# It wasn't me!

## Repudiability and Unclaimability of Ring Signatures

Sunoo Park
MIT Media Lab

Adam Sealfon
MIT CSAIL

**Abstract**

Ring signatures, introduced by [RST01], are a variant of digital signatures which certify that *one among a particular set* of parties has endorsed a message while hiding *which* party in the set was the signer. Ring signatures are designed to allow *anyone* to attach anyone else's name to a signature, as long as the signer's own name is also attached.

But what guarantee do ring signatures provide if a purported signatory wishes to denounce a signed message — or alternatively, if a signatory wishes to later come forward and claim ownership of a signature? Prior security definitions for ring signatures do not give a conclusive answer to this question: under most existing definitions, the guarantees could go either way. That is, it is consistent with some standard definitions that a non-signer might be able to *repudiate* a signature that he did not produce, or that this might be impossible. Similarly, a signer might be able to later convincingly claim that a signature he produced is indeed his own, or not. Any of these guarantees might be desirable. For instance, a whistleblower might have reason to want to later claim an anonymously released signature, or a person falsely implicated in a crime associated with a ring signature might wish to denounce the signature that is framing them and damaging their reputation. In other circumstances, it might be desirable that even under duress, a member of a ring cannot produce proof that he did or did not sign a particular signature. In any case, a *guarantee* one way or the other seems highly desirable.

In this work, we formalize definitions and give constructions of the new notions of *repudiable*, *unrepudiable*, *claimable*, and *unclaimable* ring signatures. Our repudiable construction is based on VRFs, which are implied by several number-theoretic assumptions (including strong RSA or bilinear maps); our claimable construction is a black-box transformation from any standard ring signature scheme to a claimable one; and our unclaimable construction is derived from the lattice-based ring signatures of [BK10], which rely on hardness of SIS. Our repudiable construction also provides a new construction of standard ring signatures.

## 1 Introduction

Ring signatures, introduced by [RST01], are a variant of digital signatures which certify that *one among a particular set* of parties has signed a particular message, without revealing which specific party is the signer. This set is called a "ring." Ring signatures can be useful, for example, to certify that certain leaked information comes from a privileged set of government or company 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,[1] or to enable untraceable transactions in

[1] When it comes to national security issues, for instance, there may be some reluctance among law-makers to "roll back" existing laws or reduce checking or surveillance measures, which could be due in part to the risk of ending up a scapegoat upon any future national security incident like a terrorist attack.

cryptocurrencies (as in Monero [Mon]).

In a ring signature scheme, just as in a traditional digital signature scheme, any party can create a key pair for signing and verification, and publish the verification key. Signers can produce signatures that verify with respect to any set of verification keys that includes their own, and unforgeability guarantees that no party can produce a valid signature with respect to a set of verification keys without possessing a corresponding secret key.

But what guarantee does a ring signature scheme provide if a purported signatory wishes to denounce a signed message — or alternatively, if a signatory wishes to later come forward and *claim* ownership of a signature? Given the motivation of anonymity behind the notion of a ring signature, a natural first intuition might be that parties should be able neither to denounce nor to claim a signature in a convincing way. However, depending on the threat model, we believe that the opposite guarantees — that is, to guarantee the ability to denounce or claim signatures — may be useful too, as elaborated below. Furthermore, whatever one's preference, a guarantee one way or the other seems more desirable than no guarantee either way.

Prior security definitions for ring signatures do not conclusively provide these guarantees one way or the other. That is, a non-signer might be able to *repudiate* a signature that he did not produce ("repudiability"), or this might be impossible ("unrepudiability"). Similarly, a signer might be able to later convincingly claim that a signature he produced is indeed his own ("claimability"), or be unable to do so ("unclaimability").

The most detailed taxonomy of security definitions for ring signatures was given by [BKM09], which presents a series of anonymity guarantees of increasing strength. A natural anonymity guarantee defined by [BKM09], called "anonymity against adversarially chosen keys," is informally described as follows: an adversary who controls all but $t \geq 2$ parties in a ring, and who may produce his own malformed key pairs as well as corrupt honest parties' keys, must have negligible advantage at guessing which of the $t$ honest parties produced a given signature. This anonymity definition might allow a party to ascertain whether a given signature was produced by her own signing key, and perhaps also to convince others of this fact — but it does not *guarantee* or *prohibit* either of these capabilities.

On the other hand, the strongest of the anonymity definitions of [BKM09] (called "anonymity against full key exposure") requires that even if an adversary compromises every single party in a ring, the adversary cannot identify the signers of past signatures. It is relatively straightforward to see that under such a strong anonymity guarantee, Alice would have no way to convince anyone that she did not produce the objectionable message; indeed, she herself cannot tell the difference between a signature produced using her own signing key and one produced using someone else's.

The ability to identify whether one's own signing key was used to produce a particular signature can be a feature or a bug. To protect anonymity of past signatures against a very strong adversary who might compromise all the secret keys in a ring, it seems desirable to prevent distinguishing one's own signatures from those generated by someone else. On the other hand, without the ability to distinguish, it would be virtually impossible to tell if someone had stolen your signing key. Moreover, as discussed below, it could be beneficial in certain circumstances for members of a ring to have the ability to disown signatures of messages that they have strong reasons to denounce; and conversely, in some circumstances the signer of a message might later wish to prove to the world that he was the one who produced a particular signature in the past.

We have now identified four potentially useful notions for ring signatures: *repudiability*, *unrepudiability*, *claimability*, and *unclaimability*. The main contributions in this paper consist both of

new definitions and constructions of each of these notions. Before diving into an overview of definition and constructions, we provide some discussion of why each of these notions — some of which directly oppose each other — may be meaningful and desirable: the following scenarios explore a few of the circumstances in which various of the above guarantees might be appropriate. Though some of the scenarios are phrased somewhat whimsically, we believe that each scenario illustrates a meaningful threat model motivating the definition concerned.

**Scenario 1 (Repudiability)** Let us consider a hypothetical tale, 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. As mentioned above, ring signatures are deliberately designed to allow *anyone* to attach anyone else's name to a signature, without the latter's knowledge or consent. Despite this, there could be realistic situations in which non-signing members of a ring associated with a particular message could suffer serious consequences through no fault of their own, perhaps due to the real signer adversarially trying to damage their reputation. In light of this, perhaps it would be desirable in some contexts for the owner of a verification key to be able to denounce messages, e.g., to clear her name of a crime or hate speech accusation that might otherwise impact her life in terms of reputation, job prospects, or incarceration.

**Scenario 2 (Claimability)** Our next story concerns a talented brewery employee who developed new statistical techniques to test the quality of beers. Naturally, his employer was protective of its competitive advantage since other breweries at the time may not have been using similar statistical methods. Yet, in the interest of science, they allowed him to publish his results — on condition of anonymity.[2] A credible way to prove authorship at a later date, after the need for anonymity has ceased to exist, might be very useful — especially in case of competing claims by impostors. As we see here, claiming authorship of an anonymous work may become appropriate after a passage of time. The next example illustrates quite a different type of situation in which claimability at the signer's discretion may be valuable.

Consider an employee Emily who is concerned about unethical practices at her company, and takes it upon herself to expose what is going on and publish a critical commentary. Concerned about her job security and possible retribution, as well as the credibility of her allegations, she maintains her anonymity using ring signatures. It emerges, in fact, that similar practices are prevalent across the industry: related revelations drive a wider movement of reform. Some time later, after her company has substantially reformed its practices and her fears of retribution have been allayed

[2]This is the true story of William Sealy Gosset's invention of the Student's $t$-test at Guinness Brewery in 1908 [Man00].

— perhaps by her promotion, or by a change in leadership — Emily seeks to reveal her identity and add her voice to the growing movement, providing her solidarity, legitimation, and follow-up story. In addition, if following the reforms, those involved in the earlier unethical practices were subject to stigma or even prosecution, claimability of her earlier ring signatures would allow Emily to exculpate herself.

**Scenario 3 (Unrepudiability and unclaimability)** Let us return to the government of the fictional country of Oz. The parliament of Oz is mired in partisan gridlock, with legislators from each party ruthlessly voting down any bills, however reasonable, proposed by members of the opposing party — effectively preventing any laws from being enacted and perhaps shutting down the government, which is in no party's interest. Suppose that instead of directly proposing a new law, a legislator of Oz anonymously publishes the text of the proposed bill using a ring signature scheme:

*Proposed: that free ice cream shall be provided every Tuesday.*[3]
*Signed: a member of the Parliament of Oz.*

If the signer used an unclaimable ring signature scheme, then she could not decide to reveal her identity upon a later change of heart, allowing legislators of both parties to support or oppose the bill on its merits without worrying about purely political considerations.

Unclaimability and unrepudiability may be particularly useful guarantees in scenarios where the placement of whole groups of people under duress is a substantial concern. For instance, in circumstances where an employer or authoritarian government may coercively compel individuals to provide a repudiation or proof of authorship (e.g. signing randomness) for a signature, the provable inability to do so convincingly may be essential. Unrepudiability may also be desirable in situations in which members of a ring are likely to have conflicting individual incentives but there is a possibility of collective benefit in case of cooperation, as in a prisoner's dilemma scenario.

**Summary of technical contributions.** We formalize *repudiability*, *unrepudiability*, *claimability*, and *unclaimability* of ring signatures, as well as strengthened anonymity and unforgeability definitions which are compatible with each of these notions. We show that *unclaimability* implies *unrepudiability* (intuitively, because a failed repudiation can be used as a claim). Anonymity against adversarially chosen keys is the strongest anonymity notion compatible with *repudiability* and *claimability*, and anonymity against full key exposure is implied by *unclaimability* and equivalent to *unrepudiability*.

We provide three constructions based on different assumptions, one for each of the three notions of *repudiability*, *claimability*, and *unclaimability*. Perhaps the most surprising of these is *unclaimability*, which guarantees that the signer cannot later credibly convince others that she produced a particular signature. A natural first intuition is that meaningful notions of unclaimability might be impossible to achieve, since a signer can always remember the signing randomness (and later present it as "proof" of having produced a signature). The key insight for our definition and construction of unclaimable ring signatures is that the signing randomness does not constitute a convincing claim if *anyone in the ring can also produce credible signing randomness* for any signature in which they are implicated. Our construction of *unclaimable* ring signatures is an augmentation of the

[3]Even if each party might support this legislation, they may be unwilling to do so if it were proposed by the other party, decrying their respective opponents as either fiscally irresponsible or in the pocket of Big Ice Cream.

lattice-based ring signature scheme of [BK10] that adds additional algorithms allowing anyone in the ring to generate credible signing randomness; this capability is achieved via lattice trapdoors.

Our construction of *repudiable* ring signatures is based on verifiable random functions (VRFs), which are implied by either the (strong) RSA assumption, assumptions on bilinear maps, or NIWIs and commitments; see [Bit17; GHKW17] and references therein for more detailed discussion of the assumptions that imply VRFs.[4] Our construction does not use standard ring signatures as a building block, and as such can also be viewed as a new construction of standard ring signatures. Our construction of *claimable* ring signatures, on the other hand, is a simple and generic black-box transformation from any standard ring signature scheme to a claimable one. We overview our contributions in more detail below.

### 1.1 Definitional contributions

**Repudiability.** We define a *repudiable ring signature scheme* as a ring signature scheme that is equipped with additional algorithms $\mathsf{Repudiate}$ and $\mathsf{VerRepud}$ as follows. $\mathsf{Repudiate}$ takes as input a signing key $sk$, a ring signature $\sigma$, and a "ring" $R$ (i.e., a set of verification keys), and outputs a *repudiation* $\xi$. $\mathsf{VerRepud}$ takes as input a ring $R$, a signature $\sigma$, a repudiation $\xi$, and a verification key $vk$, and outputs a a single bit indicating whether or not $\xi$ is a valid repudiation attesting that $\sigma$ was not produced by $vk$. The two requirements for a ring signature scheme to be *repudiable* are, informally, as follows.

1. *Correctness:* Any member of a ring must be able to produce valid repudiations of any signature that he did not produce.
2. *Soundness:* A cheating signer must not be able to produce a valid signature with respect to a ring, and also be able to produce valid repudiations of that signature under every verification key in that ring that he owns.

Once a ring signature scheme is equipped with these additional repudiation algorithms, the standard definitions of *unforgeability* and *anonymity* against adversarially chosen keys are insufficient to capture the natural guarantees that would be desired for a repudiable ring signature scheme: we need the release of repudiations not to compromise the unforgeability or anonymity of any future signatures. Accordingly, we modify the definitions of unforgeability and anonymity for repudiable ring signatures (Definitions 3.4 and 3.5), by additionally giving the adversary access to a *repudiation oracle*. This ensures that repudiations of past signatures do not affect the security guarantees of future signatures. See Section 3.1 for formal definitions of repudiability.

**Claimability.** We define a *claimable ring signature scheme* as a ring signature scheme equipped with additional algorithms $\mathsf{Claim}$ and $\mathsf{VerClaim}$ as follows. $\mathsf{Claim}$ takes as input a signing key $sk$, a signature $\sigma$, and a ring $R$, and outputs a claim $\zeta$. $\mathsf{VerClaim}$ takes a input a ring $R$, a verification key $vk$, a signature $\sigma$, and a claim $\zeta$, and outputs a single bit indicating whether or not $\zeta$ is a valid claim attesting that $\sigma$ was produced by $vk$. The three requirements for a claimable ring signature scheme are, informally, as follows.

1. *Correctness:* Any honest signer must be able to produce a valid claim with respect to any signature that he produced.

[4] Note that the ring signature construction of [BKM09] assumes the closely related assumption of NIZKs, which are implied by VRFs and also imply VRFs if the NIZK is subexponentially hard under a standard derandomization assumption; see [Bit17] for details.

2. *Soundness:* No adversary can produce a valid claim with respect to a signature produced by an honest signer, even if the adversary can choose the message and ring with respect to which the signature is produced, and can insert malformed verification keys into the ring.
3. *No framing:* No adversary can produce a signature together with a valid claim of that signature on behalf of an honest (non-signing) party.

As above, once a ring signature scheme is equipped with these additional claiming algorithms, the standard definitions of *unforgeability* and *anonymity* against adversarially chosen keys are insufficient. We modify the definitions of anonymity and unforgeability for claimable ring signatures (Definitions 3.10 and 3.11), by additionally giving the adversary access to a *claim oracle*. See Section 3.3 for formal definitions of repudiability.

*Repudiability* and *claimability* are compatible, i.e., a ring signature scheme can be both repudiable and claimable. Indeed, our repudiable and claimable constructions together give rise to such a scheme. Notably, the unforgeability and anonymity definitions corresponding to the natural notion of a repudiable-and-claimable ring signature scheme are *not* the conjunction of unforgeability and anonymity for repudiable ring signatures and for claimable ring signatures. Rather, the unforgeability and anonymity definitions for a repudiable-and-claimable ring signature scheme involve a stronger adversary which is simultaneously given access to both a repudiation oracle and a claim oracle. See Section 3.5 for further discussion on repudiable-and-claimable schemes.

**Unclaimability** We also introduce *unclaimable* ring signature schemes, in which the signer *provably cannot* convincingly claim that she was the one who produced the signature. As briefly mentioned above, while the signer can always save the signing randomness and reveal it along with her secret key in an attempt to claim authorship of a signature, it is not always true that this constitutes a convincing claim. In particular, such a claim is not credible if *any* member of the ring can take a valid signature and produce fake randomness that produces the desired signature using her own signing key.

The idea that a non-signer can adaptive produce fake randomness is reminiscent of deniable encryption [CDNO97], in which an encryptor and/or recipient is required to produce fake randomness "explaining" that a particular ciphertext is an encryption of an adversarially chosen message. [5]

We define an *unclaimable* ring signature scheme to capture just this requirement: that is, any member of the ring must be able to produce fake signing randomness for a signature that is distributed indistinguishably from real signing randomness. Intuitively, the only information potentially possessed by a signer but not by the other members of the ring is the signing randomness, so non-signers that can generate convincing simulated signing randomness can also convincingly simulate any additional information that might be released by the signer in an attempt to claim the signature. We consider a strong flavor of this definition in which the indistinguishability property, described informally below, is statistical.

1. *Indistinguishability:* Any member of a ring must be able to produce fake signing randomness given a signature. The signature and fake signing randomness must be distributed statistically close to an honestly generated signature and corresponding signing randomness used by that individual to sign the same message, even given all verification keys and signing keys.

We formally define unclaimability in Section 3.4.

[5] This is intuitively somewhat similar to the goal in deniable encryption [CDNO97], in which the encryptor can produce fake randomness to explain a particular ciphertext as an encryption of an arbitrary message.

| | |
|---|---|
| Repudiable | VRF (Section 4) |
| Unrepudiable | RS anonymous against FKE (Section 3.2) |
| Claimable | Transformation from any RS (Section 5) |
| Unclaimable | SIS (Section 6) |

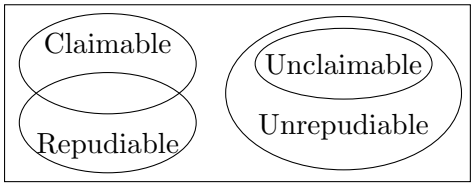


Figure 1: Summary of our results and assumptions relied on. VRF = verifiable random function, RS = ring signature, FKE = full key exposure, SIS = short integer solution problem.

*Remark* 1. Even under this definition, if the signer chooses a message to sign that corresponds to a secret known only to herself, then she may still be able to convince others that she was the signer. For instance, if the signed message is the output of a one-way function, she may be able to convince others that she was the signer by subsequently revealing the preimage. Even more flagrantly, the signed message could contain a signature using a standard (non-ring) signature scheme, directly identifying the signer. This property is rather inherent: if knowledge of the contents of the message itself at the time of signing are enough to identify the signer, then no security property on the signature scheme can enforce that the signer remains hidden, since the identification of the signer is unrelated to the signature and based only on the signed message.

Indeed, ring signatures were not designed to provide anonymity for signers who *want* to identify themselves, but rather for those who desire anonymity. Similarly, our unclaimability definition does not guarantee unclaimability for those who *want* to identify themselves, but rather provides credibility for a signer who *wants* to later be able to claim (e.g., under duress) that she could not convincingly claim the signature even if she wanted to. In particular, even an adversary with unlimited computational power who obtains the secret keys belonging to every member of the ring and a purported signing randomness from an alleged signer, he still will not be convinced of the identity of the signer, since fake signing randomness from the right distribution can be produced for every member of the ring.

**Unrepudiability** Unclaimability intuitively guarantees that no member of the ring can convincingly prove that she was the signer. A related, weaker notion that might be desirable in some circumstances is that of *unrepudiability*, which guarantees that no member of the ring can convincingly prove that she was *not* the signer. Unrepudiability is equivalent to anonymity against full key exposure and is implied by unclaimability.

## 1.2 Overview of our constructions

**Our repudiable construction** Our construction relies on ZAPs (two-round public-coin witness-indistinguishable proofs) and verifiable random functions (VRFs) as building blocks.[6] Our building blocks have some overlap with those of the ring signature construction of [BKM09], which uses ZAPs, public-key encryption (PKE), and a digital signature scheme. Both our scheme and theirs use ZAPs to achieve anonymity of the ring signatures, but with different approaches: the statements proven by the ZAPs are quite unrelated in the two constructions. Moreover, in our scheme, we do

[6] VRFs imply ZAPs, so it suffices to assume VRFs. [GO92; DN07]

not need PKE or signature schemes, and instead use VRFs directly to achieve unforgeability and repudiability. The structure of our construction is thus very different from that of [BKM09].

