm{VK}_j \in \mathrm{R} \setminus \mathcal{Q}_{\mathrm{OC}}} b_j = 0$.

  Otherwise, the experiment is defined by 0.

The advantage of $\mathcal{A}$ *is defined by*

$$\mathrm{Adv}_{\mathrm{Rep1}}(\mathcal{A}) = \Pr[\mathrm{Exp}_{\mathrm{Rep1}}(\mathcal{A}) = 1].$$

2. $\mathrm{Exp}_{\mathrm{Rep2}}(\mathcal{A})$*(Signer cannot repudiate):*

- *For each* $i \in [l]$*, the experiment generates the key pairs* $(\mathrm{VK}_i, \mathrm{SK}_i) \leftarrow \mathrm{Gen}(\lambda)$.
- *The adversary* $\mathcal{A}$ *is given input* 1^λ*,* $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$*, and oracle access to* $\mathrm{OC}(\cdot)$*,* $\mathrm{OS}(\cdot)$*,* $\mathrm{OR}(\cdot)$*. Let* $\mathcal{Q}_{\mathrm{OC}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OC*, let* $\mathcal{Q}_{\mathrm{OS}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OS*. The adversary* $\mathcal{A}$ *outputs* $(m, \mathrm{R}, \sigma, \{\xi_{i_k}\}_{\mathrm{VK}_{i_k} \in \mathcal{Q}_{\mathrm{OC}} \cap \mathrm{R}})$ *with* $\mathrm{R} \subset \{\mathrm{VK}_1, \ldots, \mathrm{VK}_l\}$.
- *The experiment compute* $b_{i_k} = \mathrm{VerRepud}(\mathrm{VK}_{i_k}, m, \mathrm{R}, \sigma, \xi_{i_k})$ *for all* $\mathrm{VK}_{i_k} \in \mathcal{Q}_{\mathrm{OC}} \cap \mathrm{R}$.
- *The output of the experiment is defined to be* 1 *if it satisfies the following condition:*
  1. $(\cdot, m, \mathrm{R}) \notin \mathcal{Q}_{\mathrm{OS}}$*,*
  2. $\mathrm{Verify}(m, \mathrm{R}, \sigma) = 1$*,*
  3. $\mathcal{Q}_{\mathrm{OC}} \cap \mathrm{R} \neq \emptyset$*,*
  4. $\bigwedge_{\mathrm{VK}_j \in \mathcal{Q}_{\mathrm{OC}} \cap \mathrm{R}} b_j = 1$.

  Otherwise, the experiment is defined by 0.

The advantage of $\mathcal{A}$ *is defined by*

$$\mathrm{Adv}_{\mathrm{Rep2}}(\mathcal{A}) = \Pr[\mathrm{Exp}_{\mathrm{Rep2}}(\mathcal{A}) = 1].$$

Furthermore, we also need that no one can forge other people's valid repudiation, and this is a new requirement we proposed.

Definition 9 (Repudiation Unforgeability). *We say that a repudiable ring signature scheme* $\mathrm{RRS} = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Verify}, \mathrm{Repudiate}, \mathrm{VerRepud})$ *satisfies repudiation unforgeability against adversarially chosen messages and keys, if for all* PPT *adversary* $\mathcal{A}$*, there is a negligible function* negl *such that, for all* $\lambda \in \mathbb{N}$*, all* $l = \mathrm{poly}(\lambda)$*, it holds that* $\mathcal{A}$ *has at most negligible advantage in the following experiment.*
$\mathrm{Exp}_{\mathrm{Reun}}(\mathcal{A})$*(Nobody cannot forge others repudiation experiment):*

- *For each* $i \in [l]$*, the experiment generates the key pairs* $(\mathrm{VK}_i, \mathrm{SK}_i) \leftarrow \mathrm{Gen}(\lambda)$.
- *The adversary* $\mathcal{A}$ *is given input* 1^λ*,* $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$*, and oracle access to* $\mathrm{OC}(\cdot)$*,* $\mathrm{OS}(\cdot)$*,* $\mathrm{OR}(\cdot)$.
- *The adversary provides a tuple* (m, R, j) *to experiment, with* $j \in [l]$*, and* $\mathrm{VK}_j \in \mathrm{R}$.
- *The experiment computes* $\sigma \leftarrow \mathrm{Sign}(\mathrm{SK}_j, m, \mathrm{R})$*, then give* σ *to adversary* $\mathcal{A}$.
- *The adversary continues to have oracle access to* $\mathrm{OC}(\cdot)$*,* $\mathrm{OS}(\cdot)$*. Let* $\mathcal{Q}_{\mathrm{OC}}$ *denote the set of all queries that* $\mathcal{A}$ *asked its oracle* OC.
- *Then the adversary* $\mathcal{A}$ *outputs* (j', ξ).
- *The output of the experiment is defined to* 1 *if it satisfies the following conditions:*
  1. $j' \notin \mathcal{Q}_{\mathrm{OC}}$,
  2. $\mathrm{VerRepud}(\mathrm{VK}_{j'}, m, \mathrm{R}, \sigma, \xi) = 1$.

  Otherwise, the experiment is defined by 0.

The advantage of $\mathcal{A}$ *is defined by*

$$\mathrm{Adv}_{\mathrm{Reun}}(\mathcal{A}) = \Pr[\mathrm{Exp}_{\mathrm{Reun}}(\mathcal{A}) = 1].$$

Remark 4. Here, we consider a stronger requirement that the adversary even can know the signing key of signature σ, they still cannot forge a valid repudiation.

In order not to cause confusion, we write RRS.Gen, RRS.Sign, RRS.Verify, RRS.Repudiate, RRS.VerRepud to denote the Gen, Sign, Verify, Repudiate and VerRepud algorithms belonging to RRS.

4 Construction of Repudiable Ring Signatures

In this section we will provide a construction of a repudiable ring signature. Our construction RRS is parameterized by NIWI, VRF and SPB, where:

- NIWI = (Prove, Verify) be a NIWI-proof system for a language $\mathcal{L}$.
- VRF = (Gen, Eval, Prove, Verify) be a verifiable random function with input domain $\{0,1\}^*$, output range $\{0,1\}^{\alpha(\lambda)}$.
- SPB = (Gen, Hash, Open, Verify) be a somewhere perfectly binding hash function with private local opening.

In the rest of the section, we use the following convention to parse a ring R, write

$$\mathrm{R} = \{\mathrm{VK}_{i_1}, \ldots, \mathrm{VK}_{i_n}\},$$

where $\mathrm{VK}_{i_k} = (pk^0_{i_k}, pk^1_{i_k})$. And we use $\mathrm{R}[k]$ denote k'th verification key VK_{i_k} in R.

We first present two languages, which will be used in the scheme latter.

Definition 10. *Let $\mathcal{L}_1$ be the following language:*

$$(m, \varphi, y^0_0, y^1_0, y^0_1, y^1_1, \mathrm{hk}_0, \mathrm{hk}_1, h_0, h_1) \in \mathcal{L}_1,$$

if and only if there are $\mathrm{VK} = (pk^0, pk^1)$*, index i, η, τ^0 τ^1 and $j \in \{0,1\}$ subject to*

$$\mathrm{SPB.Verify}(\mathrm{hk}_j, h_j, i, \mathrm{VK}, \eta) = 1;$$

and

$$\mathrm{VRF.Verify}(pk^0, (h_j, m, \varphi), y^0_j, \tau^0) = 1;$$

and

$$\mathrm{VRF.Verify}(pk^1, (h_j, m, \varphi), y^1_j, \tau^1) = 1.$$

Remark 5. This language is used to produce signature.

Definition 11. *Let $\mathcal{L}_2$ be the following language:*

$$(\mathrm{VK}, m, \varphi, y^1_0, y^1_1, h_0, h_1, z_0, z_1) \in \mathcal{L}_2,$$

if and only if there are $y'_0, y'_1 \in \{0,1\}^{\alpha(\lambda)}$ and $\tau_{00}, \tau_{01}, \tau_{10}, \tau_{11}$ subject to

$$(y'_0 \neq y^1_0) \wedge (y'_1 \neq y^1_1) = 1;$$

and

$$\mathrm{VRF.Verify}(pk^1, (h_0, m, \varphi), y'_0, \tau_{00}) = 1, \quad \mathrm{VRF.Verify}(pk^1, y'_0, z_0, \tau_{01}) = 1;$$

and

$$\mathrm{VRF.Verify}(pk^1, (h_1, m, \varphi), y'_1, \tau_{10}) = 1, \quad \mathrm{VRF.Verify}(pk^1, y'_1, z_1, \tau_{11}) = 1.$$

Remark 6. This language is used to produce repudiation, notice that we only use the second private key in this language.

4.1 Construction

Our repudiable ring signature schemes RRS = (Gen, Sign, Verify, Repudiate, VerRepud) is given as follows.

$\mathrm{RRS.Gen}(1^\lambda)$:

1. (pk^0, sk^0), $(pk^1, sk^1) \leftarrow \mathrm{VRF.Gen}(1^\lambda)$.

2. Output VK $= (pk^0, pk^1)$ and SK $= (sk^0, sk^1, \text{VK})$.

We include the verification key VK in SK so that the Sign procedure can identify the verification key in the ring corresponding to the signing key.

RRS.Sign(SK, m, R):

1. Parse R as described above and SK$=(sk^0, sk^1,$ VK).
2. If VK $\notin$ R, output $\perp$ and halt.
3. Define $i^* \in [N]$ such that $\text{R}[i^*] = \text{VK}$.
4. Compute $(\text{hk}_0, \text{shk}_0)$, $(\text{hk}_1, \text{shk}_1) \leftarrow \text{SPB.Gen}(1^\lambda, |\text{R}|, i^*)$.
5. Compute $h_0 = \text{SPB.Hash}(\text{hk}_0, \text{R})$, $h_1 = \text{SPB.Hash}(\text{hk}_1, \text{R})$
6. Compute $\eta \leftarrow \text{SPB.Open}(\text{hk}_0, \text{shk}_0, \text{R}, i^*)$.
7. Compute $\varphi \leftarrow \{0,1\}^\lambda$.
8. Compute $y_0^0 = \text{VRF.Eval}(sk^0, (h_0, m, \varphi))$, $\tau^0 = \text{VRF.Prove}(sk^0, (h_0, m, \varphi))$.
9. Compute $y_0^1 = \text{VRF.Eval}(sk^1, (h_0, m, \varphi))$, $\tau^1 = \text{VRF.Prove}(sk^1, (h_0, m, \varphi))$.
10. Compute $y_1^0 \leftarrow \{0,1\}^{\alpha(\lambda)}$, $y_1^1 \leftarrow \{0,1\}^{\alpha(\lambda)}$.
11. Set $x = (m, \varphi, y_0^0, y_0^1, y_1^0, y_1^1, \text{hk}_0, \text{hk}_1, h_0, h_1)$ and $w = (\text{VK}, i^*, \eta, \tau^0, \tau^1, 0)$.
12. Compute $\pi \leftarrow \text{NIWI.Prove}_{\mathcal{L}_1}(x, w)$.
13. Output $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \text{hk}_0, \text{hk}_1, \pi)$.

RRS.Verify(m, R, σ):

1. Parse R as above and $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \text{hk}_0, \text{hk}_1, \pi)$.
2. Compute $h_0' = \text{SPB.Hash}(\text{hk}_0, \text{R})$, $h_1' = \text{SPB.Hash}(\text{hk}_1, \text{R})$.
3. Output $\text{NIWI.Verify}_{\mathcal{L}_1}((m, \varphi, y_0^0, y_0^1, y_1^0, y_1^1, \text{hk}_0, \text{hk}_1, h_0', h_1'), \pi)$.

The above is our ring signature algorithm, which can be used separately. Now we proceed to describe the repudiation algorithms for RRS.

RRS.Repudiate(SK, m, R, σ):

1. Parse R as above, SK$= (sk^0, sk^1,$ VK), and $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \text{hk}_0, \text{hk}_1, \pi)$.
2. If VK $\notin$ R, output $\perp$ and halt.
3. Compute $b = \text{RRS.Verify}(m, \text{R}, \sigma)$, if $b = 0$ output $\perp$ and halt.
4. Compute $h_0 = \text{SPB.Hash}(\text{hk}_0, \text{R})$, $h_1 = \text{SPB.Hash}(\text{hk}_1, \text{R})$.
5. Compute $y_0' = \text{VRF.Eval}(sk^1, (h_0, m, \varphi))$, $y_1' = \text{VRF.Eval}(sk^1, (h_1, m, \varphi))$.
6. If $y_0' = y_0^1$ or $y_1' = y_1^1$, output $\perp$ and halt.
7. Compute $\tau_{00} = \text{VRF.Prove}(sk^1, (h_0, m, \varphi))$, $\tau_{10} = \text{VRF.Prove}(sk^1, (h_1, m, \varphi))$.
8. Compute $z_0 = \text{VRF.Eval}(sk^1, y_0')$, $z_1 = \text{VRF.Eval}(sk^1, y_1')$.
9. Compute $\tau_{01} = \text{VRF.Prove}(sk^1, y_0')$, $\tau_{11} = \text{VRF.Prove}(sk^1, y_1')$.
10. Set $x = (\text{VK}, m, \varphi, y_0^1, y_1^1, h_0, h_1, z_0, z_1)$, $w = (y_0', y_1', \tau_{00}, \tau_{01}, \tau_{10}, \tau_{11})$.
11. Compute $\pi' \leftarrow \text{NIWI.Prove}_{\mathcal{L}_2}(x, w)$.
12. Output $\xi = (z_0, z_1, \pi')$.

RRS.VerRepud(VK, m, R, σ, ξ):

1. Parse R as above, $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \text{hk}_0, \text{hk}_1, \pi)$, and $\xi = (z_0, z_1, \pi')$.
2. If VK $\notin$ R, output 1 and halt.
3. Compute $b = \text{RRS.Verify}(m, \text{R}, \sigma)$, if $b = 0$ output 1 and halt.
4. Compute $h_0' = \text{SPB.Hash}(\text{hk}_0, \text{R})$, $h_1' = \text{SPB.Hash}(\text{hk}_1, \text{R})$.
5. Output $\text{NIWI.Verify}_{\mathcal{L}_2}((\text{VK}, m, \varphi, y_0^1, y_1^1, h_0', h_1', z_0, z_1), \pi')$.

4.2 Signature and Repudiation Size

We will first show that our scheme has only logarithmic-size signatures and repudiations.

For a signature $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, \pi)$, the size of $\varphi, y_0^0, y_0^1, y_1^0, y_1^1$ is $\mathrm{poly}(\lambda)$ and independent of the ring-size n. And since SPB is efficient, we have $\mathrm{hk}_0, \mathrm{hk}_1$ is bounded by $\log(n) \cdot \mathrm{poly}(\lambda)$. Also by the efficiency of SPB the size of the witness τ is $\log(n) \cdot \mathrm{poly}(\lambda)$ and the SPB verification function SPB.Verify can be computed by a circuit of size $\log(n) \cdot \mathrm{poly}(\lambda)$. Therefore, the verification circuit C_x for the language $\mathcal{L}_1$ and statement $x = (m, \varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, h_0, h_1)$ has size $\log(n) \cdot \mathrm{poly}(\lambda)$. By the proof-size property of the NIWI proof it holds that $|\pi| = |C_x| \cdot \mathrm{poly}(\lambda) = \log(n) \cdot \mathrm{poly}(\lambda)$. Consequently, the size of signatures σ is $\log(n) \cdot \mathrm{poly}(\lambda)$.

For a repudiation $\xi = (z_0, z_1, \pi')$, the size of z_0, z_1 is $\mathrm{poly}(\lambda)$ and independent of the ring-size n. Using the same analysis we can also get that the size of proof π' is $\log(n) \cdot \mathrm{poly}(\lambda)$. Consequently, the size of repudiations ξ is $\log(n) \cdot \mathrm{poly}(\lambda)$.

4.3 Correctness

We now show that our scheme satisfies correctness.

Lemma 1. *The repudiable ring signature scheme* RRS *is correct, given that* NIWI *has perfect completeness,* VRF *has perfect completeness and pseudorandomness, and* SPB *has perfect correctness.*

Proof. Assume that (VK, SK) were generated by $\mathrm{RRS.Gen}(1^\lambda)$, and for any signature $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, \pi)$ is the output of $\mathrm{RRS.Sign}(\mathrm{SK}, m, \mathrm{R})$, where $\mathrm{R} = (\mathrm{VK}_1, \ldots \mathrm{VK}_n)$ is a ring generated by $\mathrm{RRS.Gen}(1^\lambda)$, and $\mathrm{R}[i] = \mathrm{VK}$. We will show that it holds $\mathrm{RRS.Verify}(m, \mathrm{R}, \sigma) = 1$. Because SPB.Hash is a deterministic algorithm, it holds $h_0' = h_0$ and $h_1' = h_1$. According to the RRS.Sign algorithm we have defined and the correctness of SPB, we have there is a η, such that it holds that

$$\mathrm{SPB.Verify}(\mathrm{hk}_0, h_0, i, \mathrm{VK}, \eta) = 1,$$

Moreover, by the definition of $y_0^0,\ y_0^1$, and completeness of VRF, there are $\tau^0,\ \tau^1$ such that it holds

$$\mathrm{VRF.Verify}(pk^0, (h_0, m, \varphi), y_0^0, \tau^0) = 1, \mathrm{VRF.Verify}(pk^1, (h_0, m, \varphi), y_0^1, \tau^1) = 1.$$

Therefore, $(m, \varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, h_0, h_1) \in \mathcal{L}_1$ and $w = (\mathrm{VK}, i, \eta, \tau^0, \tau^1, 0)$ is a witness for the membership. Thus, by the correctness of NIWI it holds that

$$\mathrm{NIWI.Verify}_{\mathcal{L}_1}((m, \varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, h_0, h_1), \pi) = 1.$$

and consequently $\mathrm{RRS.Verify}(m, \mathrm{R}, \sigma) = 1$.

Furthermore, assume $(\mathrm{VK}_j, \mathrm{SK}_j)$ were generated by $\mathrm{RRS.Gen}(1^\lambda)$, where $\mathrm{R}[j] = \mathrm{VK}_j$ and $\mathrm{VK} \neq \mathrm{VK}_j$, and $\xi = (z_0, z_1, \pi')$ is the output of RRS.Repudiate

$(\mathrm{SK}_j, m, \mathrm{R}, \sigma)$. We will show that it holds $\mathrm{RRS.VerRepud}(\mathrm{VK}_j, m, \mathrm{R}, \sigma, \xi) = 1$. Because SPB.Hash is a deterministic algorithm, it holds $h'_0 = h_0$ and $h'_1 = h_1$. Since RRS.Gen is a random algorithm, and $\mathrm{VK} \neq \mathrm{VK}_j$, it holds that $sk^1 \neq sk^1_j$. And by the pseudorandomness of VRF, we have that there is a negligible function $\mathrm{negl}(\lambda)$, a $y'_0 = \mathrm{VRF.Eval}(sk^1_j, (h_0, m, \varphi))$, and a $y'_1 = \mathrm{VRF.Eval}(sk^1_j, (h_1, m, \varphi))$, such that

$$\Pr[(y'_0 \neq y^1_0) \wedge (y'_1 \neq y^1_1)] \geq 1 - \mathrm{negl}(\lambda).$$

And by the correctness of the VRF, there are $\tau_{00}, \tau_{01}, \tau_{10}$ and τ_{11} such that it holds that

$$\mathrm{VRF.Verify}(pk^1_j, (h_0, m, \varphi), y'_0, \tau_{00}) = 1, \quad \mathrm{VRF.Verify}(pk^1_j, y'_0, z_0, \tau_{01}) = 1;$$

$$\mathrm{VRF.Verify}(pk^1_j, (h_1, m, \varphi), y'_1, \tau_{10}) = 1, \quad \mathrm{VRF.Verify}(pk^1_j, y'_1, z_1, \tau_{11}) = 1.$$

Therefore, $(\mathrm{VK}_j, m, \varphi, y^1_0, y^1_1, h_0, h_1, z_0, z_1) \in \mathcal{L}_2$ holds with overwhelming probability, and when it holds, $w = (y'_0, y'_1, \tau_{00}, \tau_{01}, \tau_{10}, \tau_{11})$ is a witness for the membership. Thus, in this case by the correctness of NIWI it holds that

$$\mathrm{NIWI.Verify}_{\mathcal{L}_2}((\mathrm{VK}_j, m, \varphi, y^1_0, y^1_1, h_0, h_1, z_0, z_1), \pi') = 1.$$

And consequently $\Pr[\mathrm{RRS.VerRepud}(\mathrm{VK}_j, m, \mathrm{R}, \sigma, \xi) = 1] \geq 1 - \mathrm{negl}(\lambda)$.