At a very high level, each signing key in our construction contains a tuple of four VRF keys. A signature consists of the output of each of the signer's VRFs on the message, along with a ZAP proof that (several of) the VRF values in the signature are correct w.r.t. the VRF verification key of some member of the ring. A repudiation for individual $i$ consists of a ZAP proof that some of the VRF values in the signature are different from the correct values for party $i$'s VRFs evaluated at the message. One complication arises because we must guarantee that the release of a repudiation for individual $i$ on a message does not subsequently allow a different member of the ring to produce a signature on the message that cannot be repudiated by individual $i$. We overcome this difficulty by relying on the witness indistinguishability property of the ZAP and ensuring that the repudiation does not reveal the actual VRF outputs of the repudiator; that is, the ZAP proof is produced with the VRF proof as a *witness*. The specific statement proven by the ZAPs is that some specific combination of at least two of the purported VRF outputs is correct. Although in the honest usage of the scheme, all four are produced correctly, we design the specific structure of the statements proved in order to allow a hybrid argument to argue indistinguishability between signatures of different signers in a ring. This scheme of proving the correctness of VRF outputs turns out also to imply unforgeability, not only repudiability, so we do not need to rely on any underlying signature scheme as building block. (In other words, our scheme can also be seen as a new construction of standard ring signatures based on VRFs.)

**Our claimable construction** We give a generic transformation from any standard ring signature scheme $\mathsf{RS}$ to a claimable one. The transformation uses commitment schemes, standard signatures, and PRFs (which are all achievable from one-way functions). The basic idea is to take a signature $\sigma_{\mathsf{RS}}$ under $\mathsf{RS}$ and append to it a *commitment* $c$ to $(vk, \sigma_{\mathsf{RS}})$ where $vk$ is the verification key of the signer. The verification algorithm simply checks whether $\sigma_{\mathsf{RS}}$ verifies. The claim consists of a decommitment revealing that $c$ is a commitment to $(vk, \sigma_{\mathsf{RS}})$. Intuitively, by the hiding property of the commitment scheme, the identity of the signer is hidden until he chooses to publish a claim.

The simple transformation just described runs into a couple of problems when examined in detail. First, what if a signer commits to $(\sigma_{\mathsf{RS}}, vk')$ where $vk'$ is not his own key but that of someone else in the ring? This ability would violate equation (6) of Definition 3.9 (claimability). To prevent such behavior, our construction actually commits to a *standard (non-ring) signature* on $(vk, \sigma_{\mathsf{RS}})$. The unforgeability property of standard signatures then guarantees, intuitively, that a signer cannot convincingly make a claim with respect to any verification key unless he knows a corresponding signing key.

A second issue encountered by the scheme thus far described is that the signer must remember the commitment randomness in order to produce a claim. It is preferable that the signer not be stateful between signing and claiming; indeed, Definition 3.9 requires this. To resolve this, our construction derives commitment randomness from a PRF. For similar reasons, the signing randomness for the standard (non-ring) signature in our construction is also derived from a PRF.

*Remark* 2. Among the constructions presented in this paper, claimability is by far the simplest. Moreover, as a generic transformation, it has the advantage of adding minimal efficiency overhead to the existing state of the art in ring signatures. The simplicity of achieving claimability is perhaps unsurprising in light of the natural intuition that claiming should be possible simply by remembering the signing randomness. As evidenced by *un*claimability, this intuition is not strictly true in general, as in certain schemes, producing signing randomness may not prove authorship.

In a nutshell, our generic transformation ensures that signing randomness is indeed a convincing proof of authorship in the resulting scheme, and moreover builds into the scheme a simple method of efficiently recovering the signing randomness without storing it explicitly.

**Our unclaimable construction** Our construction of unclaimable ring signatures is an extension of the SIS-based ring signature scheme of Brakerski and Kalai [BK10]. The construction is based on trapdoor sampling. In this overview, we describe a simplified version of the scheme. The full scheme is described in Section 6. The basic idea for obtaining unclaimability is that each identity corresponds to a public matrix $A_i \in \mathbb{Z}_q^{n \times m}$ sampled together with a secret trapdoor $T_i$. A signature will consist of short vectors $x_i \in \mathbb{Z}_q^m$ such that

$$\sum_i A_i x_i = y,$$

where $y$ is a target value. For this overview, we can think of $y$ as the output of a random oracle on the message; in the actual construction, $y$ will be obtained as the sum of additional matrix-vector products. In order to sign the message, signer $i$ first samples short vectors $x_j$ for each $j \neq i$. Then, using the lattice trapdoor $T_i$, he samples a short vector $x_i$ such that the equation

$$x_i = y - \sum_{j \neq i} A_j x_j$$

is satisfied. The signature is the list of vectors $\sigma = (x_i)_i$. Using properties of lattice trapdoors, it follows that the distribution over $(x_i)_i$ can be made statistically close no matter which trapdoor was used to produce the signature. Moreover, given a vector $x^*$ to be produced, we can sample random coins that will yield that vector under either the ordinary sampling algorithm or the trapdoor sampling algorithm. Consequently, we obtain an algorithm that can produce explanatory randomness for a signature under any identity in the ring.

Removing the random oracle to obtain ring signatures in the plain model (and unclaimable ones) requires several complications. [BK10] first describes a basic ring signature scheme with weaker unforgeability properties, in which the target vector $y$ is determined using additional matrix-vector products for matrices that depend on the bits of the message. They then amplify the security of the scheme through a sequence of transformations that ultimately yield a scheme with full unforgeability. In Section 6, we first define an algorithm for producing explanatory randomness for their basic scheme, and then describe how to modify this algorithm for each modification of the basic scheme, ultimately yielding an unclaimable ring signature scheme based on the SIS assumption.

*Remark* 3. The idea that a non-signer of a given signature can adaptively produce fake signing randomness is reminiscent of deniable encryption [CDNO97], in which an encryptor of a given ciphertext can adaptively produce fake randomness consistent with it being an encryption of a different message. In this context, it may seem somewhat surprising that our construction relies on a relatively standard assumption (SIS) while many natural definitions of deniable encryption are not known to be achievable without heavier assumptions such as indistinguishability obfuscation [SW14; CPP18]. A subtle difference that is significant here is that a deniably encrypted message must still be recoverable by the honest decryptor, while in the unclaimable ring signature setting, the signer's identity need not be recoverable by anyone.

### 1.3 Other related work

Several constructions of ring signatures based on lattice assumptions have been proposed (e.g., [BK10; Mel+13; BLO18]). The only other construction of ring signatures based on ZAPs is [BKM09], to our knowledge. Numerous other ring signature constructions have been proposed, mostly based on various assumptions on bilinear maps, many but not all of which are in the random oracle model (e.g., [Ngu05; SS10; Boo+15]).

Two additional works in the lattice trapdoor literature bear mentioning: the seminal [Ajt99], and the more recent [MP12]. The latter is more recent than [GPV08], whose trapdoors our unclaimable construction relies on (this reliance is carried over from the [BK10] construction).

**Ring signatures with additional guarantees** Since the original proposal of ring signatures by [RST01], various variant definitions have been proposed. For example, *linkable* ring signatures [LWW04] allow identification of signatures that were produced by the same signer, without compromising the anonymity of the signer within the ring. An enhancement to this notion called *designated linkability* [LSW06] does not allow linkability by default, but instead allows links to be revealed at will by a designated party. Another notion called *traceable* ring signatures [FS07] considers a setting where signatures are generated with respect to "tags" and each member may sign at most a single message (say, a vote) with respect to a particular tag, or else his identity will be revealed. *Accountable* ring signatures [XY04; Boo+15] allow a signer to assign the power to de-anonymize her signature to a specific publicly identified party.

It may seem that some of these variants of ring signature schemes have properties that would be useful for constructing *claimable* ring signatures as introduced in this paper. This implication is unsurprising in the context of our results: *all* of the above types of ring signature schemes in fact imply claimable ring signatures, since our construction of claimable ring signatures is a generic transformation from *any* ring signature scheme. It is unclear if leveraging the additional features of variant schemes would be more desirable than applying our generic transformation, which has very low overhead and moreover can be applied to a simpler, more efficient ring signature scheme that may lack these additional properties.

**Group signatures** Group signatures [CH91] are a different type of signature that allow signing w.r.t. a set of verification keys and provide anonymity of the signer within that set. This concept differs most strikingly from ring signatures in that there is a central authority that (1) sets up the group (i.e., set of signers) and issues keys to members of the group and (2) has the power to revoke the anonymity of the signer of a signature. Notions such as (un)linkability, described above, have been applied to the group signature setting as well. Notably, there has also been proposed a notion of *deniable group signatures* [Ish+16], in which the group manager may issue proofs that a particular group member did *not* sign a particular signature. This bears a little resemblance to our notion of *repudiability* in ring signatures; however, the presence of a central authority in the group signature setting means these problems are technically rather disparate. [LNWX17] construct lattice-based deniable group signatures; however, their technique for deniability is very different from ours, and relies on zero-knowledge proofs of plaintext inequality for LWE ciphertexts, which do not suffice in our setting.

## 2 Anonymity and unforgeability of ring signatures

This section overviews standard ring signature definitions: syntax, correctness, anonymity, and unforgeability. We express the anonymity and unforgeability definitions differently from prior work, as explained in their respective subsections. However, our definitions are equivalent to the correspondingly named definitions from prior work. Throughout the paper, $k$ denotes the security parameter.

**Definition 2.1** (Ring signature)**.** A *ring signature scheme* is a triple of PPT algorithms $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$, satisfying the three properties of *correctness* (Definition 2.2), *anonymity* (Definitions 2.5–2.6), and *unforgeability* (Definition 2.8). The syntax of $\mathsf{Gen}$, $\mathsf{Sign}$, and $\mathsf{Verify}$ follows.

- $\mathsf{Gen}(1^k)$ takes $k$ as input and outputs a *verification key* $vk$ and a *signing key* $sk$.
- $\mathsf{Sign}(R, sk, m)$ takes as input a signing key $sk$, a message $m$, and a set of verification keys $R = \{vk_1, \ldots, vk_N\}$, and outputs a signature $\sigma$. The set $R$ is also known as a "ring."
- $\mathsf{Verify}(R, \sigma, m)$ takes as input a set $R$ of verification keys, a signature $\sigma$, and a message $m$, and outputs a single bit indicating whether or not $\sigma$ is a valid signature on $m$ w.r.t. $R$.

Where it may not be clear from context, we sometimes write $\mathsf{RS.Gen}, \mathsf{RS.Sign}, \mathsf{RS.Verify}$ to denote the $\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}$ algorithms belonging to $\mathsf{RS}$.

**Definition 2.2** (Correctness)**.** A ring signature scheme $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfies *correctness* if there is a negligible function $\varepsilon$ such that for any $N = \mathsf{poly}(k)$, any $N$ key pairs $(vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k)$, any $i \in [N]$, and any message $m$,

$$\Pr\left[\mathsf{Verify}(R, \mathsf{Sign}(R, sk_i, m), m) = 1\right] = 1 - \varepsilon(k) \ , \tag{1}$$

where $R = \{vk_1, \ldots, vk_N\}$. $\mathsf{RS}$ satisfies *perfect correctness* if (1) holds for $\varepsilon = 0$.

*Remark* 4. Definition 2.2 considers only for honestly generated keys. One could also consider a stronger requirement that verification be successful for honestly generated signatures with respect to rings containing adversarial keys. Any scheme satisfying Definition 2.2 can be transformed into one satisfying the stronger definition, by modifying original signature algorithm $\mathsf{Sign}$ to a new algorithm $\mathsf{Sign}'$ that operates as follows. On input $R, sk, m$, for a sufficiently large polynomial $p$:

1. $\sigma \leftarrow \mathsf{Sign}(R, sk, m)$
2. $b_1, \ldots, b_p \leftarrow \mathsf{Verify}(R, \sigma, m)$
3. if $\forall i \in [p], b_i = 1$, output $\sigma$; else, go back to step 1

### 2.1 Anonymity

Prior work, notably [RST01; BKM09], has presented a variety of anonymity definitions for ring signatures. Two of the definitions from prior work are relevant to this paper: anonymity against adversarially chosen keys, and anonymity against full key exposure.

This section presents a new, generalized anonymity definition which is parametrized by *oracle sets*, and expresses the two relevant anonymity definitions as instantiations of the generalized definition. This generalized definition is useful to consolidate the existing definitions and make clear their relationship to one another; it captures not only the two definitions we rely on here, but also others from prior work. Moreover, the generalized definition will be essential to concisely express the new anonymity definitions that we introduce in later sections for anonymity of *repudiable*

and *claimable* ring signature schemes (in Sections 3.1.1 and 3.3.1 respectively). In a nutshell, this is because the new definitions need to allow the adversary access to additional oracles related to repudiation and/or claiming.

The generalized definition follows. It is parametrized by sets of oracles $\mathcal{O}_1, \mathcal{O}_2$ and an additional parameter $\alpha \in \{0, 1, 2\}$ that limits the adversary's corruptions.

**Definition 2.3** ($(\mathcal{O}_1, \mathcal{O}_2, \alpha)$-anonymity)**.** Let $\mathcal{O}_1, \mathcal{O}_2$ be sets of oracles, where each oracle in the set is parametrized by a list of key-pairs. Define $\mathsf{Corr}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$ to take as input $i \in [N]$ and output $\omega_i \leftarrow \mathsf{Gen}^{-1}(vk_i, sk_i)$.[7]

A ring signature scheme $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfies $(\mathcal{O}_1, \mathcal{O}_2, \alpha)$-*anonymity* if for any PPT adversary $\mathcal{A}$ and any polynomial $N = \mathsf{poly}(k)$, $\Pr[b' = b]$ in the above game is negligibly close to $1/2$. That is, formally, $\forall$ PPT $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $N = \mathsf{poly}(k)$, there is a negligible function $\varepsilon$ such that

$$\Pr\left[\begin{array}{l} (vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k) \\ ((m^*, i_0^*, i_1^*, R^*), \mathfrak{s}) \leftarrow \mathcal{A}_1^{\mathcal{O}_1, \mathsf{Corr}}(vk_1, \ldots, vk_N) \\ b \leftarrow \{0,1\} \\ \sigma \leftarrow \mathsf{Sign}(R^* \cup \{vk_{i_0^*}, vk_{i_1^*}\}, sk_{i_b^*}, m^*) \\ b' \leftarrow \mathcal{A}_2^{\mathcal{O}_2, \mathsf{Corr}}(\mathfrak{s}, \sigma) \end{array} : \; b' = b \wedge |\{i_0^*, i_1^*\} \cap I| \leq \alpha \right] < \frac{1}{2} + \varepsilon(k) \;, \tag{2}$$

where $I$ is the set of queries to the corruption oracle; and the notation $\mathcal{A}^{\mathcal{O}, \mathsf{Corr}}$ means that for each oracle $O$ in $\mathcal{O}$, $\mathcal{A}$ has oracle access to $O_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$, and $\mathcal{A}$ also has oracle access to $\mathsf{Corr}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$.

Definitions 2.5 and 2.6 are instantiations of Definition 2.3. They are equivalent to the correspondingly named definitions in [BKM09].

**Definition 2.4** (Signing oracle $\mathsf{OSign}$)**.** For a ring signature scheme $\mathsf{RS}$, the oracle $\mathsf{OSign}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$ is defined to take as input $i \in [n]$, a message $m$, and a set $R$, and output $\mathsf{RS.Sign}(R \cup \{vk_i\}, sk_i, m)$. When the oracle is invoked with respect to a single key pair (i.e., $\mathsf{OSign}_{(vk,sk)}$), we treat the oracle as taking only two inputs, $m$ and $R$, since $i$ is superfluous in this case.

**Definition 2.5** (Anonymity against adversarially chosen keys)**.** A ring signature scheme $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfies *anonymity against adversarially chosen keys* if it is $(\{\mathsf{OSign}\}, \varnothing, 0)$-anonymous. Moreover, $\mathsf{RS}$ satisfies *adaptive anonymity against adversarially chosen keys* if it is $(\{\mathsf{OSign}\}, \{\mathsf{OSign}\}, 0)$-anonymous.

Definition 2.5 captures the guarantee that as long as there are at least two honest parties in a ring (represented by $i_0^*, i_1^*$), even if all other parties in the ring are corrupted by an adversary, the adversary cannot tell which of the honest parties produced a signature. One can also consider an even stronger definition where the adversary may corrupt *all but one* or even *all* of the parties in the ring, as in Definition 2.6.

[7] The function $\mathsf{Gen}^{-1}$ takes as input a verification key $vk$ and signing key $sk$ produced by $\mathsf{Gen}$, and produces the randomness used by $\mathsf{Gen}$ to produce this key pair. That is, it samples from the set $\{\omega : \mathsf{Gen}(1^k; \omega) = (vk, sk)\}$. In practice we will only ever invoke $\mathsf{Gen}^{-1}$ on a key pair produced by $\mathsf{Gen}$, so we could invert efficiently by simply remembering the randomness used by $\mathsf{Gen}$, but for the purposes of this definition we will describe it as a sampling procedure. Upon the first invocation on an input $i$, $\mathsf{Corr}$ samples $\omega_i \leftarrow \mathsf{Gen}^{-1}(vk_i, sk_i)$, stores it, and outputs it. If $\mathsf{Corr}$ is queried twice on the same input $i$ then it outputs the same $\omega_i$ that was previously stored.

**Definition 2.6** (Anonymity against full key exposure)**.** A ring signature scheme $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfies *anonymity against full key exposure* if it is $(\{\mathsf{OSign}\}, \varnothing, 2)$-anonymous.

*Remark* 5. Adaptive variants of anonymity were not given or discussed in prior work.[8] In this paper, we refer primarily to *adaptive anonymity against adversarially chosen keys*: this is the strongest notion compatible with repudiability and claimability. To see this, observe that knowledge of a single one (say, $sk_{i_0^*}$) of the two challenge secret keys is sufficient to violate anonymity in a repudiable or claimable scheme, since the challenge signature $\sigma$ was produced by $sk_{i_0^*}$ if and only if repudiating (resp. claiming) $\sigma$ using $sk_{i_0^*}$ yields an invalid repudiation (resp. valid claim).

Definition 2.6 does not include an adaptive version because adaptivity does not give the adversary any additional power when he can corrupt all the keys.[9]

## 2.2 Unforgeability

The first unforgeability definition that follows is parametrized by an oracle set, taking a similar approach to our anonymity definitions above. In this section, we only give one instantiation of the parametrized definition of unforgeability. We will give other instantiations of Definition 2.7 later in the paper, in defining unforgeability for *repudiable* and *claimable* ring signature schemes (in Sections 3.1.1 and 3.3.1 respectively).

**Definition 2.7** ($\mathcal{O}$-unforgeability)**.** Let $\mathcal{O}$ be a set of oracles, where each oracle in the set is parametrized by a list of key-pairs. A ring signature scheme $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is *$\mathcal{O}$-unforgeable* if for any PPT $\mathcal{A}$ and any $N = \mathsf{poly}(k)$, there is a negligible function $\varepsilon$ such that

$$\Pr\left[\begin{array}{ll} (vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k) & \\ (R^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O},\mathsf{OSign},\mathsf{Corr}}(vk_1, \ldots, vk_N) & : \quad b = 1 \wedge R^* \subseteq \{vk_1, \ldots, vk_N\} \setminus I \\ b \leftarrow \mathsf{Verify}(R^*, \sigma^*, m^*) & \quad \wedge Q \cap \{(\cdot, m^*, R^*)\} = \varnothing \end{array}\right] < \varepsilon(k)\,,$$

where the notation $\mathcal{A}^{\mathcal{O},\mathsf{OSign},\mathsf{Corr}}$ is defined as in Definition 2.3, and $I$ and $Q$ are the sets of queries made to the corruption and signing oracles respectively.