Remark 7. Definition 5 considers only for honestly generated keys. We can also consider a stronger requirement that verify be successful for honestly generated signatures with respect to rings containing adversarial keys. And we can easily proof that our construction also satisfy this stronger requirement by the same way.

4.4 Repudiation unforgeability

We will first turn to establishing repudiation unforgeability of RRS. This is the a new requirement we proposed.

Lemma 2. *The repudiable ring signature scheme* RRS *satisfies repudiation unforgeability, given that* NIWI *has perfect soundness,* VRF *has completeness, uniqueness, and pseudorandomness.*

The main idea of the proof is that, if $\mathcal{A}$ can produce a valid repudiation, then by the perfect soundness of NIWI proof, it must has y'_0, τ_{00}, τ_{01}, such that it holds $\mathrm{VRF.Verify}(pk^1, (h_0, m, \varphi), y'_0, \tau_{00}) = 1$, and $\mathrm{VRF.Verify}(pk^1, y'_0, z_0, \tau_{01}) = 1$. Since VRF has completeness and uniqueness, we have $y'_0 = \mathrm{VRF.Eval}(pk^1, (h_0, m, \varphi))$, and $z_0 = \mathrm{VRF.Eval}(pk^1, y'_0)$, and by this we can attack the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Proof. Suppose, for contradiction, that RRS is not repudiation unforgeability, then there exist some $l = \mathrm{ploy}(\lambda)$, and some PPT adversary $\mathcal{A}$ such that $\mathcal{A}$ has non-negligible advantage in experiment $\mathrm{Exp}_{\mathrm{Reun}}$. Now, we will use this adversary $\mathcal{A}$ to build an adversary $\mathcal{B}$ that break experiment $\mathrm{Exp}_{\mathrm{VRF}}$:

Adversary $\mathcal{B}$: given input 1^λ, pk, and access to oracle $\mathrm{Eval}(sk, \cdot)$, $\mathrm{Prove}(sk, \cdot)$.

1. The adversary $\mathcal{B}$ runs $\mathrm{RRS.Gen}(1^\lambda)$, and generates l pairs of keys.
2. The adversary $\mathcal{B}$ choose a random index $i^* \in [l]$, and let $\mathrm{VK}_{i^*} = (pk^0_{i^*}, pk)$, instead of $\mathrm{VK}_{i^*} = (pk^0_{i^*}, pk^1_{i^*})$. Then $\mathcal{B}$ gives 1^λ and $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$ to $\mathcal{A}$.
3. When $\mathcal{A}$ queries its corruption oracle $\mathrm{OC}(\cdot)$ on $i \in [l]$, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ outputs SK_i;
   - If $i = i^*$, then $\mathcal{B}$ outputs a random string x to challenger, and then outputs a random bit $b' \leftarrow \{0,1\}$ and aborts.
4. When $\mathcal{A}$ queries its signing oracle $\mathrm{OS}(\cdot)$ on $i \in [l]$, m, and R, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ runs the honest signing algorithm RRS.Sign and outputs $\mathrm{RRS.Sign}(\mathrm{SK}_i, m, \mathrm{R})$;
   - If $i = i^*$, then $\mathcal{B}$ runs the honest signing algorithm RRS.Sign with the following modification: in step 9, instead of using sk to generate y^1_0 and τ^1, $\mathcal{B}$ generates these by invoking its VRF oracle.
5. When $\mathcal{A}$ queries it repudiation oracle $\mathrm{OR}(\cdot)$ on $i \in [l]$, m, R, and σ, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ runs the honest repudiating algorithm RRS.Repudiate and outputs $\mathrm{RRS.Repudiate}(\mathrm{SK}_i, m, \mathrm{R}, \sigma)$;
   - If $i = i^*$, then $\mathcal{B}$ runs the honest repudiating algorithm RRS.Repudiate with the following modification: in step 5, step 7, step 8, and step 9, instead of using sk to generate y'_0, y'_1, τ_{00}, τ_{10}, z_0, z_1, and τ_{01}, τ_{11}, $\mathcal{B}$ generates these by invoking its VRF oracle.
6. The adversary $\mathcal{A}$ outputs (m, R, j) to $\mathcal{B}$, and $\mathcal{B}$ answers in the following way:
   - If $j \neq i^*$, then $\mathcal{B}$ runs the honest signing algorithm RRS.Sign and outputs $\mathrm{RRS.Sign}(\mathrm{SK}_j, m, \mathrm{R})$ to $\mathcal{A}$;
   - If $j = i^*$, then $\mathcal{B}$ outputs a random bits x to challenger, and then outputs a random bit $b' \leftarrow \{0,1\}$ and aborts.
7. The adversary $\mathcal{A}$ continues to queries $\mathrm{OC}(\cdot)$ and $\mathrm{OS}(\cdot)$, and $\mathcal{B}$ answers queries in the same way described above.
8. Then the adversary $\mathcal{A}$ outputs (j', ξ). If $j' \neq i^*$, then $\mathcal{B}$ outputs a random bits x to challenger, and then outputs a random bit $b' \leftarrow \{0,1\}$ and aborts.
9. $\mathcal{B}$ parse $\sigma = (\varphi, y^0_0, y^1_0, y^0_1, y^1_1, \mathrm{hk}_0, \mathrm{hk}_1, \pi)$ and $\xi = (z_0, z_1, \pi')$, and do:
   - Compute $h_0 = \mathrm{SPB.Hash}(\mathrm{hk}_0, \mathrm{R})$;
   - $\mathcal{B}$ queries its oracle $\mathrm{Eval}(\cdot)$ on (h_0, m, φ) and get x.
   - $\mathcal{B}$ submits x to the VRF challenger and then receives responses y. If $y = z_0$, $\mathcal{B}$ outputs 0. Otherwise, $\mathcal{B}$ outputs a random bit.

It remains to show that the adversary $\mathcal{B}$ has non-negligible advantage in experiment $\mathrm{Exp}_{\mathrm{VRF}}$. Let us first consider the probability that $\mathcal{A}$ queries oracle $\mathrm{OC}(\cdot)$ for input i^*, or $\mathcal{A}$ outputs (m, R, i^*). Recall that this event causes $\mathcal{B}$ to abort and outputs a random bit. The distribution (i.e., verification keys and oracle responses) of the view of $\mathcal{A}$ is unaffected by $\mathcal{B}$'s choice of i^*, until the point at which $\mathcal{A}$ submits an oracle query to oracle OC for input i^*, or $\mathcal{A}$ outputs (m, R, i^*). The condition $j' \notin \mathcal{Q}_{\mathrm{OC}}$ in the experiment $\mathrm{Exp}_{\mathrm{Reun}}$ ensures that if $\mathcal{A}$ wins the game with non-negligible probability, then $\mathcal{A}$ leaves one or more

keys uncorrupted with at least non-negligible probability. Since i^* is chosen at random by $\mathcal{B}$, it follows that $\Pr[(i^* \notin \mathcal{Q}_{\mathrm{OC}}) \wedge (i^* \neq j) = 1]$ is non-negligible. Let $\mathcal{E}_1$ denote the event that $\mathcal{A}$ does not corrupt i^* and $j \neq i^*$. When $\mathcal{E}_1$ does not occur, then by the definition of $\mathcal{B}$, it will outputs a random bit, and in this time, $\mathcal{B}$ will have exactly $\frac{1}{2}$ advantage in the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Condition on $\mathcal{E}_1$ occurs, since $\mathcal{A}$ does not query corruption oracle on i^* and $j \neq i^*$, the view of $\mathcal{A}$ is identical to the view in experiment $\mathrm{Exp}_{\mathrm{Reun}}$. It follows that $\mathcal{A}$ will win the game with non-negligible probability by assumption.

Now, let us consider the probability that $j' = i^*$, condition on $\mathcal{E}_1$ occurs and $\mathcal{A}$ win the game. As also observed above, the distribution of the view of $\mathcal{A}$ is unaffected by $\mathcal{B}$'s choice of i^*, until the point at which $\mathcal{A}$ submits an oracle query to oracle OC for input i^* or $j = i^*$. Since i^* is chosen at random by $\mathcal{B}$, and $i^*, j' \in \mathrm{R}$, then in this situation we have $\Pr[i^* = j']$ must be non-negligible. Let $\mathcal{E}_2$ denote the event that $\mathcal{E}_1$ occurs and $\mathcal{A}$ win the game and $j' = i^*$. Then according to the above discussion, we have that $\mathcal{E}_2$ occurs with non-negligible probability.

If $\mathcal{E}_2$ occurs, then ξ will be a valid repudiation of i^* respect to σ, i.e.

$$\mathrm{RRS.VerRepud}(\mathrm{VK}_{i^*}, m, \mathrm{R}, \sigma, \xi) = 1.$$

Since by the perfect soundness of NIWI, there exists y_0', and τ_{00}, τ_{01}, s.t.

$$\mathrm{VRF.Verify}(pk, (h_0, m, \varphi), y_0', \tau_{00}) = 1, \quad \mathrm{VRF.Verify}(pk, y_0', z_0, \tau_{01}) = 1.$$

And since VRF has completeness and uniqueness, we have

$$x = y_0' = \mathrm{VRF.Eval}(sk, (h_0, m, \varphi)) \quad \text{and} \quad z_0 = \mathrm{VRF.Eval}(sk, y_0').$$

In this case, if the VRF challenger's bit $b = 0$, then we have $y = z_0$. Recall that this is the trigger condition for $\mathcal{B}$ to output 0. If the VRF challenger's bit $b = 1$, then y is turly random strings. Thus, by the definition of $\mathcal{B}$, $\mathcal{B}$ outputs a random bit with overwhelm probability.

Furthermore, let us consider when $\mathcal{E}_2$ occurs, the probability that $\mathcal{B}$ queries oracle $\mathrm{Eval}(sk, \cdot)$ or $\mathrm{Prove}(sk, \cdot)$ for input $x = \mathrm{VRF.Eval}(sk, (h_0, m, \varphi))$. $\mathcal{B}$ queries its oracle only when $\mathcal{A}$ queries oracle respect input i^*. When $\mathcal{A}$ queries its signature oracle, $\mathcal{B}$ will chooses a random strings φ, so there is only negligible probability such that $\mathcal{B}$ meets x. And since σ is generated by $\mathcal{B}$, and after signature produced $\mathcal{A}$ cannot query its repudiation oracle, there is only negligible probability such that $\mathcal{A}$ queries a repudiation oracle with a signature which include φ. Therefore, there is only a negligible probability such that $\mathcal{B}$ queries oracle $\mathrm{Eval}(sk, \cdot)$ or $\mathrm{Prove}(sk, \cdot)$ for input x.