We refer to the event that the conditions on the right-hand side of the colon in the above probability expression are met as a "successful forgery."

**Definition 2.8** (Unforgeability of ring signatures)**.** A ring signature scheme $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is *unforgeable* if it is $\varnothing$-unforgeable.[10]

[8] This notwithstanding, the prior constructions of [RST01; BKM09; BK10] all achieve adaptive anonymity. The constructions of [RST01; BK10] achieve perfect/statistical anonymity, respectively — the former, in the random oracle model — so adaptivity follows. The proof of anonymity of [BKM09] construction essentially suffices to prove adaptive anonymity, though the argument was not made in that paper since there was no adaptive definition of anonymity.

[9] An intermediate notion between Definitions 2.5 (which sets $\alpha = 0$) and 2.6 (which sets $\alpha = 2$) is *anonymity against attribution attacks*, defined in [BKM09], which effectively sets $\alpha = 1$. Adaptive anonymity against attribution attacks is not equivalent to the non-adaptive variant of the same.

[10] This is the definition described as *unforgeability with respect to insider corruption* in [BKM09], and is the strongest of the three unforgeability definitions considered therein.

# 3 New definitions: (un)repudiability and (un)claimability

## 3.1 Repudiable ring signatures

Repudiability addresses the question of whether or not members of a ring can prove that they did *not* sign a particular message (when they in fact did not sign it).

**Definition 3.1** (Repudiable ring signature)**.** A *repudiable ring signature scheme* is a ring signature scheme with an additional pair of algorithms $(\mathsf{Repudiate}, \mathsf{VerRepud})$, satisfying the four properties of *correctness* (Definition 2.2), *repudiability* (Definition 3.3), *anonymity* (Definition 3.4), and *unforgeability* (Definition 3.5). The syntax of $\mathsf{Repudiate}$ and $\mathsf{VerRepud}$ follows.

- $\mathsf{Repudiate}(R, sk, \sigma)$ takes as input a signing key $sk$, a ring signature $\sigma$, and a set of verification keys $R = \{vk_1, \ldots, vk_N\}$, and outputs a repudiation $\xi$.
- $\mathsf{VerRepud}(R, vk, \sigma, \xi)$ takes as input a set $R$ of verification keys, a signature $\sigma$, a repudiation $\xi$, and an identity $vk$, and outputs a single bit indicating whether or not $\xi$ is a valid repudiation of signature $\sigma$ for identity $vk$.

**Definition 3.2** (Repudiation oracle $\mathsf{ORpd}$)**.** For a repudiable ring signature scheme $\mathsf{RS}$, the oracle $\mathsf{ORpd}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$ is defined to take as input $i \in [n]$, a signature $\sigma$, and a set $R$, and output $\mathsf{RS.Repudiate}(R \cup \{vk_i\}, sk_i, \sigma)$. When the oracle is invoked with respect to a single key pair (i.e., $\mathsf{ORpd}_{(vk,sk)}$), we treat the oracle as taking only two inputs, $\sigma$ and $R$, since $i$ is superfluous in this case.

Additionally, we define the oracle $\mathsf{ORpd}^{\langle\sigma^*\rangle}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$ to output $\perp$ when it receives the signature $\sigma^*$ as input, and otherwise to give the same response as $\mathsf{ORpd}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$.

Repudiability requires two conditions, expressed by equations (3) and (4) below. Intuitively, (3) captures the requirement "good people can repudiate," i.e., that for any (possibly maliciously generated) signature, an honest party who did not produce it should be able to successfully repudiate. (4) captures the requirements that "bad people cannot repudiate a signature they produced," i.e., addressing the case where the malicious signature and repudiation are both produced using the key being verified, and thus we want the signer to be unable to produce a valid repudiation.

**Definition 3.3** (Repudiability)**.** A ring signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfies *repudiability* if equipped with algorithms $(\mathsf{Repudiate}, \mathsf{VerRepud})$ such that the following conditions hold.

1. *(Non-signers can repudiate)* Let $\mathcal{O} = \{\mathsf{OSign}\}$. For any (possibly adversarial) PPT signing algorithm $\mathcal{A}_{\mathsf{Sign}}$, there exists a negligible function $\varepsilon$ such that

$$\Pr\left[\begin{array}{l} (vk, sk) \leftarrow \mathsf{Gen}(1^k) \\ (\sigma, m, R') \leftarrow \mathcal{A}^{\mathcal{O},\mathsf{ORpd}_{(vk,sk)}}_{\mathsf{Sign}}(vk) \\ \xi \leftarrow \mathsf{Repudiate}(R', sk, \sigma) \\ b \leftarrow \mathsf{VerRepud}(R', vk, \sigma, \xi) \\ b' \leftarrow \mathsf{Verify}(R', \sigma, m) \end{array} : \begin{array}{l} b = 1 \vee b' = 0 \\ \vee Q \cap \{(\cdot, m, R')\} \neq \varnothing \end{array}\right] > 1 - \varepsilon(k)\ . \qquad (3)$$

2. *(Signer cannot repudiate)* For any (possibly adversarial) sign-and-repudiate algorithm $\mathcal{A}_{\mathsf{S\&R}}$,

there is a negligible function $\varepsilon$ such that for any $N = \mathsf{poly}(k)$,

$$\Pr\left[\begin{array}{l} (vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k) \\ (\sigma, R', m, \{\xi_{vk}\}_{vk \in R' \setminus R}) \leftarrow \mathcal{A}^{\mathcal{O}}_{\mathsf{S\&R}}(R) \\ \forall vk \in R' \setminus R,\ b_{vk} \leftarrow \mathsf{VerRepud}(R', vk, \sigma, \xi_{vk}) \\ b' \leftarrow \mathsf{Verify}(R', \sigma, m) \end{array} : \begin{array}{l} R' \cap R = \varnothing \vee \bigvee\limits_{vk \in R' \setminus R} b_{vk} = 0 \\ \vee b' = 0 \vee Q \cap \{(\cdot, m, R')\} \neq \varnothing \end{array}\right] > 1 - \varepsilon(k)\ , \tag{4}$$

where $R = \{vk_1, \ldots, vk_N\}$, $\mathcal{O} = \{\mathsf{OSign}, \mathsf{ORpd}\}$, and $Q$ is the set of queries to $\mathsf{OSign}$.

*Remark* 6. Equation 4 guarantees that a party possessing a set of signing keys cannot repudiate under all of these keys, *as long as some key in the ring is honestly generated.* If the adversary generates all keys in the ring, then he may be able to produce a repudiation under every key in the ring. However, this does not undermine the purpose of repudiability: indeed, if presented with repudiations under every key in a ring, one can confidently conclude that all keys in the ring were generated dishonestly, and thus that all parties in the ring effectively colluded to produce each signature under that ring. Similarly, given repudiations for a subset of the identities in a ring, one can conclude that *either* one of the remaining identities in the ring produced the signature *or* all of the remaining identities in the ring colluded maliciously to produce the signature. That is, either way, at least one of the remaining identities is responsible for the signature.

### 3.1.1 Anonymity and unforgeability of repudiable ring signatures

The definitions of anonymity and unforgeability need to be adapted for repudiable ring signature schemes, to incorporate a repudiation oracle as described next.

**Definition 3.4** (Anonymity of repudiable ring signatures)**.** A repudiable ring signature scheme

$$(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Repudiate}, \mathsf{VerRepud}))$$

satisfies *anonymity against adversarially chosen keys* if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $(\{\mathsf{OSign}, \mathsf{ORpd}\}, \varnothing, 2)$-anonymous (Definition 2.3) Moreover, the repudiable ring signature satisfies *adaptive* anonymity against adversarially chosen keys if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $(\{\mathsf{OSign}, \mathsf{ORpd}\}, \{\mathsf{OSign}, \mathsf{ORpd}^{\langle\sigma\rangle}\}, 2)$-anonymous, where $\sigma$ is the challenge signature in the anonymity experiment (Equation 2).

Recall from Remark 5 that adaptive anonymity against adversarially chosen keys is the strongest anonymity notion compatible with repudiability.

**Definition 3.5** (Unforgeability of repudiable ring signatures)**.** A repudiable ring signature scheme

$$(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Repudiate}, \mathsf{VerRepud}))$$

is unforgeable if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $\{\mathsf{ORpd}\}$-unforgeable (Definition 2.7).

## 3.2 Unrepudiable ring signatures

We next consider a notion where it is *not* possible for a party to prove to others that he did not produce a particular signature. In fact, though it may not be immediately apparent, a natural formalization of this notion is expressed by the definition of *anonymity against full key exposure*

(Definition 2.6): that is, the strongest of the anonymity definitions given in Section 2. The following paragraphs justify this claim with detailed intuition.

Recall that anonymity against full key exposure (FKE) preserves signer anonymity even against an adversary that obtains all of the secret keys of all members of a ring. A ring signature scheme that satisfies repudiability could not also satisfy anonymity against FKE, because of the following attack: the adversary obtains all secret keys in the ring, attempts to repudiate using each secret key, and identifies as the signer the one secret key with respect to which the repudiation algorithm does not produce a valid repudiation. With overwhelming probability, by definition of repudiability, there is exactly one such secret key.

This informal argument establishes that anonymity against FKE must imply any reasonable notion of unrepudiability. The question then arises: are the two notions equivalent? While there arguably exist meaningful definitions of unrepudiability that are weaker than anonymity against FKE, we believe anonymity against FKE is the most reasonable definition of unrepudiability, as explained next.

Any reasonable definition of unrepudiability should capture the intuitive requirement that non-signers cannot behave distinguishably from signers. A little more precisely, for any protocol that could be executed by a non-signer Nancy with respect to a signature $\sigma$ and her verification key $vk'$, the signer Sigmund of that signature must be able to engage in the same protocol with respect to his own verification key $vk$ and behave indistinguishably from Nancy. In other words, we require that if Nancy's secret key were stolen, the thief would be unable to tell whether $\sigma$ was produced by Nancy or by someone else. Indeed, if Nancy were stateless and did not remember what signatures she had produced in the past, or simply lent her secret key to someone else who used it to produce signatures, then she herself would not be able to tell. The definition of anonymity against FKE embodies almost exactly this requirement — but instead of requiring anonymity against the thief who steals just Nancy's key, the definition makes the stronger requirement that anonymity must hold even against a thief who has every secret key in the ring corresponding to $\sigma$.

Is a weaker definition, which only rules out *unilateral* repudiations by a single party, a meaningful definition of unrepudiability? Perhaps. However, it is more in keeping with the intuitive goals and standard properties of ring signatures to protect against adversaries that may have many or all secret keys in a ring: that is, to rule out even the possibility of multiple ring members *colluding* to produce a repudiation for some ring member. Thus we arrive at the following definition.

**Definition 3.6** (Unrepudiable ring signature scheme)**.** A ring signature scheme satisfies *unrepudiability* if it satisfies anonymity against full key exposure (Definition 2.6).

### 3.3 Claimable ring signatures

Claimability addresses whether the actual signer can prove later that they were the signer, without remembering the signing randomness.

**Definition 3.7** (Claimable ring signature)**.** A *claimable ring signature scheme* is a ring signature scheme with an additional pair of algorithms $(\mathsf{Claim}, \mathsf{VerClaim})$, satisfying the four properties of *correctness* (Definition 2.2), *claimability* (Definition 3.9), *anonymity* (Definition 3.10), and *unforgeability* (Definition 3.11). The syntax of $\mathsf{Claim}$ and $\mathsf{VerClaim}$ follows.

- $\mathsf{Claim}(R, sk, \sigma)$ takes as input a signing key $sk$, a ring signature $\sigma$, and a set of verification keys $R = \{vk_1, \ldots, vk_N\}$, and outputs a claim $\zeta$.

- $\mathsf{VerClaim}(R, vk, \sigma, \zeta)$ takes as input a set $R$ of verification keys, a signature $\sigma$, a claim $\zeta$, and an identity $vk$, and outputs a single bit indicating whether or not $\zeta$ is a valid claim of signature $\sigma$ for identity $vk$.

**Definition 3.8** (Claim oracle $\mathsf{OClaim}$)**.** For a claimable ring signature scheme $\mathsf{RS}$, the oracle $\mathsf{OClaim}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$ is defined to take as input $i \in [n]$, a set $R$, and a signature $\sigma$, and output $\mathsf{RS.Claim}(R, sk, \sigma)$. When the oracle is invoked with respect to a single key pair (i.e., $\mathsf{OClaim}_{(vk,sk)}$), we treat the oracle as taking only two inputs, $R$ and $\sigma$, since $i$ is superfluous in this case.

Additionally, we define the oracle $\mathsf{OClaim}^{\langle\sigma^*\rangle}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$ to output $\perp$ when it receives the signature $\sigma^*$ as input, and otherwise to give the same response as $\mathsf{OClaim}_{(vk_1,sk_1),\ldots,(vk_N,sk_N)}$.

Claimability requires three conditions, expressed by equations (5), (6), and (7) below. Informally, (5) requires that honest signers can successfully claim their signatures, (6) requires that adversarial parties cannot successfully claim a signature that they did not produce, and (7) requires that adversarial parties cannot produce a signature along with a claim that appears to be produced by an honest party (that is, falsely framing the honest party as the signer).

**Definition 3.9** (Claimability)**.** A ring signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is *claimable* if equipped with algorithms $(\mathsf{Claim}, \mathsf{VerClaim})$ such that the following conditions hold.

1. *(Honest signer can claim)* There exists a negligible function $\varepsilon$ such that for any $N = \mathsf{poly}(k)$ and $(vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k)$ and any $i \in [N]$, it holds for any message $m$ that

$$\Pr\left[\sigma \leftarrow \mathsf{Sign}(R, sk_i, m) : \mathsf{VerClaim}(R, vk_i, \sigma, \mathsf{Claim}(R, sk_i, \sigma)) = 1\right] > 1 - \varepsilon(k), \tag{5}$$

   where $R = \{vk_1, \ldots, vk_N\}$.[11]
2. *(Non-signers cannot claim)* Let $\mathcal{O} = \{\mathsf{OSign}\}$. For any (possibly adversarial) PPT sampling-and-claiming algorithm $\mathcal{A}_{\mathsf{Claim}} = (\mathcal{A}_1, \mathcal{A}_2)$, there exists a negligible function $\varepsilon$ such that

$$\Pr\left[\begin{array}{ll} (vk, sk) \leftarrow \mathsf{Gen}(1^k) & \\ (R', m, \mathfrak{s}) \leftarrow \mathcal{A}_1^{\mathcal{O},\mathsf{OClaim}_{(vk,sk)}}(vk) & \\ \sigma \leftarrow \mathsf{Sign}(R' \cup \{vk\}, sk, m) & : \quad b = 1 \wedge b' = 1 \\ (\zeta, vk') \leftarrow \mathcal{A}_2^{\mathcal{O},\mathsf{OClaim}_{(vk,sk)}}(R' \cup \{vk\}, \sigma, \mathfrak{s}) & \quad \wedge vk' \neq vk \\ b \leftarrow \mathsf{VerClaim}(R' \cup \{vk\}, vk', \sigma, \zeta) & \\ b' \leftarrow \mathsf{Verify}(R' \cup \{vk\}, \sigma, m) & \end{array}\right] < \varepsilon(k). \tag{6}$$

3. *(Malicious signer cannot frame an honest party)* For any PPT adversary $\mathcal{A}_{\mathsf{S\&C}}$, there exists a negligible function $\varepsilon$ such that

$$\Pr\left[\begin{array}{ll} (vk, sk) \leftarrow \mathsf{Gen}(1^k) & \\ (R', m, \sigma, \zeta) \leftarrow \mathcal{A}_{\mathsf{S\&C}}^{\mathcal{O},\mathsf{OClaim}_{(vk,sk)}}(vk) & : \quad b = 1 \wedge b' = 1 \\ b \leftarrow \mathsf{VerClaim}(R' \cup \{vk\}, vk, \sigma, \zeta) & \quad \wedge Q \cap \{(\cdot, \sigma)\} = \varnothing \\ b' \leftarrow \mathsf{Verify}(R' \cup \{vk\}, \sigma, m) & \end{array}\right] < \varepsilon(k). \tag{7}$$

   where $\mathcal{O} = \{\mathsf{OSign}\}$ and $Q$ is the set of queries made to oracle $\mathsf{OClaim}_{vk,sk}$.

[11] Like Definition 2.2, Equation 5 considers only honestly generated keys. See Remark 4 for further discussion.

### 3.3.1 Anonymity and unforgeability of claimable ring signatures

The definitions of anonymity and unforgeability must be adapted for claimable ring signature schemes, to allow the adversary a claim oracle as described next.

**Definition 3.10** (Anonymity of claimable ring signatures)**.** A claimable ring signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Claim}, \mathsf{VerClaim}))$ satisfies anonymity against adversarially chosen keys if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $(\{\mathsf{OSign}, \mathsf{OClaim}\}, \varnothing, 2)$-anonymous (Definition 2.3). Moreover, the repudiable ring signature satisfies *adaptive* anonymity against adversarially chosen keys if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is

$$(\{\mathsf{OSign}, \mathsf{OClaim}\}, \{\mathsf{OSign}, \mathsf{OClaim}^{\langle\sigma\rangle}\}, 2)\text{-anonymous},$$

where $\sigma$ is the challenge signature in the anonymity experiment (Equation (2)).

Recall from Remark 5 that adaptive anonymity against adversarially chosen keys is the strongest anonymity notion compatible with claimability.

**Definition 3.11** (Unforgeability of claimable ring signatures)**.** A claimable ring signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Claim}, \mathsf{VerClaim}))$ is *unforgeable* if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $\{\mathsf{OClaim}\}$-unforgeable (Definition 2.7).

## 3.4 Unclaimable ring signatures

An unclaimable ring signature scheme has the property that the signer cannot later convince anyone of her identity. That is, for any function that the true signer can compute given the signing randomness and the secret key, any other member of the ring can compute an indistinguishable function. The result is that even an adversary holding all ring members under duress cannot figure out who produced a given signature. This is true even if the ring members under duress attempt to cooperate with the adversary.

To achieve this, it suffices for any member of the ring to be able to extract signing randomness distributed indistinguishably from true signing randomness, that would produce the given signature under their secret key. More formally, the following guarantee should hold.

**Definition 3.12** (Unclaimable ring signatures)**.** A *unclaimable* ring signature scheme is a ring signature scheme augmented with an additional algorithm $\mathsf{ExtractRandomness}$ as follows.

- $\mathsf{ExtractRandomness}(R, sk, \sigma, m)$ takes as input a ring $R$, a secret key $sk$, a signature $\sigma$ and a message $m$. If $sk$ is one of the secret keys for ring $R$, and $\sigma$ is a signature on $m$ with respect to $R$, then it outputs randomness $\rho$.

$\mathsf{ExtractRandomness}$ must satisfy the following condition.

- *(Statistical unclaimability)* Let $\mathcal{R}$ be the distribution of signing randomness. For any $N = \mathsf{poly}(k)$ there is a negligible function $\epsilon$ such that the following holds. Let $(vk_1, sk_1), (vk_2, sk_2) \leftarrow \mathsf{Gen}(1^k)$. For any message $m$ and any $vk_3, \ldots, vk_N$ and $sk_3, \ldots, sk_N$, let $R = \{vk_1, \ldots, vk_N\}$ and $S = \{(i, vk_i, sk_i)\}_{i\in[N]}$. Let $\rho \leftarrow \mathcal{R}$, $\sigma_1 \leftarrow \mathsf{Sign}(R, sk_1, m; \rho)$, and $\rho_1 \leftarrow \mathsf{ExtractRandomness}(R, sk_2, \sigma_1, m)$. Let $\rho_2 \leftarrow \mathcal{R}$ and $\sigma_2 \leftarrow \mathsf{Sign}(R, sk_2, m; \rho_2)$. Then

$$(S, \rho_1, \sigma_1) \approx_\epsilon (S, \rho_2, \sigma_2).$$

Definition 3.12 is unusual among the definitions in this paper, in that it gives a statistical rather than a computational guarantee. We opted to give the statistical definition because it is simpler, it is a stronger guarantee, and our construction in this case achieves the statistical guarantee. One could also consider a computational definition.