Finally, we consider when $\mathcal{E}_1$ occurs and $\mathcal{E}_2$ does not occur, the behaviour of $\mathcal{B}$. In this situation, since VRF is pseudorandom, thus, by the definition of $\mathcal{B}$, $\mathcal{B}$ outputs a random bit with overwhelm probability.

Therefore, the advantage of the adversary $\mathcal{B}$ for experiment $\mathrm{Exp}_{\mathrm{VRF}}$ is:

$$\begin{aligned}\Pr[\mathrm{Adv}_{\mathrm{VRF}}(\mathcal{B})] =& \frac{1}{2}\cdot\Pr[\mathcal{E}_1 \text{ not occurs}] + \frac{1}{2}\cdot\Pr[\mathcal{E}_1 \text{ occurs, and } \mathcal{E}_2 \text{ not occurs}]\\ &+ \frac{1}{2}\cdot\Pr[\mathcal{E}_2 \text{ occurs}] + \frac{1}{4}\cdot\Pr[\mathcal{E}_2 \text{ occurs}]\\ &= \frac{1}{2} + \frac{1}{4}\cdot\Pr[\mathcal{E}_2 \text{ occurs}].\end{aligned}$$

Thus, $\mathcal{B}$ wins the experiment $\mathrm{Exp}_{\mathrm{VRF}}$ with non-negligible probability. This contradicts the pseudorandomness of the VRF. Therefore, RRS satisfies repudiation unforgeability.

4.5 Anonymity

We will now turn to establishing anonymity of RRS.

Lemma 3. *The repudiable ring signature scheme* RRS *satisfies anonymity against adversarially chosen keys, given that* NIWI *has perfect soundness and witness indistinguishablity,* VRF *has commpletenss, uniqueness, and pseudorandomness, and* SPB *has index hiding.*

Our strategy is to first move the index of hk_1 from j_0 to j_1 and argue indistinguishability via the index-hiding property of SPB. Next we switch y_1^0, y_1^1 to an evaluation of $\mathrm{VRF.Eval}(pk_{j_1}^0, (h_1, m, \varphi))$ and $\mathrm{VRF.Eval}(pk_{j_1}^1, (h_1, m, \varphi))$, where h_1 is a digest of R for new hk_1. This modification will not be detected due to the pseudorandom property of VRF. Now, we can switch the NIWI witness to $(\mathrm{VK}_{j_1}, \mathrm{ind}_1, \eta_1, \tau_1^0, \tau_1^1, 1)$, and by witness indistinguishability of NIWI, this signature also satisfies indistinguishability. Next, we perform the first two changes above for hk_0 and y_0^0, y_0^1, switch the witness back to the witness for $j = 0$, and finally replace y_1^0, y_1^1 with a random string. The signature in the last experiment is now a real signature of m under VK_{j_1}.

Proof. Let $\mathcal{A}$ be a PPT adversary against the anonymity of RRS. Assume that $\mathcal{A}$ makes at most $q = \mathrm{poly}(\lambda)$ queries for any oracle. Let in the following ind_0 be the index of VK_{j_0} in R, and ind_1 be the index of VK_{j_1} in R, where $(m, \mathrm{R}, j_0, j_1)$ is the challenge query of $\mathcal{A}$. Now, consider the following hybrids:

Hybrid 1: This is the real experiment with challenge bit $b = 0$.
Hybrid 2: Same as Hybrid 1, except that in σ, we compute hk_1 by $(\mathrm{hk}_1, \mathrm{shk}_1) \leftarrow \mathrm{SPB.Gen}(1^\lambda, |\mathrm{R}|, \mathrm{ind}_1)$.

Hybrid 1 and Hybrid 2 are computationally indistinguishable, given that SPB is index hiding. More specifically, there exists a reduction $\mathcal{R}_1$ such that

$$\mathrm{Adv}_{\mathrm{IH}}(\mathcal{R}_1^{\mathcal{A}}) = \mathrm{Adv}_{\mathrm{H}_1,\mathrm{H}_2}(\mathcal{A}).$$

We will provide an informal description of $\mathcal{R}_1$. The $\mathcal{R}_1$ simulates H_1 faithfully, until $\mathcal{A}$ outputs a challenger query $(m, \mathrm{R}, j_0, j_1)$. Then $\mathcal{R}_1$ gives $(|\mathrm{R}|, j_0, j_1)$ to

the index hiding experiment and receives a hashing key hk^*. $\mathcal{R}_1$ continues the simulation of H_1 faithfully, except that in the challenge signature it sets $\mathrm{hk}_1 = \mathrm{hk}^*$. In the end, $\mathcal{R}_1$ outputs the output of $\mathcal{A}$.

Clearly, if the challenge bit of the index hiding experiment is 0 then $\mathcal{R}_1$ simulates Hybrid 1 perfectly. And if the challenge bit of the index hiding experiment is 1 then $\mathcal{R}_1$ simulates Hybrid 2 perfectly. Therefore, Hybrid 1 and Hybrid 2 are computationally indistinguishable.

Hybrid 3: Same as Hybrid 2, except that we compute y_1^0 by $y_1^0 = \mathrm{VRF.Eval}(sk_{j_1}^0, (h_1, m, \varphi))$.

Hybrid 2 and Hybrid 3 are computationally indistinguishable, given that the VRF is pseudorandom. More specifically, there exists a reduction $\mathcal{R}_2$ such that

$$\mathrm{Adv}_{\mathrm{VRF}}(\mathcal{R}_2^{\mathcal{A}}) = \mathrm{Adv}_{\mathrm{H}_2,\mathrm{H}_3}(\mathcal{A}) \cdot \mathrm{poly}(\lambda).$$

The proof is very similar with the proof of lemma 2. We only provide an informal description of $\mathcal{R}_2$. The reduction $\mathcal{R}_2$ receives as input a public key pk. The $\mathcal{R}_2$ simulates H_2 faithfully, except for the following. Before the simulation stars, $\mathcal{R}_2$ chooses a random index i^* and sets $\mathrm{VK}_{i^*} = (pk, pk_{i^*}^1)$, where $pk_{i^*}^1$ is generated as in H_2 and pk is the input of $\mathcal{R}_2$. $\mathcal{R}_2$ continues the simulation of H_2 [6] until $\mathcal{A}$ announces $(j_0, j_1, m, \mathrm{R})$. If it holds $j_1 \neq i^*$, $\mathcal{R}_2$ outputs $\perp$. Otherwise, $\mathcal{R}_2$ continues the simulation of H_2 faithfully, except that in the challenge signature it sets $y_1^0 = y$, where y is the outputs of challenger when $\mathcal{B}$ gives it (h_1, m, φ). Finally, $\mathcal{R}_2$ continues the simulation and outputs whatever $\mathcal{A}$ outputs.

Hybrid 4: Same as Hybrid 3, except that we compute y_1^1 by $y_1^1 = \mathrm{VRF.Eval}(sk_{j_1}^1, (h_1, m, \varphi))$.

Hybrid 3 and Hybrid 4 are computationally indistinguishable, given that the VRF is pseudorandom. More specifically, there exists a reduction $\mathcal{R}_3$ such that

$$\mathrm{Adv}_{\mathrm{VRF}}(\mathcal{R}_3^{\mathcal{A}}) = \mathrm{Adv}_{\mathrm{H}_3,\mathrm{H}_4}(\mathcal{A}) \cdot \mathrm{poly}(\lambda).$$

The proof is very similar to the above, except that we set $\mathrm{VK}_{i^*} = (pk_{i^*}^0, pk)$, where $pk_{i^*}^0$ is generated as in H_3 and pk is the input of $\mathcal{R}_3$. And since we require $\mathcal{A}$ can not query $\mathrm{OR}(\cdot)$ with $(\cdot, m, \mathrm{R}, \cdot)$, after he has received a challenge signature, $\mathcal{A}$ can not get any advantage from repudiation.

Hybrid 5: Same as Hybrid 4, except that we compute witness by

- $\eta \leftarrow \mathrm{SPB.Open}(\mathrm{hk}_1, \mathrm{shk}_1, \mathrm{R}, \mathrm{ind}_1)$,
- $\tau^0 \leftarrow \mathrm{VRF.Prove}(sk_{j_1}^0, (h_1, m, \varphi))$, $\tau^1 \leftarrow \mathrm{VRF.Prove}(sk_{j_1}^1, (h_1, m, \varphi))$,

and use the witness $w = (\mathrm{VK}_{j_1}, \mathrm{ind}_1, \eta, \tau^0, \tau^1, 1)$ to compute π.

[6] When $\mathcal{A}$ queries its oracle, the answer of $\mathcal{R}_2$ is same as $\mathcal{B}$'s answer in Lemma 2.

Hybrid 4 and Hybrid 5 are computationally indistinguishable, give that NIWI is computationally witness indistinguishable. More specifically, there exists a reduction $\mathcal{R}_4$ against the witness indistinguishability of NIWI such that

$$\mathrm{Adv}_{\mathrm{WI}}(\mathcal{R}_4^{\mathcal{A}}) = \mathrm{Adv}_{\mathrm{H}_4,\mathrm{H}_5}(\mathcal{A}).$$

The reduction $\mathcal{R}_4$ simulates H_4 faithfully, until the challenge signature is computed. Instead of computing the proof π itself, $\mathcal{R}_4$ sends the statement $x = (m, \varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, h_0, h_1)$ and the witness $w_0 \leftarrow (\mathrm{VK}_{j_0}, \mathrm{ind}_0, \eta_0, \tau_0^0, \tau_0^1, 0)$ and $w_1 \leftarrow (\mathrm{VK}_{j_1}, \mathrm{ind}_1, \eta_1, \tau_1^0, \tau_1^1, 1)$ to the witness indistinguishability experiment. The experiment returns a proof π^*, and $\mathcal{R}_4$ use the proof π^* in the challenge signature. $\mathcal{R}_4$ continues the simulation of H_4 faithfully and outputs whatever the simulated $\mathcal{A}$ outputs.

Clearly, if the challenge bit of the witness indistinguishability experiment is 0, then $\mathcal{R}_4$ simulates H_4 perfectly. On the other hand, if the challenge bit is 1, then $\mathcal{R}_4$ simulates H_5 perfectly. Therefore, Hybrid 4 and Hybrid 5 are computationally indistinguishable.

Next, we perform the similar changes as above.

Hybrid 6: Same as Hybrid 5, except that we compute y_0^0 by $y_0^0 \leftarrow \{0,1\}^{\alpha(\lambda)}$.
Hybrid 7: Same as Hybrid 6, except that we compute y_0^1 by $y_0^1 \leftarrow \{0,1\}^{\alpha(\lambda)}$.
Hybrid 8: Same as Hybrid 7, except that in σ, we compute hk_0 by $(\mathrm{hk}_0, \mathrm{shk}_0) \leftarrow \mathrm{SPB.Gen}(1^\lambda, |\mathrm{R}|, \mathrm{ind}_1)$.
Hybrid 9: Same as Hybrid 8, except that we compute y_0^0 by $y_0^0 = \mathrm{VRF.Eval}(sk_{j_1}^0, (h_0, m, \varphi))$ instead of $y_0^0 \leftarrow \{0,1\}^{\alpha(\lambda)}$.
Hybrid 10: Same as Hybrid 9, except that we compute y_0^1 by $y_0^1 = \mathrm{VRF.Eval}(sk_{j_1}^1, (h_0, m, \varphi))$ instead of $y_0^1 \leftarrow \{0,1\}^{\alpha(\lambda)}$.
Hybrid 11: Same as Hybrid 10, except that we compute witness by
- $\eta \leftarrow \mathrm{SPB.Open}(\mathrm{hk}_0, \mathrm{shk}_0, \mathrm{R}, \mathrm{ind}_1)$,
- $\tau^0 \leftarrow \mathrm{VRF.Prove}(sk_{j_1}^0, (h_0, m, \varphi))$, $\tau^1 \leftarrow \mathrm{VRF.Prove}(sk_{j_1}^1, (h_0, m, \varphi))$,

and use the witness $w = (\mathrm{VK}_{j_1}, \mathrm{ind}_1, \eta, \tau^0, \tau^1, 0)$ to compute π.
Hybrid 12: The same as hybrid 11, except that we compute y_1^0 and y_1^1 by $y_1^0, y_1^1 \leftarrow \{0,1\}^{\alpha(\lambda)}$. This is identical to the real experiment with $b = 1$.

These hybrids are also indistinguishability, the proof is analogous, and we omit it. Therefore, RRS satisfies anonymity.

4.6 Unforgeability

We will now turn to showing that our scheme is unforgeable.

Lemma 4. *The repudiable ring signature scheme* RRS *is unforgeable, given that* NIWI *has perfect soundness,* VRF *has completeness, uniqueness, and pseudorandomness, and* SPB *has somewhere perfectly binding.*

The main idea of the proof is that, if adversary can produce a valid signature, then since the NIWI proof has perfect soundness, it must has VK, $i \in [N]$, η,

τ^0, τ^1, and $j \in \{0,1\}$ such that it holds SPB.Verify$(\mathrm{hk}_j, h_j, i, \mathrm{VK}, \eta) = 1$ and VRF.Verify$(pk^0, (h_j, m, \varphi), y_j^0, \tau^0) = 1$, and VRF.Verify$(pk^1, (h_j, m, \varphi), y_j^1, \tau^1) = 1$. Since SPB has somewhere perfectly binding, we have $\mathrm{VK} = \mathrm{R}[i]$. And by the completeness and uniqueness of the VRF, we have $y_j^0 = \mathrm{VRF.Eval}(pk^0, (h_j, m, \varphi))$, and $y_j^1 = \mathrm{VRF.Eval}(pk^1, (h_j, m, \varphi))$, and by this we can attack the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Proof. Suppose, for contradiction, that RRS is not unforgeable. Then there exist some $l = \mathrm{poly}(\lambda)$, and some PPT adversary $\mathcal{A}$ such that $\mathcal{A}$ has non-negligible advantage in experiment $\mathrm{Exp}_{\mathrm{Unf}}$. Next, we will use adversary $\mathcal{A}$ to build an adversary $\mathcal{B}$ that break experiment $\mathrm{Exp}_{\mathrm{VRF}}$:

Adversary $\mathcal{B}$: given input 1^λ, pk, and access to oracle Eval$(sk, \cdot)$, Prove$(sk, \cdot)$.

1. The adversary $\mathcal{B}$ runs RRS.Gen(1^λ), and generates l pairs of key.
2. The adversary $\mathcal{B}$ chooses a random index $i^* \in [l]$, and let $\mathrm{VK}_{i^*} = (pk, pk_{i^*}^1)$ instead of $\mathrm{VK}_{i^*} = (pk_{i^*}^0, pk_{i^*}^1)$. Then $\mathcal{B}$ give 1^λ and $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$ to $\mathcal{A}$.
3. When $\mathcal{A}$ queries its corruption oracle on $i \in [l]$, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ outputs SK_i;
   - If $i = i^*$, then $\mathcal{B}$ outputs a random string x to challenger, and then outputs a random bit $b' \leftarrow \{0,1\}$ and aborts.
4. When $\mathcal{A}$ queries its Signing oracle on $i \in [l]$, m, and R, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ runs the honest signing algorithm RRS.Sign and outputs RRS.Sign$(\mathrm{SK}_i, m, \mathrm{R})$;
   - If $i = i^*$, then $\mathcal{B}$ runs the honest signing algorihm RRS.Sign with the following modification : in step 8, instead of using sk to generate y_0^0 and τ^0, $\mathcal{B}$ invokes its VRF oracle.
5. When $\mathcal{A}$ queries its Repudiation oracle on $j \in [l]$, m, R, and σ, $\mathcal{B}$ runs the honest repudiation algorithm RRS.Repudiate on its input.[7]
6. The adversary $\mathcal{A}$ outputs (m, R, σ). The adversary $\mathcal{B}$ runs the honest verifying algorithm RRS.Verify on (m,R,σ). If RRS.Verify$(m, \mathrm{R}, \sigma) = 0$, then $\mathcal{B}$ outputs a random string x to challenger, and then outputs a random bit $b' \leftarrow \{0,1\}$ and aborts.
7. The adversary $\mathcal{B}$ parse $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, \pi)$, and do :
   - Compute $h_0 = \mathrm{SPB.Hash}(\mathrm{hk}_0, \mathrm{R})$, $h_1 = \mathrm{SPB.Hash}(\mathrm{hk}_1, \mathrm{R})$;
   - Choose a random bit $k \leftarrow \{0,1\}$.

   Then $\mathcal{B}$ submits (h_k, m, φ) to the VRF challenger and then receive responses y'. If $y' = y_k^0$, $\mathcal{B}$ outputs 0. Otherwise, $\mathcal{B}$ outputs a random bit.

It remains to show that the adversary $\mathcal{B}$ has non-negligible advantage, in experiment $\mathrm{Exp}_{\mathrm{VRF}}$. Let us first consider the probability that $\mathcal{A}$ queries oracle OS for input i^*. Recall that this event causes $\mathcal{B}$ to abort and outputs a random bit. The condition $\mathrm{R} \subset \{\mathrm{VK}_1, \ldots, \mathrm{VK}_l\} \setminus \mathcal{Q}_{\mathrm{OC}}$ in the experiment $\mathrm{Exp}_{\mathrm{Unf}}$ ensures

[7] Since sk is not used by RRS.Repudiate, $\mathcal{B}$ does not need to invoke the VRF oracle here.

that if $\mathcal{A}$ wins the game with non-negligible probability, then $\mathcal{A}$ leaves one or more keys uncorrupted with at least non-negligible probability. Since i^* is chosen at random by $\mathcal{B}$, it follows that $\Pr[i^* \notin \mathcal{Q}_{\mathrm{OC}}]$ is non-negligible. Let $\mathcal{E}_1$ denote the event that $\mathcal{A}$ does not corrupt i^*. When $\mathcal{E}_1$ does not occur, then by the definition of $\mathcal{B}$, it will outputs a random bit, and in this time, $\mathcal{B}$ will have exactly $\frac{1}{2}$ advantage in the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Condition on $\mathcal{E}_1$ occurs, since $\mathcal{A}$ does not query corruption oracle on i^*, the view of $\mathcal{A}$ is identical to the view in experiment $\mathrm{Exp}_{\mathrm{Unf}}$. It follows that $\mathcal{A}$ will win the game with non-negligible probability by assumption. Let $\mathcal{E}_2$ denote the event that event $\mathcal{E}_1$ occurs and adversary $\mathcal{A}$ win the game. When $\mathcal{E}_1$ occurs and $\mathcal{E}_2$ does not occur, by the definition of $\mathcal{B}$, it will outputs a random bit, and in this time, $\mathcal{B}$ will have exactly $\frac{1}{2}$ advantage in the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Condition on $\mathcal{E}_2$ occurs, by the perfect soundness of NIWI, there exists VK, i, η, τ^0, and τ^1, $j \in \{0,1\}$ such that it holds $\mathrm{SPB.Verify}(\mathrm{hk}_j, h_j, i, \mathrm{VK}, \eta) = 1$, $\mathrm{VRF.Verify}(pk^0, (h_j, m, \varphi), y_j^0, \tau^0) = 1$, $\mathrm{VRF.Verify}(pk^1, (h_j, m, \varphi), y_j^1, \tau^1) = 1$. Since SPB has somewhere perfectly binding, we have $\mathrm{VK} = \mathrm{R}[i]$. And by the completeness and uniqueness of the VRF, we have $y_j^0 = \mathrm{VRF.Eval}(pk^0, (h_j, m, \varphi))$, and $y_j^1 = \mathrm{VRF.Eval}(pk^1, (h_j, m, \varphi))$. When $i = i^*$, and $j = k$, this moreover implies $y_k^0 = \mathrm{VRF.Eval}(pk, (h_k, m, \varphi))$. In this case, if the VRF challenger's bit $b = 0$, then we have $y' = y_k^0$. Recall that this is the trigger condition for $\mathcal{B}$ to output 0. If the VRF challenger's bit $b = 1$, then y' is turly random strings. Thus, by the definition of $\mathcal{B}$, $\mathcal{B}$ outputs a random bit with overwhelm probability. [8]

Let $\mathcal{E}_3$ denote the event that event $\mathcal{E}_2$ occurs and $i = i^*$, $j = k$. Now we consider the probability that $\mathcal{E}_3$ occurs. As also observed above, the distribution of the view of $\mathcal{A}$ is unaffected by $\mathcal{B}$'s choice of i^*, until the point at which $\mathcal{A}$ submits an oracle query to oracle OC for input i^*. Since i^* is chosen at random by $\mathcal{B}$, and $i^*, i \in \mathrm{R}$, then in this situation we have $\Pr[i^* = i]$ must be non-negligible. Condition on $i = i^*$ occurs, since k is chosen at random by $\mathcal{B}$, and $j, k \in \{0,1\}$, we have $\Pr[j = k] = \frac{1}{2}$. Therefore, according to the above discussion, we have that $\mathcal{E}_3$ occurs with non-negligible probability.