*Remark* 7 (Claimability is not the opposite of unclaimability). According to these definitions, unclaimability is *not* technically the opposite of claimability (even when ignoring the fact that the formal definitions give a statistical guarantee for unclaimability but a computational guarantee for claimability). Claimability requires the ability to "voluntarily claim" a signature *without remembering the signing randomness*, whereas unclaimability rules out the ability to "claim under duress" *even given the signing randomness*. For voluntary claims, the natural and stronger definition is to guarantee the ability to claim adaptive, without "planning ahead" and without the storage requirement of remembering the signing randomness. In contrast, when considering attempts to claim under duress, the natural and stronger definition is to rule out the possibility of successful claims even in the presence of the signing randomness.

*Remark* 8 (Unclaimability protects *honest* signers). An adversarial signer who *wants* to claim can devise ways of credibly later claiming a ring signature, even when using an unclaimable ring signature scheme.[12] This does not decrease the utility of an unclaimable ring signaature scheme for honest signers who *want* their signatures to be unclaimable.

### 3.4.1 Unclaimability implies unrepudiability

Any unclaimable ring signature scheme is also unrepudiable. Recall that the definition of unclaimability captures the idea that for any function that the true signer can compute given the signing randomness and the secret key, any other member of the ring can compute an indistinguishable function. Intuitively, the implication follows from the fact that repudiation would require a non-signer to behave in a way that *distinguishable* from any possible behavior of the actual signer.

**Theorem 3.13.** *Any unclaimable ring signature scheme is also unrepudiable.*

*Proof.* Recall that unrepudiability (Definition 3.6) is defined as anonymity against full key exposure (Definition 2.6), which is $(\{\mathsf{OSign}\}, \varnothing, 2)$-anonymity under the framework of Definition 2.3. Thus, a ring signature scheme is unrepudiable if for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and polynomial $N$, it satisfies (2) of Definition 2.3 for $(\mathcal{O}_1, \mathcal{O}_2, \alpha) = (\{\mathsf{OSign}\}, \varnothing, 2)$. Note that since $\alpha = 2$, we can consider without loss of generality only adversaries that use the corruption oracle to learn all $N$ secret keys in (2), and do not make any queries to the signing oracle (since the adversary can produce signatures himself, using the secret keys).

To establish the theorem, it suffices to show that for any unclaimable ring signature scheme, the view of the adversary $\mathcal{A}$ is indistinguishable between the cases $b = 0$ and $b = 1$. Unclaimability (Definition 3.12) directly implies that these two views are indistinguishable. To see this, recall that the adversary's inputs in (2) of Definition 2.3 are an honestly generated set of verification keys $vk_1, \ldots, vk_N$ and a signature $\sigma$ produced by the honest signing algorithm using a secret key corresponding to some $vk_i, i \in [N]$. The theorem follows. □

[12] For example, an adversarial signer might use a PRG output as his signing randomness, or append it to his message, and remember the preimage. If he later revealed the preimage, it would likely serve as a credible claim to authorship of the signature.

*Remark* 9. It is unclear that the reverse implication holds even for a computational definition of unclaimability. The main complication is that unclaimability must hold even if the signing randomness is saved, while this is not an issue for unrepudiability. For instance, an algorithm that appends a commitment to the signing randomness (or to a random nonce) could be unrepudiable, but could be claimed by a signer who remembered the signing randomness.

### 3.5 Repudiable-and-claimable ring signatures

Suppose that $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is a ring signature scheme, and there are algorithms $\mathsf{Repudiate}$, $\mathsf{VerRepud}$, $\mathsf{Claim}$, and $\mathsf{VerClaim}$ such that

$$(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Repudiate}, \mathsf{VerRepud})) \quad \text{and} \quad (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Claim}, \mathsf{VerClaim}))$$

are a repudiable ring signature scheme and a claimable ring signature scheme respectively (Definitions 3.1 and 3.7). The seven algorithms together do *not* necessarily satisfy the natural notion of a "repudiable-and-claimable" ring signature scheme.

The reason, in a nutshell, is that Definition 3.1 (repudiability) does *not* allow the adversary a claim oracle, and likewise Definition 3.7 (claimability) does *not* allow the adversary a repudiation oracle. Indeed, it would not make sense even syntactically for the "other oracles" to be provided: since each of Definitions 3.1 and 3.7 is defined with respect to a *quintuple* of algorithms either containing $\mathsf{Repudiate}$ but not $\mathsf{Claim}$, or vice versa, the concept of the "other oracle" is undefined within the scope of each definition.

Thus, it could be that when an adversary has access to both a claim and a repudiation oracle, the resulting scheme is no longer secure. Indeed, there are simple (though arguably unnatural) examples of schemes where this happens, such as the following.

**Example 1.** Given *any* ring signature scheme, augment the signing key $sk$ to a new signing key $sk' = (sk, \eta_0, \eta_1)$ that additionally contains a pair $\eta_0, \eta_1$ such that $\eta_0$ is sampled uniformly randomly and $\eta_0 \oplus \eta_1 = sk$. $\mathsf{Sign}$ works just as in the original scheme, using only $sk$ and ignoring $\eta_0, \eta_1$. $\mathsf{Repudiate}$ produces repudiations just as in the original scheme, but additionally appends $\eta_0$ to every repudiation. $\mathsf{Claim}$ produces claims just as in the original scheme, but additionally appends $\eta_1$ to every repudiation. This modified scheme would be repudiable if the original scheme was, and also claimable if the original scheme was. However, an adversary that could see both a repudiation and a claim would straightforwardly be able to recover $sk$ and thereby forge signatures.

The natural security definition for a repudiable-and-claimable ring signature scheme is to include both repudiation and claim oracles throughout the repudiability, claimability, anonymity, and unforgeability definitions. As the resulting formal definitions are somewhat repetitive, we defer them to Appendix A.

## 4 Repudiable construction

### 4.1 Building blocks

**ZAPs** ZAPs are two-message public coin witness indistinguishable proofs [DN07].

**Definition 4.1** (ZAP)**.** A ZAP for an NP language $L$ with witness relation $\mathcal{R}_L$ is a triple of algorithms $\mathsf{ZAP}_L = (\mathsf{ZAP.Setup}_L, \mathsf{ZAP.Prove}_L, \mathsf{ZAP.Verify}_L)$, where $\mathsf{ZAP.Setup}$ and $\mathsf{ZAP.Prove}$ are PPT and $\mathsf{ZAP.Verify}$ is polynomial-time and deterministic, satisfying the following properties.

**Public coin.** For some polynomial $\ell = \ell(k)$, $\mathsf{ZAP.Setup}$ is the algorithm that on input $1^k$, outputs a uniformly random element of $\{0,1\}^\ell$.

**Completeness.** For $(x, w) \in \mathcal{R}_L$ and any $\rho \in \{0,1\}^{\ell(k)}$ we have

$$\Pr_{\pi \leftarrow \mathsf{ZAP.Prove}(\rho, x, w)}[\mathsf{ZAP.Verify}(\rho, \pi, x) = 1] = 1.$$

**Adaptive soundness.** There exists a negligible function $\epsilon$ such that

$$\Pr_{\rho \leftarrow \mathsf{ZAP.Setup}(1^k)}[\exists (x, \pi) : x \notin L \wedge \mathsf{ZAP.Verify}(\rho, \pi, x)] \leq \epsilon(k).$$

**Witness indistinguishability.** For any sequences $\{\rho_k\}_{k\in\mathbb{N}}$, $\{x_k\}_{k\in\mathbb{N}}$, $\{w_{0,k}\}_{k\in\mathbb{N}}$, $\{w_{1,k}\}_{k\in\mathbb{N}}$, where for all $k$, $\rho_k \in \{0,1\}^{\ell(k)}$, $x_k \in L$ and $(x_k, w_{0,k}), (x_k, w_{1,k}) \in \mathcal{R}_L$, the following pair of ensembles is computationally indistinguishable:

$$\{\mathsf{ZAP.Prove}(\rho_k, x_k, w_{0,k})\}_{k\in\mathbb{N}} \overset{c}{\approx} \{\mathsf{ZAP.Prove}(\rho_k, x_k, w_{1,k})\}_{k\in\mathbb{N}} .$$

In this work, for simplicity, we will assume use of a ZAP for some NP-complete language $L_{\mathrm{NP}}$ (with witness relation $\mathcal{R}_{L_{\mathrm{NP}}}$) and for any $L \in \mathrm{NP}$ with witness relation $\mathcal{R}_L$, we define $\mathsf{ZAP.Prove}_L$ and $\mathsf{ZAP.Verify}_L$ as follows.

- $\mathsf{ZAP.Prove}_L$ takes as input a triple $(\rho, x, w)$. If $(x, w) \notin \mathcal{R}_L$, then output $\perp$. Otherwise, use an NP reduction on $(x, w)$ to get a pair $(x_{\mathrm{NP}}, w_{\mathrm{NP}}) \in \mathcal{R}_{L_{\mathrm{NP}}}$, and output $\mathsf{ZAP.Prove}(\rho, x, w)$.
- $\mathsf{ZAP.Verify}_L$ takes as input a triple $(\rho, \pi, x)$, uses the same NP reduction to obtain $x_{\mathrm{NP}}$ (which is in $L_{\mathrm{NP}}$ iff $x \in L$), and outputs $\mathsf{ZAP.Verify}(\rho, \pi, x)$.

Verifiable random functions (VRFs) [MRV99] are another main building block of our construction. The important property of VRFs that we rely on is *residual pseudorandomness*, i.e., that VRF outputs on inputs for which the adversary has not received proofs remain indistinguishable from random.

**Definition 4.2** (VRF)**.** A *verifiable random function* (VRF) is a tuple of algorithms $\mathsf{VRF} = (\mathsf{VRF.Gen}, \mathsf{VRF.Eval}, \mathsf{VRF.Prove}, \mathsf{VRF.Verify})$, where $\mathsf{Gen}$ and $\mathsf{Verify}$ are PPT and $\mathsf{Eval}$ and $\mathsf{Prove}$ are polynomial time and deterministic, satisfying:

**Complete provability** With probability at least $1 - 2^{-\Omega(k)}$ over $(pk, sk) \leftarrow \mathsf{VRF.Gen}(1^k)$, we have for all inputs $x$ that

$$\Pr[\mathsf{VRF.Verify}(pk, x, \mathsf{VRF.Eval}(sk, x), \mathsf{VRF.Prove}(sk, x)) = 1] > 1 - 2^{-\Omega(k)} .$$

**Unique provability** For all $pk, x, y_1, y_2, \tau_1, \tau_2$ with $y_1 \neq y_2$, for either $i = 1$ or $i = 2$ it holds that

$$\Pr[\mathsf{VRF.Verify}(pk, x, y_i, \tau_i) = 1] < 2^{-\Omega(k)}.$$

**Residual pseudorandomness** Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a probabilistic polynomial-time adversary, where both $\mathcal{A}_1$ and $\mathcal{A}_2$ get oracle access to the VRF evaluation and prove algorithms. Let $(pk, sk) \leftarrow \mathsf{VRF.Gen}(1^k)$, and let

$$(x, \mathfrak{s}) \leftarrow \mathcal{A}_1^{\mathsf{VRF.Eval}(sk,\cdot),\mathsf{VRF.Prove}(sk,\cdot)}(1^k, pk).$$

Let $b \leftarrow \{0,1\}$, and let $v$ be either $\mathsf{VRF.Eval}(sk, x)$ or uniformly random, depending on the choice bit $b$. Let

$$b' = \mathcal{A}_2^{\mathsf{VRF.Eval}(sk,\cdot),\mathsf{VRF.Prove}(sk,\cdot)}(1^k, v, \mathfrak{s}).$$

Then there is a negligible function $\epsilon$ such that $\Pr[b = b' \text{ and } x \notin Q] < 1/2 + \epsilon(k)$, where $Q$ is the set of oracle queries made by $\mathcal{A}$ to either oracle.

For simplicity, we assume that $\mathsf{Eval}$ takes inputs $x$ of any length, i.e., $x \in \{0,1\}^*$.

**Definition 4.3.** The *verification failure probability* of a VRF $\mathsf{VRF}$ is

$$\Pr\left[\begin{array}{ll} (pk, sk) \leftarrow \mathsf{VRF.Gen}(1^k) & \\ b \leftarrow \mathsf{VRF.Verify}(pk, x, \mathsf{VRF.Eval}(sk, x), \mathsf{VRF.Prove}(sk, x)) & : \quad b = 0 \end{array}\right].$$

The residual pseudorandomness property of the VRF still holds even if the adversary gets to query many key pairs at once, and gets to adaptively choose to learn some of the secret keys (in this case, residual pseudorandomness holds for the uncorrupted keys only).

**Lemma 4.4** (Parallel VRF Game)**.** *Let* $\mathsf{VRF}$ *be a a VRF. Then* $\forall$ *PPT* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *and all* $N = \mathsf{poly}(k)$*, there is a negligible function* $\varepsilon$ *such that*

$$\Pr\left[\begin{array}{ll} (pk_1, sk_1), \ldots, (pk_N, sk_N) \leftarrow \mathsf{VRF.Gen}(1^k) & \\ (m^*, \mathfrak{s}) \leftarrow \mathcal{A}_1^{\mathcal{V},\mathsf{Corr}}(vk_1, \ldots, vk_N) & \\ \forall i \in [N], y_{i,0} \leftarrow \mathsf{VRF.Eval}(sk_i, m^*) & \\ \forall i \in [N], y_{i,1} \leftarrow \$ & : \quad b = b' \wedge \forall i \in [N] \setminus C, (i, m^*) \notin Q \\ b \leftarrow \{0,1\} & \\ b' \leftarrow \mathcal{A}_2(\mathfrak{s}, (y_{i,b})_{i \in [N] \setminus C}) & \end{array}\right] < 1/2 + \varepsilon(k), \tag{8}$$

*where* $\mathcal{V}$ *is an oracle that takes as input a pair* $(i, m)$ *and outputs*

$$(y, \tau) = (\mathsf{VRF.Eval}(sk_i, m), \mathsf{VRF.Prove}(sk_i, m)),$$

*and* $\mathsf{Corr}$ *is an oracle that takes as input an index* $i \in [N]$ *and outputs* $sk_i$*, and* $C$ *denotes the set of queries made to the corruption oracle, and* $Q$ *denotes the set of the queries to* $\mathcal{V}$*.*

We will refer to the game described by (8) as the Parallel VRF Game.

## 4.2 Construction

**Construction 4.5.** Our construction $\mathsf{R\text{-}RS}$ is parametrized by $\mathsf{ZAP}$, $\mathsf{VRF}$, and $M$, where

- $\mathsf{ZAP}$ is a ZAP;
- $\mathsf{VRF}$ is a VRF with input domain $\{0,1\}^*$, whose $\mathsf{Verify}$ algorithm takes $\nu$ bits of randomness and whose verification failure probability (Definition 4.3) is $\varepsilon$;[13]

[13] $\varepsilon$ need not be explicitly known, as discussed in footnote 14.

- $M$ is a polynomial satisfying $M \geq (\nu + k)/\log_2(1/\varepsilon)$.[14]

We first present the $\mathsf{Gen}$ algorithm of our ring signature scheme $\mathsf{R\text{-}RS}$.

$\mathsf{R\text{-}RS.Gen}(1^k)$

1. $(vk^1_{\mathsf{VRF}}, sk^1_{\mathsf{VRF}}), \ldots, (vk^4_{\mathsf{VRF}}, sk^4_{\mathsf{VRF}}) \leftarrow \mathsf{VRF.Gen}(1^k)$.
   Let $\vec{vk}_{\mathsf{VRF}} = (vk^1_{\mathsf{VRF}}, \ldots, vk^4_{\mathsf{VRF}})$ and $\vec{sk}_{\mathsf{VRF}} = (sk^1_{\mathsf{VRF}}, \ldots, sk^4_{\mathsf{VRF}})$.
2. $\rho \leftarrow \mathsf{ZAP.Setup}(1^k)$.
3. $\vec{\alpha} = (\alpha_1, \ldots, \alpha_M) \leftarrow (\{0,1\}^\nu)^M$.
4. Output $vk = (\vec{vk}_{\mathsf{VRF}}, \rho, \vec{\alpha})$ and $sk = (\vec{sk}_{\mathsf{VRF}}, vk)$.[15]

In the rest of the section, we (implicitly) use the following convention to parse a ring $R$.

$$\begin{aligned} &\textit{Write } R = \{vk_1, \ldots, vk_N\}. \\ &\textit{For each } i \in [N], \textit{write } vk_i = (\vec{vk}^{\,i}_{\mathsf{VRF}} = (vk^{i,1}_{\mathsf{VRF}}, \ldots, vk^{i,4}_{\mathsf{VRF}}), \rho_i, \vec{\alpha}_i = (\alpha^i_1, \ldots, \alpha^i_M)). \end{aligned} \tag{9}$$

**Definition 4.6.** Let $L$ be the following NP language.

$$\Big\{\big(R, m, \varphi, (y_1, y_2, y_3, y_4)\big) \;:\; \exists i^*, \tau_1, \tau_2, \tau_3, \tau_4, \gamma \text{ s.t. } (b_1 \vee b_2) \wedge (b_3 \vee b_4) \\ \text{where } \forall \eta \in \{1,2,3,4\}, b_\eta = \bigwedge_{i \in [N], j \in [M]} \mathsf{VRF.Verify}(vk^{i^*,\eta}_{\mathsf{VRF}}, (R, m, \varphi), y_\eta, \tau_\eta; \alpha^i_j \oplus \gamma)\Big\} .$$

We now present the $\mathsf{Sign}$ and $\mathsf{Verify}$ algorithms of our construction.

$\mathsf{R\text{-}RS.Sign}(R, sk, m)$

1. Parse $R$ as described above and $sk = ((sk^1_{\mathsf{VRF}}, \ldots, sk^4_{\mathsf{VRF}}), vk)$.
2. If $vk \notin R$ output $\perp$ and halt.
3. Define $i^* \in [N]$ such that $vk_{i^*} = vk$.
4. $\gamma \leftarrow \{0,1\}^\nu$. (This is used as part of the ZAP witness in Step 6.)
5. $\varphi \leftarrow \{0,1\}^k$. (This is used as a salt for the VRF input in Step 7, and output in Step 8.)
6. For $\eta \in \{1,2,3,4\}$, let $y_\eta = \mathsf{VRF.Eval}(sk^\eta_{\mathsf{VRF}}, (R, m, \varphi))$ and $\tau_\eta = \mathsf{VRF.Prove}(sk^\eta_{\mathsf{VRF}}, (R, m, \varphi))$.
   Let $\vec{y} = (y_1, \ldots, y_4)$.
7. For each $i \in [N]$, let $\pi_i \leftarrow \mathsf{ZAP.Prove}_L(\rho_i, (R, m, \varphi, \vec{y}), (i^*, \tau_1, \perp, \tau_3, \perp, \gamma))$.
   Let $\vec{\pi} = (\pi_1, \ldots, \pi_N)$.
8. Output $\sigma = (\vec{\pi}, \vec{y}, \varphi)$.