Finally, we consider when $\mathcal{E}_2$ occurs and $\mathcal{E}_3$ does not occur, the behaviour of $\mathcal{B}$. In this situation, since VRF is pseudorandom, thus, by the definition of $\mathcal{B}$, $\mathcal{B}$ outputs a random bit with overwhelm probability.

Therefore, the advantage of the adversary $\mathcal{B}$ for experiment $\mathrm{Exp}_{\mathrm{VRF}}$ is:

$$\begin{aligned}\Pr[\mathrm{Adv}_{\mathrm{VRF}}(\mathcal{B})] =& \frac{1}{2} \cdot \Pr[\mathcal{E}_1 \text{ not occurs}] + \frac{1}{2} \cdot \Pr[\mathcal{E}_1 \text{ occurs, and } \mathcal{E}_2 \text{ not occurs}] \\ &+ \frac{1}{2} \cdot \Pr[\mathcal{E}_2 \text{ occurs, and } \mathcal{E}_3 \text{ not occurs}] \\ &+ \frac{1}{2} \cdot \Pr[\mathcal{E}_3 \text{ occurs}] + \frac{1}{4} \cdot \Pr[\mathcal{E}_3 \text{ occurs}] \\ &= \frac{1}{2} + \frac{1}{4} \cdot \Pr[\mathcal{E}_3 \text{ occurs}].\end{aligned}$$

[8] Besides, in this case, the probability that $\mathcal{B}$ queries its oracle on (h_k, m, φ) is negligible.

Thus, $\mathcal{B}$ wins the experiment $\mathrm{Exp}_{\mathrm{VRF}}$ with non-negligible probability. This contradicts the pseudorandomness of the VRF. Therefore, RRS satisfies unforgeability.

4.7 Repudiability

We will now turn to showing that our scheme is repudiable.

Lemma 5. *The repudiable ring signature scheme* RRS *is repudiable, given that* NIWI *has perfect completeness, perfect soundness,* VRF *has completeness, uniqueness, pseudorandomness, and* SPB *has somewhere perfectly binding and index hiding.*

The main idea of the proof is that, by the definition of repudiability, we need to proof non-signer can repudiate and signer cannot repudiate separately. So we consider these two situations in turn. The proof of non-signer can repudiate is very similar with the proof of Lemma 4, if there exist a PPT adversary $\mathcal{A}$ break the experiment $\mathrm{Exp}_{\mathrm{Rep1}}$, we will use the adversary to build an adversary which can break experiment $\mathrm{Exp}_{\mathrm{VRF}}$. The proof of signer cannot repudiate is also by the same way.

Proof. Suppose, for contradiction, that RRS is not repudiability. Then, it must be that there exist some $l = \mathrm{poly}(\lambda)$, and some PPT adversary $\mathcal{A}$ such that $\mathcal{A}$ has non-negligible advantage in either experiment $\mathrm{Exp}_{\mathrm{Rep1}}$ or experiment $\mathrm{Exp}_{\mathrm{Rep2}}$. Let us consider the two cases separately.

Case 1: There exist some $l = \mathrm{poly}(\lambda)$, and some PPT adversary $\mathcal{A}$ such that $\mathcal{A}$ has non-negligible advantage in experiment $\mathrm{Exp}_{\mathrm{Rep1}}$. Now, we will use adversary $\mathcal{A}$ to build an adversary $\mathcal{B}$ that break the experiment $\mathrm{Exp}_{\mathrm{VRF}}$:

Adversary $\mathcal{B}$: given input 1^λ, pk, and access to oracle $\mathrm{Eval}(sk, \cdot)$, $\mathrm{Prove}(sk, \cdot)$.

1. The adversary $\mathcal{B}$ runs $\mathrm{RRS.Gen}(1^\lambda)$, and generates l pairs of keys.
2. The adversary $\mathcal{B}$ choose a random index $i^* \in [l]$, and let $\mathrm{VK}_{i^*} = (pk^0_{i^*}, pk)$, instead of $\mathrm{VK}_{i^*} = (pk^0_{i^*}, pk^1_{i^*})$. Then $\mathcal{B}$ gives 1^λ and $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$ to $\mathcal{A}$.
3. When $\mathcal{A}$ queries its corruption oracle $\mathrm{OC}(\cdot)$ on $i \in [l]$, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ outputs SK_i;
   - If $i = i^*$, then $\mathcal{B}$ outputs a random bits x to challenger, and then outputs a random bit $b' \leftarrow \{0, 1\}$ and aborts.
4. When $\mathcal{A}$ queries its signing oracle $\mathrm{OS}(\cdot)$ on $i \in [l]$, m, and R, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ runs the honest signing algorithm RRS.Sign and outputs $\mathrm{RRS.Sign}(\mathrm{SK}_i, m, \mathrm{R})$;
   - If $i = i^*$, then $\mathcal{B}$ runs the honest signing algorithm RRS.Sign with the following modification: in step 9, instead of using sk to generate y^1_0 and τ^1, $\mathcal{B}$ generates these by invoking its VRF oracle.

5. When $\mathcal{A}$ queries it repudiation oracle $\mathrm{OR}(\cdot)$ on $i \in [l]$, m, R, and σ, $\mathcal{B}$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}$ runs the honest repudiating algorithm RRS.Repudiate and outputs $\mathrm{RRS.Repudiate}(\mathrm{SK}_i, m, \mathrm{R}, \sigma)$;
   - If $i = i^*$, then $\mathcal{B}$ runs the honest repudiating algorithm RRS.Repudiate with the following modification: in step 5, step 7, step 8, and step 9, instead of using sk to generate y'_0, y'_1, τ_{00}, τ_{10}, z_0, z_1, and τ_{01}, τ_{11}, $\mathcal{B}$ generates these by invoking its VRF oracle.
6. The adversary $\mathcal{A}$ outputs (m, R, σ).
7. The adversary $\mathcal{B}$ parse $\sigma = (\varphi, y^0_0, y^1_0, y^0_1, y^1_1, \mathrm{hk}_0, \mathrm{hk}_1, \pi)$, and do:
   - Compute $h_0 = \mathrm{SPB.Hash}(\mathrm{hk}_0, \mathrm{R})$, $h_1 = \mathrm{SPB.Hash}(\mathrm{hk}_1, \mathrm{R})$;
   - Choose a random bit $k \leftarrow \{0, 1\}$.
8. Finally, $\mathcal{B}$ submits (h_k, m, φ) to the VRF challenger and then receive responses y. If $y = y^1_k$ $\mathcal{B}$ outputs 0. Otherwise, $\mathcal{B}$ outputs a random bit.

It remains to show that the adversary $\mathcal{B}$ has non-negligible advantage in experiment $\mathrm{Exp}_{\mathrm{VRF}}$. Let us first consider the probability that $\mathcal{A}$ queries oracle $\mathrm{OS}(\cdot)$ for input i^*. Recall that this event causes $\mathcal{B}$ to abort and outputs a random bit. The distribution (i.e., verification keys and oracle responses) of the view of $\mathcal{A}$ is unaffected by $\mathcal{B}$'s choice of i^*, until the point at which $\mathcal{A}$ submits an oracle query to oracle OC for input i^*. The condition $\mathrm{R} \setminus \mathcal{Q}_{\mathrm{OC}} \neq \emptyset$ in the experiment $\mathrm{Exp}_{\mathrm{Rep1}}$ ensures that if $\mathcal{A}$ wins the game with non-negligible probability, then $\mathcal{A}$ leaves one or more keys uncorrupted with at least non-negligible probability. Since i^* is chosen at random by $\mathcal{B}$, it follows that $\Pr[i^* \notin \mathcal{Q}_{\mathrm{OC}}]$ is non-negligible. Let $\mathcal{E}_1$ denote the event that $\mathcal{A}$ does not corrupt i^*. When $\mathcal{E}_1$ does not occur, then by the definition of $\mathcal{B}$, it will output a random bit, and in this time, $\mathcal{B}$ will have exactly $\frac{1}{2}$ advantage in the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Condition on $\mathcal{E}_1$ occurs, since $\mathcal{A}$ does not query corruption oracle on i^*, the view of $\mathcal{A}$ is identical to the view in experiment $\mathrm{Exp}_{\mathrm{Rep1}}$. It follows that $\mathcal{A}$ will win the game with non-negligible probability by assumption. Let $\mathcal{E}_2$ denote the event that event $\mathcal{E}_1$ occurs and adversary $\mathcal{A}$ win the game. When $\mathcal{E}_1$ occurs and $\mathcal{E}_2$ does not occur, then since VRF is pseudorandom, by the definition of $\mathcal{B}$, $\mathcal{B}$ will output a random bit with overwhelm probability.

Condition on $\mathcal{E}_2$ occurs, by the definition of experiment $\mathrm{Exp}_{\mathrm{Rep1}}$, we have RRS.VerRepud will reject on an honestly generated repudiation ξ_j, generated with respect to VK_j, i.e. $\exists j \in \mathrm{R} \setminus \mathcal{Q}_{\mathrm{OC}}$, s.t.

$$\mathrm{VerRepud}(\mathrm{VK}_j, m, \mathrm{R}, \sigma, \xi_j) = 0, \text{ where } \xi_j \leftarrow \mathrm{Repudiate}(\mathrm{SK}_j, m, \mathrm{R}, \sigma).$$

Since ξ_j is honestly generated with respect to VK_j, and by the definition of RRS.Repudiate and RRS.VerRepud, we have either $y'_0 = y^1_0$ or $y'_1 = y^1_1$, where $y'_0 = \mathrm{VRF.Eval}(sk^1_j, (h_0, m, \varphi))$ and $y'_1 = \mathrm{VRF.Eval}(sk^1_j, (h_1, m, \varphi))$. Since if the above two formulas are not true, then by the perfect completeness of the NIWI and the completeness of the VRF, we have $\mathrm{VerRepud}(\mathrm{VK}_j, m, \mathrm{R}, \sigma, \xi_j) = 1$.

Now, let us consider the probability that $j = i^*$ condition on $\mathcal{E}_2$ occurs. As also observed above, the distribution of the view of $\mathcal{A}$ is unaffected by $\mathcal{B}$'s choice of i^*, until the point at which $\mathcal{A}$ submits an oracle query to oracle OC for input

i^*. Since i^* is chosen at random by $\mathcal{B}$, and $i^*, j \in \mathrm{R}$, then $\Pr[i^* = j]$ must be non-negligible. Let $\mathcal{E}_3$ denote the event that $\mathcal{E}_2$ occurs and $j = i^*$. Then when $\mathcal{E}_2$ occurs and $\mathcal{E}_3$ does not occur, since VRF is pseudorandom, by the definition of $\mathcal{B}$, $\mathcal{B}$ will output a random bit with overwhelm probability.

Finally, let us consider the situation when $\mathcal{E}_3$ occurs. In this situation, we have either $y'_0 = y^1_0$ or $y'_1 = y^1_1$, where $y'_0 = \mathrm{VRF.Eval}(sk, (h_0, m, \varphi))$ and $y'_1 = \mathrm{VRF.Eval}(sk, (h_1, m, \varphi))$. Without lose generation, we can assume only the first equation is true, then when $\mathcal{B}$ chooses $k = 0$, and VRF challenger's bit $b = 0$, we have $y = y^1_k$. Recall that this is the trigger condition for $\mathcal{B}$ to output 0. Otherwise, by the pseudorandomness of VRF, we have $\mathcal{B}$ will output a random bit with overwhelm probability.

We now consider the the probability that $\mathcal{B}$ queries oracle $\mathrm{Eval}(sk, \cdot)$ or $\mathrm{Prove}(sk, \cdot)$ for input (h_k, m, φ). Since $\mathcal{A}$ is not allow queries its oracle on m and R, and SPB is collision resistant[9], there is only negligible probability such that $\mathcal{B}$ queries oracle $\mathrm{Eval}(sk, \cdot)$ or $\mathrm{Prove}(sk, \cdot)$ for input (h_k, m, φ).

Therefore, the advantage of the adversary $\mathcal{B}$ for experiment $\mathrm{Exp}_{\mathrm{VRF}}$ is:

$$\begin{aligned}\Pr[\mathrm{Adv}_{\mathrm{VRF}}(\mathcal{B})] \geq & \frac{1}{2} \cdot \Pr[\mathcal{E}_1 \text{ not occurs}] + \frac{1}{2} \cdot \Pr[\mathcal{E}_1 \text{ occurs, and } \mathcal{E}_2 \text{ not occurs}] \\ & + \frac{1}{2} \cdot \Pr[\mathcal{E}_2 \text{ occurs, and } \mathcal{E}_3 \text{ not occurs}] \\ & + \frac{1}{4} \cdot \Pr[\mathcal{E}_3 \text{ occurs}] + \frac{3}{8} \cdot \Pr[\mathcal{E}_3 \text{ occurs}] \\ & = \frac{1}{2} + \frac{1}{8} \cdot \Pr[\mathcal{E}_3 \text{ occurs}].\end{aligned}$$

We have shown that $\mathcal{B}$ wins the experiment $\mathrm{Exp}_{\mathrm{VRF}}$ with non-negligible probability. This contradicts the security of the VRF. Therefore, there is no adversary can break experiment $\mathrm{Exp}_{\mathrm{Rep1}}$.

Case 2: There exist some $l = \mathrm{poly}(\lambda)$, and some PPT adversary $\mathcal{A}'$ such that $\mathcal{A}'$ has non-negligible advantage in experiment $\mathrm{Exp}_{\mathrm{Rep2}}$. And now we will use the adversary $\mathcal{A}'$ to build an adversary $\mathcal{B}'$ that break the experiment $\mathrm{Exp}_{\mathrm{VRF}}$:

Adversary $\mathcal{B}'$: given input 1^λ, pk, and access to oracle $\mathrm{Eval}(sk, \cdot)$, $\mathrm{Prove}(sk, \cdot)$.

1. The adversary $\mathcal{B}'$ runs $\mathrm{RRS.Gen}(1^\lambda)$, and generates l pairs of keys.
2. The adversary $\mathcal{B}'$ choose a random index $i^* \in [l]$, and let $\mathrm{VK}_{i^*} = (pk, pk^1_{i^*})$, instead of $\mathrm{VK}_{i^*} = (pk^0_{i^*}, pk^1_{i^*})$. Then $\mathcal{B}'$ gives 1^λ and $\mathrm{VK}_1, \ldots, \mathrm{VK}_l$ to $\mathcal{A}'$.
3. When $\mathcal{A}'$ queries its corruption oracle $\mathrm{OC}(\cdot)$ on $i \in [l]$, $\mathcal{B}'$ answers this query in the following way:
   - If $i \neq i^*$, then $\mathcal{B}'$ outputs SK_i;
   - If $i = i^*$, then $\mathcal{B}'$ outputs a random bits x to challenger, and then outputs a random bit $b' \leftarrow \{0, 1\}$ and aborts.
4. When $\mathcal{A}'$ queries its Signing oracle on $i \in [l]$, m, and R, $\mathcal{B}'$ answers this query in the following way:

[9] Because SPB is index hiding

- If $i \neq i^*$, then $\mathcal{B}'$ runs the honest signing algorithm RRS.Sign and outputs RRS.Sign($\mathrm{SK}_i, m, \mathrm{R}$);
- If $i = i^*$, then $\mathcal{B}'$ runs the honest signing algorihm RRS.Sign with the following modification : in step 8, instead of using sk to generate y_0^0 and τ^0, $\mathcal{B}'$ invokes its VRF oracle.

5. When $\mathcal{A}'$ queries its Repudiation oracle on $j \in [l]$, m, R, and σ, $\mathcal{B}'$ runs the honest repudiation algorithm RRS.Repudiate on its input.[10]
6. The adversary $\mathcal{A}'$ outputs $(m, \mathrm{R}, \sigma, \{\xi_{i_k}\}_{\mathrm{VK}_{i_k} \in \mathcal{Q}_{\mathrm{OC}} \cap \mathrm{R}})$. The adversary $\mathcal{B}'$ runs the honest verifying algorithm RRS.Verify on (m,R, σ). If RRS.Verify(m, $\mathrm{R}, \sigma) = 0$, then $\mathcal{B}'$ outputs a random string x to challenger, and then outputs a random bit $b' \leftarrow \{0,1\}$ and aborts.
7. The adversary $\mathcal{B}'$ runs the honest verrepud algorithm RRS.VerRepud, $b_{i_k} =$ RRS.VerRepud($\mathrm{VK}_{i_k}, m, \mathrm{R}, \sigma, \xi_{i_k}$) for all $\mathrm{VK}_{i_k} \in \mathcal{Q}_{\mathrm{OC}} \cap \mathrm{R}$. If there exists a $b_{i_k} = 0$, then $\mathcal{B}'$ outputs a random string x to challenger, and then outputs a random bit $b' \leftarrow \{0,1\}$ and aborts.
8. The adversary $\mathcal{B}'$ parse $\sigma = (\varphi, y_0^0, y_0^1, y_1^0, y_1^1, \mathrm{hk}_0, \mathrm{hk}_1, \pi)$, and do :
   - Compute $h_0 = \mathrm{SPB.Hash}(\mathrm{hk}_0, \mathrm{R})$, $h_1 = \mathrm{SPB.Hash}(\mathrm{hk}_1, \mathrm{R})$;
   - Choose a random bit $k \leftarrow \{0,1\}$.

   Then $\mathcal{B}'$ submits (h_k, m, φ) to the VRF challenger and then receive responses y'. If $y' = y_k^0$, $\mathcal{B}'$ outputs 0. Otherwise, $\mathcal{B}'$ outputs a random bit.

It remains to show that the adversary $\mathcal{B}'$ has non-negligible advantage, in experiment $\mathrm{Exp}_{\mathrm{VRF}}$. Let us first consider the probability that $\mathcal{A}'$ queries oracle OS for input i^*. Recall that this event causes $\mathcal{B}'$ to abort and outputs a random bit. If the adversary $\mathcal{A}'$ queries all keys, then we have $\mathrm{R} \subset \mathcal{Q}_{OC}$, in this situation we can proof $\mathcal{A}'$ cannot win the game $\mathrm{Exp}_{\mathrm{Rep2}}$. Since in this case, if $\mathcal{A}'$ wins the game, then σ will be a valid signature for m, R. By perfect soundness of NIWI, it must has VK, $i \in [N]$, η, τ^0, τ^1, and $j \in \{0,1\}$ such that it holds SPB.Verify($\mathrm{hk}_j, h_j, i, \mathrm{VK}, \eta$) $= 1$ and VRF.Verify($pk^0, (h_j, m, \varphi), y_j^0, \tau^0$) $= 1$, and VRF.Verify($pk^1, (h_j, m, \varphi), y_j^1, \tau^1$) $= 1$. By somewhere perfectly binding of SPB, we have $\mathrm{VK} = \mathrm{R}[i]$. And by the completeness and uniqueness of the VRF, we have $y_j^0 =$ VRF.Eval($pk^0, (h_j, m, \varphi)$), and $y_j^1 =$ VRF.Eval($pk^1, (h_j, m, \varphi)$). Now, let us consider the repudiation of VK, if ξ is the a valid repudiation, then by the perfect soundness of NIWI proof, it must has y_0', y_1' τ_{00}, τ_{01}, τ_{10}, τ_{11}, such that it holds VRF.Verify($pk^1, (h_0, m, \varphi), y_0', \tau_{00}$) $= 1$, and VRF.Verify($pk^1, y_0', z_0, \tau_{01}$) $= 1$, VRF.Verify($pk^1, (h_1, m, \varphi), y_1', \tau_{10}$) $= 1$, and VRF.Verify($pk^1, y_1', z_1, \tau_{11}$) $= 1$. Since VRF has completeness and uniqueness, we have $y_0' =$ VRF.Eval($pk^1, (h_0, m, \varphi)$), and $z_0 =$ VRF.Eval(pk^1, y_0'), $y_1' =$ VRF.Eval($pk^1, (h_1, m, \varphi)$), and $z_1 =$ VRF.Eval(pk^1, y_1'). Therefore, by the uniqueness of VRF, we have $(y_0' = y_0^1) \vee (y_1' = y_1^1) = 1$ with overwhelm probability. This is contradictory. Since we assume that $\mathcal{A}$ wins the game with non-negligible probability, then $\mathcal{A}$ leaves one or more keys uncorrupted in R with at least non-negligible probability. Since i^* is chosen at random by $\mathcal{B}$, it follows that $\Pr[i^* \notin \mathcal{Q}_{OC}]$ is non-negligible. Let $\mathcal{E}_1$ denote the event that $\mathcal{A}$ does not corrupt i^*. When $\mathcal{E}_1$ does not occur, then by the

[10] Since sk is not used by RRS.Repudiate, $\mathcal{B}'$ does not need to invoke the VRF oracle here.

definition of $\mathcal{B}$, it will outputs a random bit, and in this time, $\mathcal{B}$ will have exactly $\frac{1}{2}$ advantage in the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Condition on $\mathcal{E}_1$ occurs, since $\mathcal{A}$ does not query corruption oracle on i^*, the view of $\mathcal{A}$ is identical to the view in experiment $\mathrm{Exp}_{\mathrm{Rep2}}$. It follows that $\mathcal{A}$ will win the game with non-negligible probability by assumption. Let $\mathcal{E}_2$ denote the event that event $\mathcal{E}_1$ occurs and adversary $\mathcal{A}$ win the game. When $\mathcal{E}_1$ occurs and $\mathcal{E}_2$ does not occur, by the definition of $\mathcal{B}$, it will outputs a random bit, and in this time, $\mathcal{B}$ will have exactly $\frac{1}{2}$ advantage in the experiment $\mathrm{Exp}_{\mathrm{VRF}}$.