$\mathsf{R\text{-}RS.Verify}(R, \sigma, m)$

1. Parse $R$ as above and $\sigma = ((\pi_1, \cdots, \pi_N), \vec{y}, \varphi)$.
2. Output $\bigwedge_{i \in [N]} \mathsf{ZAP.Verify}_L(\rho_i, \pi_i, (R, m, \varphi, \vec{y}))$.

Now that we have described the main algorithms of $\mathsf{R\text{-}RS}$, we proceed to describe the repudiation algorithms for $\mathsf{R\text{-}RS}$.

[14] This inequality is required in order to invoke Corollary 4.11. As explained in Remark 11, a satisfactory value of $M$ can be set even without knowledge of $\varepsilon$. If $\varepsilon$ happens to be known, a smaller value of $M$ can be chosen.

[15] We include the verification key in $sk$ so that the $\mathsf{Sign}$ procedure can identify the verification key in the ring corresponding to the signing key.

**Definition 4.7.** Let $L'$ be the following NP language:

$$\left\{ \left(R, m, \varphi, (y_1, \ldots, y_4), vk = (\vec{vk}_{\mathsf{VRF}}, \rho, \vec{\alpha})\right) \ : \ \exists i^*, y'_1, \ldots, y'_4, \tau'_1, \ldots, \tau'_4, \gamma \text{ s.t.} \right.$$
$$\left( \left(b'_1 \wedge b'_2\right) \vee \left(b'_3 \wedge b'_4\right) \right) \wedge vk = vk_{i^*}, \text{ where } \forall \eta \in \{1,2,3,4\},$$
$$\left. b'_\eta = \left( y'_\eta \neq y_\eta \wedge \bigwedge_{i \in [N], j \in [M]} \mathsf{VRF.Verify}(vk_{\mathsf{VRF}}^{i^*,\eta}, (R, m, \varphi), y'_\eta, \tau'_\eta; \alpha_j^i \oplus \gamma) \right) \right\} .$$

$\mathsf{R\text{-}RS.Repudiate}(R, sk, \sigma)$

1. Parse $R$ as above, $sk = ((sk_{\mathsf{VRF}}^1, \ldots, sk_{\mathsf{VRF}}^4), vk)$, and $\sigma = (\vec{\pi}, \vec{y}, \varphi)$.
2. If $vk \notin R$ output $\perp$ and halt.
3. Define $i^* \in [N]$ such that $vk_{i^*} = vk$.
4. For $\eta \in \{1,2\}$: let $y'_\eta = \mathsf{VRF.Eval}(sk_{\mathsf{VRF}}^\eta, (R, m, \varphi))$ and let $\tau'_\eta = \mathsf{VRF.Prove}(sk_{\mathsf{VRF}}^\eta, (R, m, \varphi))$.
5. $\gamma \leftarrow \{0,1\}^\nu$. (This is used as part of the ZAP witness in Step 6.)
6. For each $i \in [N]$, let $\xi_i \leftarrow \mathsf{ZAP.Prove}_{L'}(\rho_i, (R, m, \varphi, \vec{y}, vk), (i^*, y'_1, y'_2, \perp, \perp, \tau'_1, \tau'_2, \perp, \perp, \gamma))$.
7. Output $\xi = (\xi_1, \ldots, \xi_N)$.

$\mathsf{R\text{-}RS.VerRepud}(R, vk, \sigma, \xi)$

1. Parse $R$ as above. If $vk \notin R$, output 1 and halt.
2. Parse $\sigma = (\vec{\pi}, \vec{y}, \varphi)$, and $\xi = (\xi_1, \ldots, \xi_N)$.
3. Output $\bigwedge_{i \in [N]} \mathsf{ZAP.Verify}_{L'}(\rho_i, \xi_i, (R, m, \varphi, \vec{y}, vk))$.

*Remark* 10. As written, the size of the VRF input $(R, m, \varphi)$ scales with the size of the ring $R$, and we have assumed that the VRF has input domain $\{0,1\}^*$, i.e., can take variable-length inputs. When this is not the case, or when it is desirable for efficiency reasons to evaluate the VRF on a smaller input, the scheme can be straightforwardly modified by employing a collision-resistant hash function, and evaluating the VRF on the *hash of* $(R, m, \varphi)$ rather than on $(R, m, \varphi)$ directly. We have presented the version of the scheme without the hash function, for simplicity of exposition.

## 4.3 Security proof

**Theorem 4.8.** *Let* VRF *be a VRF and* ZAP *be a ZAP. Then* R-RS *is a repudiable ring signature scheme.*

*Proof.* Follows directly from Lemmata 4.9 (correctness), 4.12 (repudiability), 4.13 (unforgeability), and 4.14 (anonymity). □

**Lemma 4.9** (Correctness of R-RS)**.** R-RS *satisfies correctness (Definition 2.2).*

Correctness is immediate so we omit the proof. Before presenting the proof of repudiability, we establish the following supporting lemma and corollary, which proceed according to an argument of [DN07].

**Lemma 4.10.** *Let $\mathcal{V}$ be a randomized algorithm that takes $\nu$ bits of randomness and outputs one bit. Let $\beta \in \{0,1\}$ be a bit, and let $x$ be an input such that for some negligible $\varepsilon$,*

$$\Pr\left[\mathcal{V}(1^k, x) = \beta\right] \geq 1 - \varepsilon \ . \tag{10}$$

*Let $M$ be a polynomial such that $M \geq (\nu + k)/\log_2(1/\varepsilon)$. (Note that the right-hand side is at most polynomial since the numerator is polynomial and the denominator is super-constant.) Then the following probability is overwhelming:*

$$\Pr_{(\alpha_1,\ldots,\alpha_M)\leftarrow(\{0,1\}^\nu)^M}\left[\forall \gamma \in \{0,1\}^\nu,\ \exists i \in [M]\ s.t.\ \mathcal{V}(1^k, x; \alpha_i \oplus \gamma) = \beta\right] . \tag{11}$$

*Proof.* Fix any $\gamma \in \{0,1\}^\nu$. Let $\psi_{i,\gamma} = \alpha_i \oplus \gamma$ for each $i \in [M]$. Since the $\alpha_i$ are distributed randomly and independently, the distribution of $(\psi_{i,\gamma})_{i\in[M]}$ is uniform over $(\{0,1\}^\nu)^M$ even when conditioned on $\gamma$. Therefore, conditioned on any given $\gamma$,

$$\Pr\left[\forall i \in [M],\ \mathcal{V}(1^k, x, ; \psi_{i,\gamma}) \neq \beta\right] < \varepsilon^M .$$

There are $2^\nu$ possible values of $\gamma$, so by a union bound,

$$\Pr\left[\exists \gamma \in \{0,1\}^\nu \text{ s.t. } \forall i \in [M],\ \mathcal{V}(1^k, x; \psi_{i,\gamma}) \neq \beta\right] < 2^\nu \cdot \varepsilon^M .$$

Since $M \geq (\nu + k)/\log_2(1/\varepsilon) = \log_\varepsilon(2^{-(\nu+k)})$ by assumption, the right-hand side is at most $2^{-k}$, which is negligible. The lemma follows. □

The next corollary states that the implication established in Lemma 4.10 in fact goes both ways.

**Corollary 4.11.** *Let $\mathcal{V}$ be a randomized algorithm takes $\nu$ bits of randomness and outputs one bit. Let $\beta \in \{0,1\}$ be a bit, and let $\varepsilon$ be a negligible function. Let $M$ be a polynomial such that $M \geq (\nu + k)/\log_2(1/\varepsilon)$. Then (10) holds if and only if (11) is overwhelming.*

*Proof.* Follows from applying Lemma 4.10 for both $\beta = 0$ and $\beta = 1$. □

*Remark* 11. For all large enough $k$, Corollary 4.11 holds for any $M \geq \nu + k$. This is because if (10) holds for some negligible $\varepsilon$, then for all large enough $k$, (10) also holds for $\varepsilon = 1/2$ (or indeed, any constant $\varepsilon$). Substituting $\varepsilon = 1/2$ into $M \geq (\nu + k)/\log_2(1/\varepsilon)$ yields $M \geq \nu + k$. In particular, this means that a satisfactory $M$ can be chosen without knowledge of $\varepsilon$.

Next, we give the proof of repudiability of R-RS.

**Lemma 4.12** (Repudiability of R-RS)**.** R-RS *is repudiable (Definition 3.3).*

*Proof.* Suppose, for contradiction, that R-RS is not repudiable. Then by Definition 3.3, it must be that either (3) or (4) does not hold. We consider these two possibilities in turn.

Suppose first that (3) does not hold for R-RS. Then there is a PPT $\mathcal{A}_{\mathsf{Sign}}$ that generates a valid signature $\sigma$ with respect to some ring $R$, so that $\sigma$ is not repudiable by some honest party in the ring. That is, the following probability is non-negligible:

$$\Pr\left[\begin{array}{l|ll} & (vk, sk) \leftarrow \mathsf{Gen}(1^k) & \\ & (\sigma, m, R) \leftarrow \mathcal{A}^{\mathcal{O}}_{\mathsf{Sign}}(vk) & \\ & \xi \leftarrow \mathsf{Repudiate}(R, sk, \sigma) & : \quad b = 0 \wedge b' = 1 \\ & b \leftarrow \mathsf{VerRepud}(R, vk, \sigma, \xi) & \quad \wedge Q \cap \{(\cdot, m, R')\} = \varnothing \\ & b' \leftarrow \mathsf{Verify}(R, \sigma, m) & \end{array}\right] , \tag{12}$$

where $Q$ is the set of queries to $\mathsf{OSign}$.

Based on $\mathcal{A}_{\mathsf{Sign}}$, we build an adversary $\mathcal{A}$ for Parallel VRF Game (defined in Lemma 4.4), as follows. $\mathcal{A}$ first invokes $\mathsf{R\text{-}RS.Gen}(1^k)$ to obtain $(vk, sk)$. The $vk$ is a tuple $(\vec{vk}_{\mathsf{VRF}}, \rho, \vec{\alpha})$, where $\vec{vk}_{\mathsf{VRF}}$ can be parsed further as $(vk^1_{\mathsf{VRF}}, \ldots, vk^4_{\mathsf{VRF}})$.

$\mathcal{A}$ obtains two verification keys $vk^{3,*}_{\mathsf{VRF}}, vk^{4,*}_{\mathsf{VRF}}$ from the VRF challenger, and replaces $vk^3_{\mathsf{VRF}}$ and $vk^4_{\mathsf{VRF}}$ with these keys, setting

$$vk' = ((vk^1_{\mathsf{VRF}}, vk^2_{\mathsf{VRF}}, vk^{3,*}_{\mathsf{VRF}}, vk^{4,*}_{\mathsf{VRF}}), \rho, \vec{\alpha}).$$

Let $sk^{3,*}_{\mathsf{VRF}}, sk^{4,*}_{\mathsf{VRF}}$ be the VRF secret keys corresponding to $vk^{3,*}_{\mathsf{VRF}}, vk^{4,*}_{\mathsf{VRF}}$, respectively.[16]

Next, $\mathcal{A}$ runs $\mathcal{A}_{\mathsf{Sign}}(vk')$, answering $\mathcal{A}_{\mathsf{Sign}}$'s oracle queries as follows.

- On query $(m'', R'')$ to $\mathsf{OSign}$: $\mathcal{A}$ runs the honest signing algorithm $\mathsf{R\text{-}RS.Sign}$ on input $(R'' \cup \{vk'\}, sk, m'')$, with the following modification: in step 6, instead of using $sk^3_{\mathsf{VRF}}$ and $sk^4_{\mathsf{VRF}}$ to generate $y_3, \tau_3$, and $y_4, \tau_4$, $\mathcal{A}$ invokes its VRF oracle.
- On query $(\sigma'', R'')$ to $\mathsf{ORpd}$: $\mathcal{A}$ runs the honest repudiation algorithm $\mathsf{R\text{-}RS.Repudiate}$ on input $(R'' \cup \{vk'\}, sk, \sigma'')$. (Note that $sk^3_{\mathsf{VRF}}$ and $sk^4_{\mathsf{VRF}}$ are not used in algorithm $\mathsf{R\text{-}RS.Repudiate}$, so we don't need to invoke the VRF oracle here.)

Let $(\sigma, m, R)$ be the output of $\mathcal{A}_{\mathsf{Sign}}$. $\mathcal{A}$ parses $\sigma = (\vec{\pi}, \vec{y}, \varphi)$ and $\vec{y} = (y_1, \ldots, y_4)$. Then $\mathcal{A}$ sends $(R, m, \varphi)$ to the VRF challenger, receiving responses $y'_3$ and $y'_4$. If $y_3 = y'_3$ or $y_4 = y'_4$, $\mathcal{A}$ outputs 0. Otherwise, $\mathcal{A}$ outputs a random bit. Let us now consider $\mathcal{A}$'s behavior in the two cases where the VRF challenger's bit $b$ is equal to 0 and equal to 1.

**Case $b = 0$ in Parallel VRF Game.** In this case, the view of $\mathcal{A}_{\mathsf{Sign}}$ is identical to the view in (12), so by assumption $\mathcal{A}_{\mathsf{Sign}}$ will win the game — i.e., produce a signature that verifies but is not repudiable by an honest party — with non-negligible probability. Note that whenever $\mathcal{A}_{\mathsf{Sign}}$ wins the game, the condition $Q \cap \{(\cdot, m, R')\} \neq \varnothing$ in (12) implies that $\mathcal{A}$ has not previously made an oracle query on the VRF challenge message $(R, m, \varphi)$ during the query phase. Let us suppose that $\mathcal{A}_{\mathsf{Sign}}$ wins the game described in (12), and consider the implications.

By definition, if $\mathsf{R\text{-}RS.VerRepud}$ rejects (with non-negligible probability) on an honestly generated repudiation $\xi = (\xi_i)_{i \in [|R|]}$ generated with respect to $vk'$, then

$$\exists i \in [|R|] \text{ s.t. } \mathsf{ZAP.Verify}_{L'}(\rho_i, \xi_i, (R, m, \varphi, \vec{y}, vk')) = 0\ . \tag{13}$$

By the completeness of the ZAP and the complete provability of the VRF, since $\xi$ is honestly generated with respect to $vk'$, the statement $\neg b'_3 \vee \neg b'_4$ then holds with overwhelming probability, where $b'_3, b'_4$ are as defined in Definition 4.7. Expanding the definition of $b'_3, b'_4$, and again using that $\xi$ is honestly generated, we have that

$$\begin{gathered}\exists \eta \in \{3,4\}, i' \in [|R|], j' \in [M], \gamma \text{ s.t.} \\ \mathsf{VRF.Verify}(vk^{\eta,*}_{\mathsf{VRF}}, (R, m, \varphi), y_\eta, \mathsf{VRF.Prove}(sk^{\eta,*}_{\mathsf{VRF}}, (R, m, \varphi)); \alpha^{i'}_{j'} \oplus \gamma) = 0\ .\end{gathered} \tag{14}$$

Since $vk' \in R$ is honestly generated by assumption, we have that the $\vec{\alpha} = (\alpha_1, \ldots, \alpha_M)$ within $vk'$ is distributed uniformly over $(\{0,1\}^\nu)^M$. Then applying Corollary 4.11 (setting the algorithm $\mathcal{V}$ to be $\mathsf{VRF.Verify}$): (14) implies *either*

$$\begin{gathered}\exists \eta \in \{3,4\} \text{ and } \tau \text{ s.t.} \\ \Pr\left[\mathsf{VRF.Verify}(vk^{\eta,*}_{\mathsf{VRF}}, (R, m, \varphi), y_\eta, \tau) = 1\right] \text{ is overwhelming,}\end{gathered} \tag{15}$$

[16] Note that $sk^{3,*}_{\mathsf{VRF}}, sk^{4,*}_{\mathsf{VRF}}$ are generated by the VRF challenger and not accessible by $\mathcal{A}$.

*or* a negligible probability event occurred. By the complete and unique provability of the VRF, (15) implies that

$$y_3 = \mathsf{VRF.Eval}(sk^{3,*}_{\mathsf{VRF}}, (R, m, \varphi)) \text{ or } y_4 = \mathsf{VRF.Eval}(sk^{4,*}_{\mathsf{VRF}}, (R, m, \varphi)) \ . \tag{16}$$

Chaining together the implications, we conclude that (16) holds with all but negligible probability conditioned on the non-negligible-probability event of $\mathcal{A}_{\mathsf{Sign}}$ winning the game described in (12).

Finally, by definition of Parallel VRF Game, when $b = 0$,

$$y'_3 = \mathsf{VRF.Eval}(sk^{3,*}_{\mathsf{VRF}}, (R, m, \varphi)) \text{ and } y'_4 = \mathsf{VRF.Eval}(sk^{4,*}_{\mathsf{VRF}}, (R, m, \varphi)) \ . \tag{17}$$

From (16) and (17): when $b = 0$, there is a non-negligible probability that

$$y_3 = y'_3 \text{ or } y_4 = y'_4 \ . \tag{18}$$

Recall that (18) is the trigger condition for $\mathcal{A}$ to output 0. Therefore, when $b = 0$, $\mathcal{A}$ outputs 0 with non-negligible probability (and outputs a random bit the rest of the time).

**Case $b = 1$ in Parallel VRF Game.** In this case, $y'_3$ and $y'_4$ are uniformly random and independent of the rest of the experiment, so they will be distinct from $y_3$ and $y_4$ with overwhelming probability. Consequently, in this case, with all but negligible probability $\mathcal{A}$ outputs a random bit.

Thus, $\mathcal{A}$ wins Parallel VRF Game with non-negligible probability. This contradicts the security of the VRF. Therefore, R-RS satisfies (3).

It remains to show that R-RS satisfies (4). The argument for this part of the proof follows a somewhat similar outline to the argument already presented. Due to space constraints, the rest of the proof is deferred to Appendix B. □

**Lemma 4.13** (Unforgeability of R-RS)**.** R-RS *is unforgeable (in the sense of Definition 3.5).*

*Proof.* This proof is very similar to the second half of the proof of Lemma 4.12.

Suppose that this is not the case. Then there exists some $N = \mathsf{poly}(k)$ and some PPT $\mathcal{B}$ such that the following probability is non-negligible, where $I$ and $Q$ are the sets of queries made to the corruption and signing oracles respectively:

$$\Pr\left[\begin{array}{ll} (vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k) & \\ (R^*, m^*, \sigma^*) \leftarrow \mathcal{B}^{\mathsf{OSign},\mathsf{ORpd},\mathsf{Corr}}(vk_1, \ldots, vk_N) & : \quad b = 1 \wedge R^* \subseteq \{vk_1, \ldots, vk_N\} \setminus I \\ b \leftarrow \mathsf{Verify}(R^*, \sigma^*, m^*) & \quad \wedge Q \cap \{(\cdot, m^*, R^*)\} = \varnothing \end{array}\right] . \tag{19}$$

We build an adversary $\mathcal{A}$ to the Parallel VRF Game of Lemma 4.4. $\mathcal{A}$ first samples

$$(vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k)$$

and parses each $vk_i$ and $sk_i$ as follows:

$$vk_i = (\vec{vk}^{\,i}_{\mathsf{VRF}} = (vk^{i,1}_{\mathsf{VRF}}, vk^{i,2}_{\mathsf{VRF}}, vk^{i,3}_{\mathsf{VRF}}, vk^{i,4}_{\mathsf{VRF}}), \rho_i, \vec{\alpha}_i)$$
$$sk_i = (\vec{sk}^{\,i}_{\mathsf{VRF}} = (sk^{i,1}_{\mathsf{VRF}}, sk^{i,2}_{\mathsf{VRF}}, sk^{i,3}_{\mathsf{VRF}}, sk^{i,4}_{\mathsf{VRF}}), vk_i)$$

Then $\mathcal{A}$ chooses a random $i^* \leftarrow [N]$, obtains a pair of verification keys $vk^{*,3}_{\mathsf{VRF}}, vk^{*,4}_{\mathsf{VRF}}$ from the VRF challenger, and lets

$$vk^* = (\vec{vk}^{\,i}_{\mathsf{VRF}} = (vk^{i,1}_{\mathsf{VRF}}, vk^{i,2}_{\mathsf{VRF}}, vk^{*,3}_{\mathsf{VRF}}, vk^{*,4}_{\mathsf{VRF}}), \rho_i, \vec{\alpha}_i).$$

Let $sk^{*,3}_{\mathsf{VRF}}, sk^{*,4}_{\mathsf{VRF}}$ denote the VRF secret keys corresponding to $vk^{*,3}_{\mathsf{VRF}}, vk^{*,4}_{\mathsf{VRF}}$, respectively.