Condition on $\mathcal{E}_2$ occurs, by the perfect soundness of NIWI, somewhere perfectly binding of SPB and completeness and uniqueness of VRF, we have there exist VK, i, and $j \in \{0,1\}$ such that $\mathrm{VK} = \mathrm{R}[i]$, $y_j^0 = \mathrm{VRF.Eval}(pk^0, (h_j, m, \varphi))$, and $y_j^1 = \mathrm{VRF.Eval}(pk^1, (h_j, m, \varphi))$. And according to the above discussion, we have $\mathrm{VK} \notin \mathcal{Q}_{OC}$. And furthermore, when $i = i^*$, and $j = k$, this moreover implies $y_k^0 = \mathrm{VRF.Eval}(pk, (h_k, m, \varphi))$.

In this case, if the VRF challenger's bit $b = 0$, then we have $y' = y_k^0$. Recall that this is the trigger condition for $\mathcal{B}$ to output 0. If the VRF challenger's bit $b = 1$, then y' is turly random strings. Thus, by the definition of $\mathcal{B}$, $\mathcal{B}$ outputs a random bit with overwhelm probability. [11]

Let $\mathcal{E}_3$ denote the event that event $\mathcal{E}_2$ occurs and $i = i^*$, $j = k$. Now we consider the probability that $\mathcal{E}_3$ occurs. As also observed above, the distribution of the view of $\mathcal{A}$ is unaffected by $\mathcal{B}$'s choice of i^*, until the point at which $\mathcal{A}$ submits an oracle query to oracle OC for input i^*. Since i^* is chosen at random by $\mathcal{B}$, and $i^*, i \in \mathrm{R}$, then in this situation we have $\Pr[i^* = i]$ must be non-negligible. Condition on $i = i^*$ occurs, since k is chosen at random by $\mathcal{B}$, and $j, k \in \{0,1\}$, we have $\Pr[j = k] = \frac{1}{2}$. Therefore, according to the above discussion, we have that $\mathcal{E}_3$ occurs with non-negligible probability.

Finally, we consider when $\mathcal{E}_2$ occurs and $\mathcal{E}_3$ does not occur, the behaviour of $\mathcal{B}$. In this situation, since VRF is pseudorandom, thus, by the definition of $\mathcal{B}$, $\mathcal{B}$ outputs a random bit with overwhelm probability.

Therefore, the advantage of the adversary $\mathcal{B}$ for experiment $\mathrm{Exp}_{\mathrm{VRF}}$ is:

$$\begin{aligned}\Pr[\mathrm{Adv}_{\mathrm{VRF}}(\mathcal{B})] =& \frac{1}{2} \cdot \Pr[\mathcal{E}_1 \text{ not occurs}] + \frac{1}{2} \cdot \Pr[\mathcal{E}_1 \text{ occurs, and } \mathcal{E}_2 \text{ not occurs}] \\ &+ \frac{1}{2} \cdot \Pr[\mathcal{E}_2 \text{ occurs, and } \mathcal{E}_3 \text{ not occurs}] \\ &+ \frac{1}{2} \cdot \Pr[\mathcal{E}_3 \text{ occurs}] + \frac{1}{4} \cdot \Pr[\mathcal{E}_3 \text{ occurs}] \\ &= \frac{1}{2} + \frac{1}{4} \cdot \Pr[\mathcal{E}_3 \text{ occurs}].\end{aligned}$$

Thus, $\mathcal{B}$ wins the experiment $\mathrm{Exp}_{\mathrm{VRF}}$ with non-negligible probability. This contradicts the security of the VRF. Therefore, there is no adversary can break experiment $\mathrm{Exp}_{\mathrm{Rep2}}$. This concludes the proof.

[11] Besides, in this case, the probability that $\mathcal{B}$ queries its oracle on (h_k, m, φ) is negligible.

5 Conclusions

Ring signatures are a well-studied cryptographic primitive with many applications. Repudiable ring signatures are a more stronger cryptographic primitive than ring signatures, which can allow non-signer repudiate a signature that was not produced by him. In this paper we improved the state-of-the-art by introducing a scheme with signature and repudiation size that is logarithmic in the number of ring members, while at the same time relying on standard assumptions and not requiring a trusted setup.

References

1. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short accountable ring signatures based on DDH. In: European Symposium on Research in Computer Security. Springer, Cham, 2015: 243-265.
2. Backes, M., Döttling, N., Hanzlik, L., Kluczniak, K., Schneider, J.: Ring Signatures: Logarithmic-Size, No Setupfrom Standard Assumptions. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Cham. 2019: 281-311.
3. Barak, B., Ong, S.J., Vadhan, S.: Derandomization in cryptography. In: SIAM Journal on Computing, 2007, 37(2): 380-400.
4. Bitansky, N., Paneth, O.: ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In: Theory of Cryptography Conference. Springer, Berlin, Heidelberg, 2015: 401-427.
5. Dwork, C., Naor, M.: Zaps and their applications. In: Proceedings 41st Annual Symposium on Foundations of Computer Science. IEEE, 2000: 283-293.
6. Dodis, Y.: Efficient construction of (distributed) verifiable random functions. In: International Workshop on Public Key Cryptography. Springer, Berlin, Heidelberg, 2003: 1-17.
7. Dodis, Y., Kiayias, A., Nicolosi, A., Shoup, V.: Anonymous identification in ad hoc groups. In: International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg. 2004: 609-626.
8. Dodis, Y., Yampolskiy, A.: A verifiable random function with short proofs and keys. In: Public Key Cryptography. (2005) 416C431.
9. Fujisaki, E., Suzuki, K.: Traceable ring signature. In: International Workshop on Public Key Cryptography. Springer, Berlin, Heidelberg, 2007: 181-200.
10. Groth, J., Ostrovsky, R., Sahai, A.: New techniques for noninteractive zero-knowledge. In: Journal of the ACM, 2012, 59(3): 11.
11. Hubacek, P., Wichs, D.: On the communication complexity of secure function evaluation with long output. In: Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science. ACM, 2015: 163-172.
12. Libert, B., Peters, T., Qian, C.: Logarithmic-size ring signatures with tight security from the DDH assumption. In: European Symposium on Research in Computer Security. Springer, Cham. 2018: 288-308.
13. Liu, J.K, Wei, V.K, Wong, D.S.: Linkable spontaneous anonymous group signature for ad hoc groups. In: Australasian Conference on Information Security and Privacy. Springer, Berlin, Heidelberg, 2004: 325-335.

14. Lysyanskaya, A.: Unique signatures and verifiable random functions from the DH-DDH separation. In: Annual International Cryptology Conference. Springer, Berlin, Heidelberg, 2002: 597-612.
15. Micali, S., Rabin, M., Vadhan, S.: Verifiable random functions. In: 40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039). IEEE, 1999: 120-130.
16. Noether S.: Ring SIgnature Confidential Transactions for Monero. In: IACR Cryptology ePrint Archive. 2015, 1098.
17. Okamoto, T., Pietrzak, K., Waters, B., Wichs, D.: New realizations of somewhere statistically binding hashing and positional accumulators. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2015: 121-145.
18. Park S., Sealfon A.: It Wasn't Me! Repudiability and unclaimability of ring signature. In: Annual International Cryptology Conference. Springer, Cham, 2019: 159-190.
19. Rivest R.L., Shamir A., Tauman Y.: How to leak a secret. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2001: 552-565.
20. Xu, S., Yung, M.: Accountable ring signatures: a smart card approach. In: Smart Card Research and Advanced Applications VI. Springer, Boston, MA, 2004: 271-286.

A Attack on [18]

In [18], they claim their repudiable ring signature scheme satisfies adaptive anonymity against adversarially chosen keys they proposed. But we find it is wrong, their construction can only satisfies anonymity (Definition 6) we proposed. Here, we will give an attack on the anonymity of their construction.

Let R-RS be the repudiable ring signature proposed in [18], and let $\mathrm{OC}(\cdot)$ be the corrpution oracle, $\mathrm{OS}(\cdot)$ be the signing oracle, and $\mathrm{OR}(\cdot)$ be the repudiation oracle.

Adversary $\mathcal{A}$: given 1^λ, $\mathrm{VK}_1, \cdots, \mathrm{VK}_l$, $\mathrm{OC}(\cdot)$, $\mathrm{OS}(\cdot)$, and $\mathrm{OR}(\cdot)$.

1. The adversary $\mathcal{A}$ chooses $(m, \mathrm{R}, j_0, j_1)$ at random, and gives it to experiment.
2. The challenger gives $\mathcal{A}$ a signature σ, $\mathcal{A}$ parse $\sigma = ((\pi_1, \cdots \pi_n), (y_1, \cdots y_4), \varphi)$, and then $\mathcal{A}$ chooses $y' \leftarrow \{0,1\}^l$, and let $\sigma' = ((\pi_1, \cdots \pi_n), (y_1, \cdots y_3, y'), \varphi)$.
3. The adversary $\mathcal{A}$ gives the input $j_0, m, \mathrm{R}, \sigma'$ to its repudiation oracle $\mathrm{OR}(\cdot)$, and get repudiation $\xi = (\xi_1, \cdots, \xi_n)$.
4. The adversary $\mathcal{A}$ runs the honest verification algorithm R-RS.VerRepud and outputs $\text{R-RS}(\mathrm{R}, \mathrm{VK}_{j_0}, \sigma', \xi)$ to challenger.

We can find that if challenger's bit $b = 0$, then by the definition of R-RS.Repudiate, in this situation ξ cannot pass through the verification. When $b = 1$, since R-RS.Repudiate only use y_1, y_2, that change does not affect the generation of reputation, and ξ can pass through the verification. Therefore, $\mathcal{A}$ has non-negligible advantage in the experiment.