Let $vk^*_{i^*} = vk^*$ and let $vk^*_i = vk_i$ for every $i \neq i^*$. Let $R^* = \{vk^*_1, \ldots, vk^*_N\}$. $\mathcal{A}$ then runs $\mathcal{B}$ on input $R^*$, answering $\mathcal{B}$'s oracle queries as follows.

- On query $(i'', m'', R'')$ to $\mathsf{OSign}$: $\mathcal{A}$ runs the honest signing algorithm $\mathsf{R\text{-}RS.Sign}$ on input $(R'' \cup \{vk^*_{i''}\}, sk_{i''}, m'')$, with the following modification if $i = i^*$: in step 6, instead of using $sk^{i^*,3}_{\mathsf{VRF}}$ and $sk^{i^*,4}_{\mathsf{VRF}}$ to generate $y_3, \tau_3$, and $y_4, \tau_4$, $\mathcal{A}$ invokes its VRF oracle.
- On query $(i'', \sigma'', R'')$ to $\mathsf{ORpd}$: $\mathcal{A}$ runs the honest repudiation algorithm $\mathsf{R\text{-}RS.Repudiate}$ on input $(R'' \cup \{vk^*_{i''}\}, sk_{i''}, \sigma'')$. ($sk^3_{\mathsf{VRF}}$ and $sk^4_{\mathsf{VRF}}$ are not used by $\mathsf{R\text{-}RS.Repudiate}$, so $\mathcal{A}$ does not need to invoke the VRF oracle here.)
- On each invocation of $\mathsf{Corr}$ on some index $i \in [N]$: if $i = i^*$ then $\mathcal{A}$ outputs a random bit and aborts; otherwise, $\mathcal{A}$ responds to $\mathcal{B}$ with $sk_i$.

Let $(R', m, \sigma)$ be the output of $\mathcal{B}$. $\mathcal{A}$ parses $\sigma = (\vec{\pi}, \vec{y}, \varphi)$ and $\vec{y} = (y_1, \ldots, y_4)$, and submits $(R', m, \varphi)$ to the VRF challenger and then receive responses $y'_3$ and $y'_4$. If $y'_3 = y_3$ or $y'_4 = y_4$, $\mathcal{A}$ outputs 0. Else, $\mathcal{A}$ outputs a random bit.

It remains to show that $\mathcal{A}$ distinguishes with non-negligible advantage, in the parallel VRF security game, between VRF outputs and random values.

Let us consider the behavior of $\mathcal{A}$ in the two cases where the VRF challenger's bit is equal to 0 and equal to 1.

**Case $b = 0$ in Parallel VRF Game.** In this case, the response that $\mathcal{A}$ receives to the challenge $(R', m)$ consists of VRF outputs on input $(R', m)$ with respect to the keys $vk^{*,3}_{\mathsf{VRF}}, vk^{*,4}_{\mathsf{VRF}}$. In particular, whenever $\mathcal{B}$ does not query $\mathsf{Corr}$ on $i^*$, the view of $\mathcal{B}$ is identical to the view in (19). So, conditioned on $\mathcal{B}$ not corrupting $i^*$, $\mathcal{B}$ will win the game — i.e., produce a valid signature — with non-negligible probability, by assumption.

Let us consider the probability that $\mathcal{B}$ queries $\mathsf{Corr}$ for input $i^*$. Recall that this event causes $\mathcal{A}$ to abort and output a random bit. The distribution of the view (i.e., verification keys and oracle responses) of $\mathcal{B}$ is unaffected by $\mathcal{A}$'s choice of $i^*$, until the point at which $\mathcal{B}$ submits an oracle query to $\mathsf{Corr}$ for input $i^*$ (if at all). The condition $R^* \subseteq \{vk_1, \ldots, vk_N\} \setminus I$ in (19) ensures that if $\mathcal{B}$ wins the game with non-negligible probability, then $\mathcal{B}$ leaves one or more keys uncorrupted with at least that non-negligible probability. Since $i^*$ is chosen at random by $\mathcal{A}$, it follows that $\Pr[i^* \notin I]$ is non-negligible.

Let $E'$ denote the event that $\mathcal{A}$ does not abort (i.e., $i^* \notin I$) and $\mathcal{B}$'s output signature verifies (i.e., $\mathsf{R\text{-}RS.Verify}(R', \sigma, m) = 1$). We have established that $E'$ occurs with non-negligible probability. Then, by the same argument given from (28) to (32) in the proof of Lemma 4.12, we have that *either*

$$\exists j^* \in [|R'|], \eta \in \{3, 4\}, \text{ and } \tau \text{ s.t.}$$
$$\Pr\left[\mathsf{VRF.Verify}(vk^{j^*,\eta}_{\mathsf{VRF}}, (R', m, \varphi), y_\eta, \tau) = 1\right] \text{ is overwhelming,}$$

*or* a negligible probability event occurred. When $j^* = i^*$, this moreover implies

$$y_3 = \mathsf{VRF.Eval}(sk^{*,3}_{\mathsf{VRF}}, (R', m, \varphi)) \text{ or } y_4 = \mathsf{VRF.Eval}(sk^{*,3}_{\mathsf{VRF}}, (R', m, \varphi)) \ . \tag{20}$$

Let us consider the probability that $j^* = i^*$. As also observed above, the distribution of the view (i.e., verification keys and oracle responses) of $\mathcal{B}$ is unaffected by $\mathcal{A}$'s choice of $i^*$, until the point at which $\mathcal{B}$ submits an oracle query to $\mathsf{Corr}$ for input $i^*$ (if at all). Since $i^*$ is chosen at

random by $\mathcal{A}$ (and is thus independent of $j^*$), and $i^*, j^* \in [|R'|]$, $\Pr[i^* = j^*]$ must be non-negligible. Therefore, (20) holds with non-negligible probability.

Finally, by definition of the Parallel VRF Game, whenever the challenger's bit $b$ is 0,

$$y'_3 = \mathsf{VRF.Eval}(sk^{*,3}_{\mathsf{VRF}}, (R', m, \varphi)) \text{ or } y'_4 = \mathsf{VRF.Eval}(sk^{*,3}_{\mathsf{VRF}}, (R', m, \varphi)) \ . \tag{21}$$

From (20) and (21) we have that with non-negligible probability,

$$y_3 = y'_3 \text{ or } y_4 = y'_4 \ . \tag{22}$$

Recall that (22) is the trigger condition for $\mathcal{A}$ to output 0. We conclude that when the VRF challenger's bit $b = 0$, the trigger condition for $\mathcal{A}$ to output 0 is met with non-negligible probability; and by construction, $\mathcal{A}$ outputs a random bit the rest of the time (i.e., when the trigger condition is not met).

**Case $b = 1$ in Parallel VRF Game.** In this case, the response that $\mathcal{A}$ receives to the challenge message $(R, m, \varphi)$ consists of truly random strings instead of VRF outputs, and so $\Pr[y'_3 = y_3 \vee y'_4 = y_4]$ is negligible. Thus, $\mathcal{A}$ outputs a random bit with overwhelming probability.

We have shown that $\mathcal{A}$'s output is non-negligibly different depending on the VRF challenger's bit, and so $\mathcal{A}$ must win the Parallel VRF Game with non-negligible probability. This contradicts the security of the VRF. Therefore, R-RS is unforgeable. □

**Lemma 4.14** (Anonymity of R-RS)**.** R-RS *satisfies adaptive anonymity against adversarially chosen keys (Definition 3.4).*

*sketch.* The proof is a hybrid argument using the security of VRFs and ZAPs to change the values $y_2$ and $y_4$ within the signature first to truly random values, then to VRF outputs w.r.t. party $i^*_1$ rather than $i^*_0$. Then the same procedure is applied to $y_1$ and $y_3$, so that finally $y_1, \ldots, y_4$ are all VRF outputs w.r.t. party $i^*_1$. Details of the proof are in Appendix C due to space constraints. □

# 5 Claimable transformation

In this section, we give a simple black-box transformation from any ring signature to a claimable ring signature scheme. The transformation relies on one-way functions. If the original scheme was repudiable, the resulting scheme is moreover *claimable-and-repudiable*.

## 5.1 Building blocks

We assume familiarity with the standard notions of commitment schemes, standard signatures, and PRFs, and simply establish syntax in this subsection.[17] For simplicity, we denote a commitment scheme by a single algorithm Com, and assume that the decommitment simply consists of the commitment randomness.

**Definition 5.1** (Commitment scheme)**.** A *commitment scheme* Com has the following syntax: Com takes as input a message $\mu$ and randomness $r$ and outputs a commitment $c$. Sometimes we leave $r$ implicit and write $\mathsf{Com}(\mu)$ instead of $\mathsf{Com}(\mu; r)$. The *decommitment* of $c$ is the randomness $r$.

A commitment scheme must satisfy the following properties.

[17] We refer to any standard textbook (e.g., [KL14]) for the relevant security definitions.

- *Hiding:* For all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\exists$ negligible $\varepsilon$ s.t. $\forall k \in \mathbb{N}$,

$$\Pr\left[\begin{array}{ll} (\mu_0, \mu_1, \mathfrak{s}) \leftarrow \mathcal{A}_1(1^k) & \\ b \leftarrow \{0,1\} & \\ c \leftarrow \mathsf{Com}(\mu_b) & : \quad b' = b \\ b' \leftarrow \mathcal{A}_2(c, \mathfrak{s}) & \end{array}\right] \leq 1/2 + \varepsilon(k) \ . \tag{23}$$

- *Binding:* For all PPT adversaries $\mathcal{A}$, $\exists$ negligible $\varepsilon$ s.t. $\forall k \in \mathbb{N}$,

$$\Pr\left[\quad (c, \mu, r, \mu', r') \leftarrow \mathcal{A}(1^k) \quad : \quad \mu \neq \mu' \wedge \mathsf{Com}(\mu; r) = c = \mathsf{Com}(\mu'; r') \quad\right] \leq \varepsilon(k) \ .$$

**Definition 5.2** (Standard signature (syntax))**.** A *signature scheme* is a triple of PPT algorithms $\Sigma = (\Sigma.\mathsf{Gen}, \Sigma.\mathsf{Sign}, \Sigma.\mathsf{Verify})$ with the following syntax:

- $\Sigma.\mathsf{Gen}(1^k)$ takes as input the security parameter $k$ and outputs a *verification key* $vk$ and a *signing key* $sk$.
- $\Sigma.\mathsf{Sign}(sk, m)$ takes as input a signing key $sk$ and a message $m$, and outputs a signature $\sigma$.
- $\Sigma.\mathsf{Verify}(vk, \sigma, m)$ takes as input a verification key $vk$, a signature $\sigma$, and a message $m$, and outputs a single bit indicating whether or not $\sigma$ is a valid signature on $m$ w.r.t. $vk$.

**Definition 5.3** (PRF (syntax))**.** A *pseudorandom function (PRF)* is a pair of algorithms $\mathsf{PRF} = (\mathsf{PRF.Gen}, \mathsf{PRF.Eval})$, where:

- $\mathsf{PRF.Gen}$ is a PPT algorithm that takes as input $1^k$ and outputs a *PRF key* $sk_{\mathsf{PRF}}$, and
- $\mathsf{PRF.Eval}$ is a polynomial-time deterministic algorithm that takes as input a PRF key $sk_{\mathsf{PRF}}$ and $x \in \{0,1\}^*$ and outputs a string $r$.

For simplicity, we assume PRFs that take arbitrary-length inputs (i.e., $\{0,1\}^*$).

## 5.2 The transformation

Our transformation builds on any ring signature scheme, $\mathsf{RS}$, to construct a claimable ring signature scheme $\mathsf{C\text{-}RS}$. The basic idea is to take a signature $\sigma_{\mathsf{RS}}$ under $\mathsf{RS}$ and append to it a *commitment* $c$ to $(vk, \sigma_{\mathsf{RS}})$ where $vk$ is the verification key of the signer. The verification algorithm simply checks whether $\sigma_{\mathsf{RS}}$ verifies. The claim consists of a decommitment revealing that $c$ is a commitment to $(vk, \sigma_{\mathsf{RS}})$. Intuitively, by the hiding property of the commitment scheme, the identity of the signer is hidden until he chooses to publish a claim.

The simple transformation just described runs into a couple of problems when examined in detail. First, what if a signer commits to $(\sigma_{\mathsf{RS}}, vk')$ where $vk'$ is not his own key but that of someone else in the ring? This ability would violate equation (6) of Definition 3.9 (claimability). To prevent such behavior, our construction actually commits to a *standard (non-ring) signature* on $(vk, \sigma_{\mathsf{RS}})$. The unforgeability property of standard signatures then guarantees, intuitively, that a signer cannot convincingly make a claim with respect to any verification key unless he knows a corresponding signing key.

A second hurdle encountered by the scheme thus far described is that the signer must remember the commitment randomness in order to produce a claim. It is preferable that the signer need not be stateful in between signing and claiming; and indeed, recall that Definition 3.9 formalizes this property. To resolve this, our construction derives commitment randomness from a PRF. For similar reasons, the signing randomness for the standard (non-ring) signature in our construction is also derived from a PRF.

The formal description of the transformation follows.

**Construction 5.4.** Our transformation C-RS is parametrized by the following:

- RS, a ring signature scheme,
- $\Sigma$, a standard signature scheme,
- Com, a commitment scheme, and
- PRF, a PRF.

For convenience, and without loss of generality, we assume that the commitment randomness of Com, the signing randomness of $\Sigma$, and the output of PRF.Eval all have the same length of $\nu$ bits.

C-RS.Gen$(1^k)$

1. Let $(vk_{\mathsf{RS}}, sk_{\mathsf{RS}}) \leftarrow \mathsf{RS.Gen}(1^k)$.
2. Let $(vk_{\Sigma}, sk_{\Sigma}) \leftarrow \Sigma.\mathsf{Gen}(1^k)$.
3. Let $sk_{\mathsf{PRF}} \leftarrow \mathsf{PRF.Gen}(1^k)$.
4. Output $vk = (vk_{\mathsf{RS}}, vk_{\Sigma})$ and $sk = (vk, sk_{\mathsf{RS}}, sk_{\Sigma}, sk_{\mathsf{PRF}})$.

In the rest of the construction, we implicitly parse verification keys and signing keys of C-RS as $vk = (vk_{\mathsf{RS}}, vk_{\Sigma})$ and $sk = (vk, sk_{\mathsf{RS}}, sk_{\Sigma}, sk_{\mathsf{PRF}})$ respectively. Also, for a ring

$$R = \left(vk_1 = (vk^1_{\mathsf{RS}}, vk^1_{\Sigma}), \ldots, vk_N = (vk^N_{\mathsf{RS}}, vk^N_{\Sigma})\right) \ ,$$

we write $\mathsf{RS}(R)$ to denote $(vk^1_{\mathsf{RS}}, \ldots, vk^N_{\mathsf{RS}})$.

C-RS.Sign$(R, sk, m)$

1. Let $\sigma_{\mathsf{RS}} \leftarrow \mathsf{RS.Sign}(\mathsf{RS}(R), sk_{\mathsf{RS}}, m)$.
2. Let $r_{\Sigma} = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 0))$.
3. Let $\sigma_{\Sigma} = \Sigma.\mathsf{Sign}(sk_{\Sigma}, (vk, \sigma_{\mathsf{RS}}); r_{\Sigma})$.
4. Let $r_{\mathsf{Com}} = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 1))$.
5. Let $c = \mathsf{Com}((vk, \sigma_{\Sigma}); r_{\mathsf{Com}})$.
6. Let $\sigma = (\sigma_{\mathsf{RS}}, c)$.
7. If $\mathsf{C\text{-}RS.VerClaim}(R, vk, \sigma, \mathsf{C\text{-}RS.Claim}(R, sk, \sigma)) = 1$, output $\sigma$.
8. Otherwise, output $(\perp, \perp)$.

C-RS.Verify$(R, \sigma = (\sigma_{\mathsf{RS}}, c), m)$

1. If $\sigma_{\mathsf{RS}} = \perp$, output 0.
2. Otherwise, output $\mathsf{RS.Verify}(\mathsf{RS}(R), \sigma_{\mathsf{RS}}, m)$.

C-RS.Claim$(R, sk, \sigma = (\sigma_{\mathsf{RS}}, c))$

1. Let $r'_{\Sigma} = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 0))$.
2. Let $r'_{\mathsf{Com}} = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 1))$.
3. Let $\sigma'_{\Sigma} = \Sigma.\mathsf{Sign}(sk_{\Sigma}, (vk, \sigma_{\mathsf{RS}}); r'_{\Sigma})$.
4. If $c \neq \mathsf{Com}(\sigma'_{\Sigma}, r'_{\mathsf{Com}})$, output $\zeta = \perp$.
5. Otherwise, output $\zeta = (r'_{\mathsf{Com}}, \sigma'_{\Sigma})$.

C-RS.VerClaim$(R, vk, \sigma = (\sigma_{\mathsf{RS}}, c), \zeta = (r'_{\mathsf{Com}}, \sigma'_{\Sigma}))$

1. Let $c' = \mathsf{Com}((vk, \sigma'_{\Sigma}); r'_{\mathsf{Com}})$.
2. Output $(c = c') \wedge \Sigma.\mathsf{Verify}(vk_{\Sigma}, \sigma'_{\Sigma}, (vk, \sigma_{\mathsf{RS}}))$.

If $\mathsf{RS}$ is a *repudiable* ring signature scheme equipped with algorithms $(\mathsf{Repudiate}, \mathsf{VerRepud})$, then we additionally define $\mathsf{C\text{-}RS.Repudiate}$ and $\mathsf{C\text{-}RS.VerRepud}$ to simply run the $\mathsf{Repudiate}$ and $\mathsf{VerRepud}$ algorithms of $\mathsf{RS}$, as follows.

$\mathsf{C\text{-}RS.Repudiate}(R, sk, \sigma = (\sigma_{\mathsf{RS}}, c))$

1. Output $\mathsf{RS.Repudiate}(\mathsf{RS}(R), sk, \sigma_{\mathsf{RS}})$.

$\mathsf{C\text{-}RS.VerRepud}(R, vk, \sigma = (\sigma_{\mathsf{RS}}, c), \xi)$

1. Output $\mathsf{RS.VerRepud}(\mathsf{RS}(R), sk, \sigma_{\mathsf{RS}}, \xi)$.

**Theorem 5.5.** $\mathsf{C\text{-}RS}$ *is a claimable ring signature scheme (Theorem 5.6). Moreover, if* $\mathsf{RS}$ *is a repudiable ring signature scheme, then* $\mathsf{C\text{-}RS}$ *is repudiable-and-claimable (Theorem 5.11).*

*Proof.* Follows from Theorems 5.6 and 5.11. □

**Theorem 5.6** (Claimability of $\mathsf{C\text{-}RS}$)**.** $\mathsf{C\text{-}RS}$ *is a claimable ring signature scheme.*

*Proof.* Follows from Lemmata 5.7–5.10, which establish the properties of correctness, claimability, unforgeability, and anonymity, respectively. □

**Lemma 5.7** (Correctness of $\mathsf{C\text{-}RS}$)**.** $\mathsf{C\text{-}RS}$ *satisfies correctness (Definition 2.2).*

Correctness is immediate, so we omit the proof.

**Lemma 5.8** (Claimability of $\mathsf{C\text{-}RS}$)**.** $\mathsf{C\text{-}RS}$ *is claimable (Definition 3.9).*

*Proof.* We show that $\mathsf{C\text{-}RS}$ satisfies each of the three conditions of Definition 3.9. The first condition is immediate by the correctness of the signature scheme $\Sigma$, since the use of the $\mathsf{PRF}$ ensures that the values $(r'_\Sigma, r'_{\mathsf{Com}}, \sigma'_\Sigma)$ computed in $\mathsf{Claim}$ are the same as the corresponding values computed in $\mathsf{Sign}$ and that the commitments $c, c'$ match.

For the second condition, assume for contradiction that there exists some PPT malicious claiming algorithm $\mathcal{A}_{\mathsf{Claim}} = (\mathcal{A}_1, \mathcal{A}_2)$ that is able to claim a signature produced by a different party. That is, that the following probability is non-negligible:

$$\Pr\left[\begin{array}{l} (vk, sk) \leftarrow \mathsf{C\text{-}RS.Gen}(1^k) \\ (R', m) \leftarrow \mathcal{A}_1^{\mathcal{O}}(vk) \\ \sigma \leftarrow \mathsf{C\text{-}RS.Sign}(R' \cup \{vk\}, sk, m) \\ (\zeta, vk') \leftarrow \mathcal{A}_2^{\mathcal{O}}(R' \cup \{vk\}, \sigma) \\ b \leftarrow \mathsf{C\text{-}RS.VerClaim}(R' \cup \{vk\}, vk', \sigma, \zeta) \\ b' \leftarrow \mathsf{C\text{-}RS.Verify}(R' \cup \{vk\}, \sigma, m) \end{array} : \begin{array}{l} b = 1 \wedge b' = 1 \\ \wedge vk' \neq vk \end{array}\right], \tag{24}$$

where $\mathcal{O} = \{\mathsf{OSign}, \mathsf{OClaim}_{vk,sk}\}$. We will produce an adversary $\mathcal{B}$ that breaks the binding property of the commitment scheme. Let $\mathcal{B}$ first run the experiment in Equation 24, invoking the malicious claiming algorithm $\mathcal{A}_{\mathsf{Claim}}$ and using its knowledge of the secret key $sk$ to answer the oracle queries of $\mathcal{A}_{\mathsf{Claim}}$. Let $\mathcal{B}$ then compute $\zeta' = \mathsf{C\text{-}RS.Claim}(R' \cup \{vk\}, sk, \sigma)$ and $b'' = \mathsf{C\text{-}RS.VerClaim}(R' \cup \{vk\}, vk, \sigma, \zeta')$. Since the signature verifies correctly with non-negligible probability, the check on line 7 of $\mathsf{C\text{-}RS.Sign}$ must pass, and so we must have that $b'' = 1$ whenever $b' = 1$. Conseqently with non-negligible probability we have that $b = b' = b'' = 1$ and $vk \neq vk'$. Conditioning on

this event, we have that $\sigma = (\sigma_{\mathsf{RS}}, c)$, $\zeta = (r_{\mathsf{Com}}, \sigma_\Sigma)$, and $\zeta' = (r'_{\mathsf{Com}}, \sigma'_\Sigma)$ are such that $c = \mathsf{Com}((vk, \sigma_\Sigma), r_{\mathsf{Com}}) = \mathsf{Com}((vk', \sigma'_\Sigma), r'_{\mathsf{Com}})$. But $vk \neq vk'$, so with non-negligible probability $\mathcal{B}$ generates two different openings to the same commitment, breaking the binding property of the commitment. This concludes the proof of the second property of Definition 3.9.

For the third condition, assume for contradiction that there exists some PPT malicious signing-and-claiming algorithm $\mathcal{A}_{\mathsf{S\&C}}$ that produces a signature and claims it on behalf of a different party. That is, assume that the following probability is non-negligible:

$$\Pr\left[\begin{array}{ll} (vk, sk) \leftarrow \mathsf{C\text{-}RS.Gen}(1^k) & \\ (R', m, \sigma, \zeta) \leftarrow \mathcal{A}_{\mathsf{S\&C}}^{\mathcal{O}, \mathsf{OClaim}_{(vk,sk)}}(vk) & : \quad b = 1 \wedge b' = 1 \\ b \leftarrow \mathsf{C\text{-}RS.VerClaim}(R' \cup \{vk\}, vk, \sigma, \zeta) & \quad \wedge Q \cap \{(\cdot, \sigma)\} = \varnothing \\ b' \leftarrow \mathsf{C\text{-}RS.Verify}(R' \cup \{vk\}, \sigma, m) & \end{array}\right], \tag{25}$$

where $\mathcal{O} = \{\mathsf{OSign}\}$ and $Q$ is the set of queries made to the oracle $\mathsf{OClaim}_{(vk,sk)}$. We will construct an adversary $\mathcal{B}$ that breaks the unforgeability property of the signature scheme $\Sigma$. $\mathcal{B}$ first requests a verification key $vk^*_\Sigma$ from the challenger for $\Sigma$. It samples keys $(vk, sk) \leftarrow \mathsf{C\text{-}RS.Gen}(1^k)$ as in the beginning of the experiment in Equation 25, but replaces $vk_\Sigma$ with $vk^*_\Sigma$ and $sk_\Sigma$ with $\perp$ in $vk$ and $sk$, respectively. It then invokes the adversary $\mathcal{A}_{\mathsf{S\&C}}$ on inputs $(1^k, vk)$ to produce values $(R', m, \sigma, \zeta)$, using the challenger for the signature scheme $\Sigma$ to respond to the oracle queries of $\mathcal{A}_{\mathsf{S\&C}}$ as follows:

- When $\mathcal{A}_{\mathsf{S\&C}}$ queries oracle $\mathsf{OSign}$ on input $(m, R)$, algorithm $\mathcal{B}$ first computes $\sigma_{\mathsf{RS}}$ as in $\mathsf{C\text{-}RS.Sign}$. It then invokes the challenger for $\Sigma$ on message $(vk, \sigma_{\mathsf{RS}})$, receiving signature $\sigma_\Sigma$. (If the challenger has already been queried on this message $(vk, \sigma_{\mathsf{RS}})$ in a previous invocation of $\mathsf{OSign}$, then instead of querying the challenger again, use the same value $\sigma_\Sigma$ sent by the challenger in the previous invocation.) It then proceeds as in steps 4–8 of algorithm $\mathsf{C\text{-}RS.Sign}$ computing value $r_{\mathsf{Com}}$ and commitment $c$, testing whether $\mathsf{C\text{-}RS.VerClaim}$ succeeds (where in the inner invocation of $\mathsf{C\text{-}RS.Claim}$ we set $\sigma'_\Sigma$ to be the challenger-produced signature $\sigma_\Sigma$ instead of running $\Sigma.\mathsf{Sign}$), and returning $\sigma = (\sigma_{\mathsf{RS}}, c)$.
- When $\mathcal{A}_{\mathsf{S\&C}}$ queries oracle $\mathsf{OClaim}_{(vk,sk)}$ on input $(R, \sigma)$, algorithm $\mathcal{B}$ first tests whether $\sigma$ was an output returned by a previous invocation of the oracle $\mathsf{OSign}$. If not, it immediately returns $\perp$. If it is, then let $\sigma_\Sigma$ be the signature returned by the challenger for $\Sigma$ on that invocation of $\mathsf{OSign}$. Compute value $r_{\mathsf{Com}}$ as in algorithm $\mathsf{C\text{-}RS.Claim}$, and output $\zeta = (r_{\mathsf{Com}}, \sigma_\Sigma)$.

Finally, algorithm $\mathcal{B}$ parses $\zeta = (r'_{\mathsf{Com}}, \sigma'_\Sigma)$ and outputs $\sigma'_\Sigma$ as its forgery.

We now outline a hybrid argument to show that this adversary $\mathcal{B}$ breaks the unforgeability property of the signature scheme.

**Hybrid 1.** The experiment with adversary $\mathcal{B}$ as just described.

**Hybrid 2.** Instead of the signature scheme challenger using actual randomness to generate the signatures in the oracle queries to $\mathsf{OSign}$, use pseudorandomness obtained by a PRF invocation $r_\Sigma = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 0))$

**Hybrid 3.** Instead of responding to both types of oracle queries as above, use actual invocations of $\mathsf{C\text{-}RS.Sign}$ and $\mathsf{C\text{-}RS.Claim}$, using the secret key $sk_\Sigma$ known to the challenger for the unforgeability game.

Indistinguishability of Hybrids 1 and 2 follows from the pseudorandomness of the PRF. For Hybrids 2 and 3, note first that the two experiments behave identically on invocations of the oracle

$\mathsf{OSign}$. It remains to consider invocations of the oracle $\mathsf{OClaim}$. By the binding property of the commitment scheme and the pseudorandomness of the PRF, with all but negligible probability adversary $\mathcal{A}_{\mathsf{S\&C}}$ in Hybrid 3 will be unable to find an input to the oracle $\mathsf{OClaim}$ that does not yield output $\perp$, except for inputs that were previously produced as output to the oracle $\mathsf{OSign}$. Consequently, except with negligible probability, the oracle $\mathsf{OClaim}$ will behave identically in Hybrids 2 and 3. But Hybrid 3 is exactly the experiment in Equation 25, so with non-negligible probability adversary $\mathcal{A}_{\mathsf{S\&C}}$ in this experiment produces a signature $\sigma = (\sigma_{\mathsf{RS}}, c)$ and claim $\zeta = (r'_{\mathsf{Com}}, \sigma'_\Sigma)$ such that $\sigma$ was not a query to oracle $\mathsf{OClaim}_{(vk,sk)}$ and $\Sigma.\mathsf{Verify}(vk_\Sigma, \sigma'_\Sigma, (vk, \sigma_{\mathsf{RS}}))$ outputs 1, i.e. $\sigma'_\Sigma$ verifies as a valid signature of the message $(vk, \sigma_{\mathsf{RS}})$ under key $vk_\Sigma$. By the hybrid argument, it follows that with non-negligible probability, $\mathcal{B}$ successfully produces a valid signature for message $(vk, \sigma_{\mathsf{RS}})$.

It remains to argue that this signature is a valid forgery, i.e. that its message is distinct from each of the signatures produced by the challenger. To achieve this, we will make a small modification to the adversary $\mathcal{B}$; call the new adversary $\mathcal{B}'$. Let $L = L(k)$ be a (polynomial) upper bound on the number of queries made by $\mathcal{B}$ to the oracle $\mathsf{OSign}$, and choose at random an index $i^* \in [L]$ On the $i^*$th query to $\mathsf{OSign}$, rather than invoking the ring signature challenger to obtain $\sigma_\Sigma$ and computing commitment $c$, instead choose a random string $\sigma^*$ and compute the commitment with respect to that. By the hiding property of the commitment scheme, as long as the oracle $\mathsf{OClaim}$ is not queried on this signature, the output of the modified adversary $\mathcal{B}'$ is indistinguishable from that of $\mathcal{B}$. We already have by assumption that with non-negligible probability $\mathcal{A}_{\mathsf{S\&C}}$ wins the experiment in Equation 25, but moreover we have with non-negligible probability that in addition either the signature $\sigma$ produced by $\mathcal{A}_{\mathsf{S\&C}}$ was never the output of oracle $\mathsf{OSign}$ (and that $i^*$ is greater than the number of queries made to oracle $\mathsf{OSign}$) or that it was the output of the $i^*$th query to $\mathsf{OSign}$ (and no earlier query). In the latter event, since $\mathcal{A}_{\mathsf{S\&C}}$ wins the experiment in Equation 25, this signature cannot have been a query to oracle $\mathsf{OClaim}$, and so in either case, the behavior of $\mathcal{B}'$ is indistinguishable from $\mathcal{B}$. But in either case, the signature $\sigma$ in the experiment with $\mathcal{B}'$ was not produced by invoking the challenger to the signature scheme, so it follows that (except with negligible probability), the signature $\sigma'_\Sigma$ from the claim $\zeta$ was not produced by the signature scheme challenger. Putting everything together, we have that with non-negligible probability, the adversary $\mathcal{A}_{\mathsf{S\&C}}$ in the experiment with $\mathcal{B}'$ produces a valid signature for a message distinct from any message signed by the signature scheme challenger. This violates the unforgeability property of the signature scheme and yields a contradiction, and so the third property of Definition 3.9 is also satisfied. □

**Lemma 5.9** (Unforgeability of $\mathsf{C\text{-}RS}$)**.** $\mathsf{C\text{-}RS}$ *is unforgeable (in the sense of Definition 3.11).*

*Proof.* The proof is by reduction to the unforgeability of $\mathsf{RS}$.[18] Suppose, for contradiction, that there is a PPT adversary $\mathcal{A}$ that violates unforgeability of $\mathsf{C\text{-}RS}$ (Definition 3.11). Then we construct another adversary $\mathcal{B}$ that violates unforgeability of $\mathsf{RS}$ *without having access to a* $\mathsf{OClaim}$ *oracle.* On input $(vk_1, \ldots, vk_N)$ which are verification keys of $\mathsf{RS}$, $\mathcal{B}$ behaves as follows.

1. For each $i \in [N]$:
   - Sample $(vk^i_\Sigma, sk^i_\Sigma) \leftarrow \Sigma.\mathsf{Gen}(1^k)$.

[18] Note that the unforgeability guarantee we have on $\mathsf{RS}$ is standard unforgeability of ring signatures (Definition 2.8), which does not give the adversary a $\mathsf{OClaim}$ oracle. If we had the stronger guarantee that $\mathsf{RS}$ were unforgeable in the presence of a $\mathsf{OClaim}$ oracle, then the unforgeability of $\mathsf{C\text{-}RS}$ would follow immediately, since signatures of $\mathsf{C\text{-}RS}$ contain signatures of $\mathsf{RS}$.

- Sample $sk^i_{\mathsf{PRF}} \leftarrow \mathsf{PRF.Gen}(1^k)$.
- Let $vk^*_i = (vk_i, vk^i_\Sigma)$ and $sk^*_i = (sk_i, sk^i_\Sigma, sk^i_{\mathsf{PRF}})$.

2. Run $\mathcal{A}$ on input $(vk^*_1, \ldots, vk^*_N)$, answering $\mathcal{A}$'s oracle queries as follows.
   (a) For each query $(i, m, R)$ to C-RS.OSign:
   - Query RS.OSign on $(i, m, R)$ and receive response $\sigma_{\mathsf{RS}}$.
   - Let $r_\Sigma = \mathsf{PRF.Eval}(sk^i_{\mathsf{PRF}}, (vk^*_i, \sigma_{\mathsf{RS}}, 0))$.
   - Let $\sigma_\Sigma = \Sigma.\mathsf{Sign}(sk^i_\Sigma, (vk^*_i, \sigma_{\mathsf{RS}}); r_\Sigma)$.
   - Let $r_{\mathsf{Com}} = \mathsf{PRF.Eval}(sk^i_{\mathsf{PRF}}, (vk^*_i, \sigma_{\mathsf{RS}}, 1))$.
   - Let $c = \mathsf{Com}(\sigma_\Sigma; r_{\mathsf{Com}})$.
   - Output $\sigma = (\sigma_{\mathsf{RS}}, c)$.

   (b) For each query $(i, R, \sigma)$ to C-RS.OClaim:
   - Parse $\sigma$ as $(\sigma_{\mathsf{RS}}, c)$.
   - Let $r'_\Sigma = \mathsf{PRF.Eval}(sk^i_{\mathsf{PRF}}, (vk^*_i, \sigma_{\mathsf{RS}}, 0))$.
   - Let $r'_{\mathsf{Com}} = \mathsf{PRF.Eval}(sk^i_{\mathsf{PRF}}, (vk^*_i, \sigma_{\mathsf{RS}}, 1))$.
   - Let $\sigma'_\Sigma = \Sigma.\mathsf{Sign}(sk^i_\Sigma, (vk^i_\Sigma, \sigma_{\mathsf{RS}}); r'_\Sigma)$.
   - Output $\zeta = (r'_{\mathsf{Com}}, \sigma'_\Sigma)$.

3. Upon receiving an output $(R', m', \sigma')$ from $\mathcal{A}$: parse $\sigma'$ as $(\sigma'_{\mathsf{RS}}, c')$, define $R'' = \{vk_i : vk^*_i \in R'\}$, and output $(R'', m', \sigma'_{\mathsf{RS}})$.

By construction of C-RS and $\mathcal{B}$, whenever $\mathcal{A}$ successfully forges with respect to C-RS, $\mathcal{B}$ successfully forges with respect to RS. Moreover, $\mathcal{B}$'s responses to $\mathcal{A}$'s oracle queries are, by construction, distributed identically to the oracle responses in the unforgeability experiment for C-RS. Therefore, $\mathcal{A}$'s probability of successful forgery is the same when $\mathcal{B}$ runs $\mathcal{A}$ in the above reduction, as in the unforgeability experiment. By our supposition, $\mathcal{A}$'s forging probability in the unforgeability experiment is some non-negligible $\epsilon$, so it follows that $\mathcal{A}$'s forging probability in the above reduction is also $\epsilon$, and therefore $\mathcal{B}$'s forging probability with respect to RS is in turn $\epsilon$. This contradicts the unforgeability of RS. The lemma follows. □

**Lemma 5.10** (Anonymity of C-RS)**.** *If* RS *satisfies anonymity (resp., adaptive anonymity) against adversarially chosen keys (Definition 2.5), then* C-RS *satisfies anonymity (resp., adaptive anonymity) against adversarially chosen keys (Definition 3.10).*

*Proof.* We give the proof that C-RS satisfies *adaptive* anonymity whenever RS satisfies adaptive anonymity. The non-adaptive version of the statement has a slightly simpler proof: the proof follows the same structure, but certain steps of the proof become unnecessary. In the rest of the proof, we write "anonymity" to mean "adaptive anonymity against adversarially chosen keys."

We begin with a hybrid argument. Recall the anonymity experiment (Definition 2.3) for anonymity against adversarially chosen keys (Definition 3.10), shown below as "Hybrid 0."

**Anonymity experiment (Hybrid 0)**

$(vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k)$

$((m^*, i_0^*, i_1^*, R^*), \mathfrak{s}) \leftarrow \mathcal{A}_1^{\mathsf{OSign},\mathsf{OClaim},\mathsf{Corr}}(vk_1, \ldots, vk_N)$

$b \leftarrow \{0,1\}$

$\sigma \leftarrow \mathsf{Sign}(R^* \cup \{vk_{i_0^*}, vk_{i_1^*}\}, sk_{i_b^*}, m^*)$

$b' \leftarrow \mathcal{A}_2^{\mathsf{OSign},\mathsf{OClaim}^{\langle\sigma\rangle},\mathsf{Corr}}(\mathfrak{s}, \sigma)$

We now define two signing algorithms $\mathsf{Sign}_1$ and $\mathsf{Sign}_2$ which are slight variants of C-RS.Sign. For $\iota \in \{1,2\}$, we define Hybrid $\iota$ to be the same as Hybrid 0 except that the invocation of $\mathsf{Sign}$ in the fourth line of the experiment is replaced by an invocation of $\mathsf{Sign}_\iota$. In the descriptions of $\mathsf{Sign}_1$ and $\mathsf{Sign}_2$ below, changes from the preceding hybrid are marked in blue, and steps which are entirely removed are "crossed out" and shown in red.

$\mathsf{Sign}_1(R, sk, m)$

1. Let $\sigma_{\mathsf{RS}} \leftarrow \mathsf{RS.Sign}(\mathsf{RS}(R), sk_{\mathsf{RS}}, m)$.
2. Let $r_\Sigma = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 0))$.
3. Let $\sigma_\Sigma = \Sigma.\mathsf{Sign}(sk_\Sigma, (vk, \sigma_{\mathsf{RS}}); r_\Sigma)$.
4. Let $r_{\mathsf{Com}} \leftarrow \{0,1\}^\nu$.
5. Let $c = \mathsf{Com}(\sigma_\Sigma; r_{\mathsf{Com}})$.
6. Output $\sigma = (\sigma_{\mathsf{RS}}, c)$.

$\mathsf{Sign}_2(R, sk, m)$

1. Let $\sigma_{\mathsf{RS}} \leftarrow \mathsf{RS.Sign}(\mathsf{RS}(R), sk_{\mathsf{RS}}, m)$.
2. ~~Let $r_\Sigma = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 0))$.~~
3. ~~Let $\sigma_\Sigma = \Sigma.\mathsf{Sign}(sk_\Sigma, (vk, \sigma_{\mathsf{RS}}); r_\Sigma)$.~~
4. Let $r_{\mathsf{Com}} \leftarrow \{0,1\}^\nu$.
5. Let $c = \mathsf{Com}(0; r_{\mathsf{Com}})$.
6. Output $\sigma = (\sigma_{\mathsf{RS}}, c)$.

**Hybrid 1 is indistinguishable from Hybrid 0.** This follows from PRF security as long as there are no other variables in $\mathcal{A}$'s view that are correlated with the PRF output in Hybrid 0, namely, $r_{\mathsf{Com}} = \mathsf{PRF.Eval}(sk_{\mathsf{PRF}}, (vk, \sigma_{\mathsf{RS}}, 1))$. Since PRF security guarantees that PRF outputs on different inputs are computationally indistinguishable from uniform and independent strings, it suffices to establish that nowhere else in the anonymity experiment is the PRF evaluated on the specific input $(vk, \sigma_{\mathsf{RS}}, 1)$. The only PRF evaluations during the experiment are to compute the $r_\Sigma$ and $r_{\mathsf{Com}}$ values used by the $\mathsf{OSign}$ oracle when responding to oracle queries. The PRF inputs used to compute $r_\Sigma$ values are distinct, by construction, by from those used to compute $r_{\mathsf{Com}}$ values (since the former end in 0 and the latter end in 1). The PRF inputs used to compute $r_{\mathsf{Com}}$ values for $\mathsf{OSign}$ queries are also, with overwhelming probability, distinct from the challenge input $(vk, \sigma_{\mathsf{RS}}, 1)$. Since each such PRF input is of the format $(vk', \sigma'_{\mathsf{RS}}, 1)$ where $\sigma'_{\mathsf{RS}}$ is an honestly generated signature under $\mathsf{RS}$, this follows from the following two observations.

1. The anonymity of $\mathsf{RS}$ implies that multiple honestly generated signatures under the *same* key pair must be *distinct* with overwhelming probability. (Otherwise, the adversary could break anonymity by querying a signature under every key in the ring on many messages, and then checking the challenge signature for equality with any of the preceding signatures.)
2. The unforgeability of $\mathsf{RS}$ implies that honestly generated signatures under an honestly generated key pair $(vk, sk)$ are *distinct* from honestly generated signatures under any other — possibly adversarially generated[19] — key pair.

[19] With knowledge of $vk$ but not $sk$.

**Hybrid 2 is indistinguishable from Hybrid 1.** This follows from the hiding property of the commitment, since all that has changed from Hybrid 1 is the value committed to by $c$.

The rest of the proof gives a reduction between Hybrid 2 and the anonymity of $\mathsf{RS}$. Note that the anonymity guarantee we have on $\mathsf{RS}$ is anonymity of standard ring signatures (Definition 2.8), which does not give the adversary a $\mathsf{OClaim}$ oracle.

Suppose, for contradiction, that there is a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that violates anonymity of $\mathsf{C\text{-}RS}$ (Definition 3.10). Then we construct another adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ that violates anonymity of $\mathsf{RS}$ *without having access to a* $\mathsf{OClaim}$ *oracle*. On input $(vk_1, \ldots, vk_N)$ which are verification keys of $\mathsf{RS}$, $\mathcal{B}_1$ behaves as follows.

1. For each $i \in [N]$, construct $vk_i^* = (vk_i, vk_\Sigma^i)$ and $sk_i^* = (sk_i, sk_\Sigma^i, sk_{\mathsf{PRF}}^i)$ exactly as described in Step 1 in the proof of Lemma 5.9.
2. Run $\mathcal{A}$ on input $(vk_1^*, \ldots, vk_N^*)$, answering $\mathcal{A}$'s oracle queries exactly as described in Step 2 in the proof of Lemma 5.9.
3. Upon receiving an output $((m', i_0', i_1', R'), \mathfrak{s}')$ from $\mathcal{A}$: define $R'' = \{vk_i : vk_i^* \in R'\}$, let $\mathfrak{s}''$ be all of $\mathcal{B}$'s internal state, and output $((m', i_0', i_1', R''), \mathfrak{s}'')$.

Then, on input $(\mathfrak{s}'', \sigma_{\mathsf{RS}})$, $\mathcal{B}_2$ behaves as follows.

1. Let $c = \mathsf{Com}(0; r)$ for truly random $r$.
2. Let $\sigma = (\sigma_{\mathsf{RS}}, c)$.
3. Run $\mathcal{A}_2$ to obtain $b' \leftarrow \mathcal{A}_2(\mathfrak{s}', \sigma)$.
4. Output $b'$.

$\mathcal{A}$'s view between the reduction run by $\mathcal{B}$ is identically distributed to $\mathcal{A}$'s view in the experiment of Hybrid 2. Moreover, by construction, $\mathcal{B}_2$'s guess is correct exactly when $\mathcal{A}_2$ guesses $b'$ correctly. Therefore, $\mathcal{B}$'s success probability in the anonymity experiment of $\mathsf{RS}$ is negligibly close to $\mathcal{A}$'s success probability in the anonymity experiment of $\mathsf{C\text{-}RS}$. By supposition, the latter probability is non-negligibly greater than $1/2$. It follows that $\mathcal{B}$ violates the anonymity of $\mathsf{RS}$, which is contradiction. The lemma follows. □

**Theorem 5.11** (Repudiability-and-claimability of $\mathsf{C\text{-}RS}$)**.** *If* $\mathsf{RS}$ *is repudiable, then* $\mathsf{C\text{-}RS}$ *is a repudiable-and-claimable ring signature scheme. (Definition A.1).*

*sketch.* Suppose $\mathsf{RS}$ is repudiable. We need to prove that $\mathsf{C\text{-}RS}$ satisfies the definitions (which are given in Appendix A) of repudiability, claimability, anonymity, and unforgeability of repudiable-and-claimable ring signature schemes.

The proofs of anonymity and unforgeability are essentially identical to the proofs of Lemmata 5.10 and 5.9, respectively. Those proofs reduce the anonymity/unforgeability of $\mathsf{C\text{-}RS}$ to the anonymity/unforgeability of $\mathsf{RS}$ (as defined in Section 2), respectively, under the assumption that $\mathsf{RS}$ is a *standard* (i.e., not necessarily repudiable) ring signature scheme. The same proof structure suffices to argue that in the case that $\mathsf{RS}$ is a repudiable ring signature scheme, that the anonymity/unforgeability of $\mathsf{C\text{-}RS}$ reduces to the anonymity/unforgeability notions for *repudiable* ring signature schemes (as defined in Section 3.1), which by assumption are satisfed by $\mathsf{RS}$.

It remains to prove repudiability and claimability of $\mathsf{C\text{-}RS}$, according to Definitions A.2 and A.3, respectively.

**Repudiability** We reduce the repudiability of C-RS to the repudiability of RS. Suppose, for contradiction, that C-RS did not satisfy Definition A.2. Then the repudiability of RS could be violated by an adversary $\mathcal{A}$ that instantiates its own commitment and signature schemes and PRF, and runs the adversary $\mathcal{B}$ that breaks the repudiability of C-RS, while:

- augmenting each signature under RS with a corresponding commitment so that it appears indistinguishable from a signature under C-RS in $\mathcal{B}$'s view; and
- remembering the commitment randomness for any such commitments; and
- answering oracle calls to OClaim by producing the appropriate decommitments (using the remembered commitment randomness); and
- answering oracle calls to ORpd by passing them to its own ORpd oracle for RS.

**Claimability** The proof structure for claimability is very similar to that of Lemma 5.8, which argues that C-RS satisfies each of the three conditions of Definition 3.9 in turn.

- The argument for the first condition (i.e., that honest claims are validated by VerClaim) goes through unchanged.
- The argument for the second condition (i.e., that non-signers cannot successfully claim) also goes through unchanged: the adversary $\mathcal{B}$ constructed in the proof of Lemma 5.8 has knowledge of the secret key $sk$, which it can use to answer ORpd queries using the honest repudiation algorithm.
- The argument for the third condition needs to be augmented with a description of how $\mathcal{B}$ responds to oracle queries to ORpd that are made by $\mathcal{A}$. Just as above, $\mathcal{B}$ answers these oracle queries by running the honest repudiation algorithm. □

# 6 Unclaimable construction

In this section we show how to construct unclaimable ring signatures from lattice assumptions. The scheme is exactly the SIS-based ring signature scheme of Brakerski and Kalai [BK10], augmented with an additional algorithm ExtractRandomness.

We first give a very brief summary of necessary background on lattice trapdoors; see [GPV08] and Appendix D for details.

## 6.1 Lattice trapdoor sampling

Let $q \in \mathbb{N}$, $m' \in \mathbb{N}$, and $\beta \in \mathbb{Z}$ be functions of security parameter $n$. The (inhomogeneous, average-case) *short integer solution* ($\mathsf{SIS}_{q,m,\beta}$) assumption states that given $A \leftarrow \mathbb{Z}_q^{n \times m'}$, $v \leftarrow \mathbb{Z}_q^n$, it is computationally hard to find $x \in \mathbb{Z}_q^{m'}$ such that $Ax = v$ and $\|x\| \leq \beta$. For polynomial $m', \beta$ and prime $q \geq \beta \cdot \omega(\sqrt{n \log n})$, the SIS problem is known to be as hard as approximating worst-case lattice problems, in particular the Shortest Independent Vectors Problem (SIVP), to within a factor of $\beta \cdot \tilde{O}(\sqrt{n})$ [MR07; GPV08].

Let $D_{\Lambda,s,c}$ denote the discrete Gaussian distribution over $n$-dimensional lattice $\Lambda$, centered at $c \in \mathbb{R}^n$ and with parameter $s$. We note the existence of the following algorithms, described in [GPV08][20]:

[20] These are given by the algorithms TrapGen, SampleD and SampleISIS in [GPV08].

- There is an algorithm $\mathsf{TrapdoorSamp}$ that on input a security parameter $1^n$ produces a matrix $A \in \mathbb{Z}_q^n$ and a trapdoor $T$, where $A$ is statistically close to uniform and $T$ is a short basis for the lattice $\Lambda^\perp(A)$.
- There is an algorithm $\mathsf{SampleDist}$ that samples from the discrete Gaussian distribution $D_{\mathbb{Z}^{m'},s,0}$.
- There is an algorithm $\mathsf{SampleCond}$ that on input a matrix $A$, trapdoor $T$, parameter $s$ and vector $u$, produces a sample $x$ distributed statistically close to the discrete Gaussian distribution $D_{\mathbb{Z}^{m'},s,0}$ conditioned on $Ax = u$. We have that $\|x\|_2 \leq s\sqrt{n}$ with probability 1.

We will also require additional algorithms that given output values of the algorithms $\mathsf{SampleDist}$ and $\mathsf{SampleCond}$, respectively, sample randomness under which the algorithm produces the desired output.

- There is an algorithm $\mathsf{ExplainDist}$ that on input an image vector $x$ and parameter $s$, samples randomness $\rho$ that yields output $x$ under algorithm $\mathsf{SampleDist}$, i.e. samples from the distribution $\{\rho|\mathsf{SampleDist}(s;\rho) = x\}$.
- There is an algorithm $\mathsf{ExplainCond}$ that on input matrix $A$, trapdoor $T$, parameter $s$, vector $u$ and image vector $x$, samples randomness $\rho$ that yields output $x$ under algorithm $\mathsf{SampleCond}$ with inputs $(A, T, s, u)$, i.e. samples from the distribution $\{\rho|\mathsf{SampleCond}(A, T, s, u; \rho) = x\}$.

We describe the algorithms $\mathsf{ExplainDist}$ and $\mathsf{ExplainCond}$ in Appendix D. We will use a slight modification of the $\mathsf{SampleCond}$ algorithm of [GPV08] that uses the basis randomization technique of [CHKP10]. We need the following lemma.

**Lemma 6.1.** *Let $(A_1, T_1)$ and $(A_2, T_2)$ be sampled from $\mathsf{TrapdoorSamp}$, let $y \in \mathbb{Z}_q^n$, and let $s \geq \max(\|\tilde{T}_1\|, \|\tilde{T}_2\|) \cdot \omega(\sqrt{\log n})$, where the tilde denotes Gram-Schmidt orthogonalization. Sample vectors $x_1$ and $x_2'$ from $\mathsf{SampleDist}$. Let $x_2 \leftarrow \mathsf{SampleCond}(A_2, T_2, s, y - A_1 x_1)$, and let $x_1' \leftarrow \mathsf{SampleCond}(A_1, T_1, s, y - A_2 x_2')$. Then the distributions $(A_1, T_1, A_2, T_2, x_1, x_2)$ and $(A_1, T_1, A_2, T_2, x_1', x_2')$ are statistically close.*

Intuitively, this lemma says that the sampled vectors are distributed the same independently of which of the two trapdoors was used. This follows immediately from Lemma 3.3 of [CHKP10].

## 6.2 The basic construction of [BK10]

We now describe the construction of [BK10].[21] Brakerski and Kalai first construct a base version of their scheme that satisfies a weaker security notion, and augment it to fully secure ring signatures in a series of steps.

Let the message space be $\{0,1\}^\ell$, and let $X = \{x \in \mathbb{Z}_q^{m'} : \|x\|_2 \leq s\sqrt{m'}\}$ for some $s = \omega(\sqrt{n \log n \log q})$ be the set of "short" vectors.

The key generation algorithm samples a matrix with an $\mathsf{SIS}$ trapdoor, and an additional set of $2\ell$ matrices, two corresponding to each bit of the message. It additionally samples a target vector $y$, and outputs the matrices and target vector as the verification key and the trapdoor as the signing key.

$\mathsf{BK\text{-}RS.Gen}(1^k)$

[21] The original presentation of [BK10] introduces an abstraction they call *ring trapdoor functions*, and instantiates this abstraction from biliear group assumptions as well as the $\mathsf{SIS}$ assumption. We present their $\mathsf{SIS}$-based construction more explicitly, without the additional layer of abstraction, in order to make the role of the $\mathsf{SIS}$ trapdoor more apparent.

1. Let $(A, T) \leftarrow \mathsf{TrapdoorSamp}(1^k)$.
2. For $(i, b) \in [\ell] \times \{0, 1\}$, let $A_{i,b} \leftarrow \mathbb{Z}_q^{n \times m'}$.
3. Let $y \leftarrow \mathbb{Z}_q^n$.
4. Output $vk = (A, (A_{j,b})_{(j,b)\in[\ell]\times\{0,1\}}, y)$ and $sk = (vk, T)$.

The signing algorithm proceeds as follows. A target vector $y$ is selected from the lexicographically first verification key. For each identity in the ring, short vectors are sampled for matrices corresponding to each bit of the message to be signed, as well as the additional matrix. Finally, the trapdoor in the signing key is used to obtain a short vector, which is sampled from the same distribution conditioned on having a particular product with the matrix $A_{i^*}$ corresponding to the signer, i.e., conditioned on Equation 26 being satisfied. The signature consists of the list of short vectors for each identity and each index of the message.

$\mathsf{BK\text{-}RS.Sign}(R, sk, m; \rho)$

1. Parse $R = (vk_1, \ldots, vk_N)$ and $sk = (vk, T)$.
2. For $i \in [N]$, parse $vk_i = (A_i, (A_{j,b}^{(i)})_{(j,b)\in[\ell\times\{0,1\}}, y_i)$.
3. Let $y = y_i$, where $i \in [N]$ is the index for which $vk_i$ is lexicographically first.
4. If $vk \notin R$, output $\perp$ and halt.
5. Define $i^* \in [N]$ be such that $vk_{i^*} = vk$.
6. Using the trapdoor $T_A$ for $A_{i^*}$, we can sample $(x_j^{(i)})_{i\in[N],j\in\{0\}\cup[\ell]}$ such that
$$\sum_{i\in[N]} A_i x_0^{(i)} + \sum_{\substack{i\in[N]\\ j\in[\ell]}} A_{j,m_j}^{(i)} x_j^{(i)} = y. \tag{26}$$
That is, for $(i, j) \in [N]\times\{0\}\cup[\ell]$ other than the pair $(i^*, 0)$, we invoke algorithm $\mathsf{SampleDist}$ to sample $x_j^{(i)} \in$ independently from the discrete Gaussian distribution $\mathcal{X}$. Finally, we invoke algorithm $\mathsf{SampleCond}$ use the trapdoor $T$ for $A_{i^*}$ to sample $x_0^{(i^*)}$ from a distribution statistically close to the distribution $\mathcal{X}$ conditioned on Equation 26 being satisfied.
7. Output $\sigma = (x_j^{(i)})_{i\in[N],j\in\{0\}\cup[\ell]}$.

The verification procedure simply checks that each vector in the signature has short entries and that Equation 26 is satisfied.

$\mathsf{BK\text{-}RS.Verify}(R, \sigma, m)$

1. Parse $R = (vk_1, \ldots, vk_N)$.
2. For $i \in [N]$, parse $vk_i = (A_i, (A_{j,b}^{(i)})_{(j,b)\in[\ell\times\{0,1\}}, y_i)$.
3. Parse $\sigma = (x_j^{(i)})_{i\in[N],j\in\{0\}\cup[\ell]}$.
4. For each $x_j^{(i)}$ for $i \in [N], j \in \{0\} \cup [\ell]$, if $x_j^{(i)} \notin X$ then immediately reject.
5. Let $y = y_i$, where $i \in [N]$ is the index for which $A_{i^*}$ is lexicographically first.
6. Accept if Equation 26 above is satisfied, and otherwise reject.

Thus far we have simply described the basic ring signature scheme of [BK10]. We augment this scheme by providing an $\mathsf{ExtractRandomness}$ algorithm. In order to do so, we must produce "explaining randomness" that maps to the desired output vector under the algorithms $\mathsf{SampleDist}$ and $\mathsf{SampleCond}$. We do this using algorithms $\mathsf{ExplainDist}$ and $\mathsf{ExplainCond}$, as described in Appendix D.

BK-RS.ExtractRandomness$(R, sk, \sigma, m)$

1. Parse $R = (vk_1, \ldots, vk_N)$ and $sk = (vk, T)$.
2. For $i \in [N]$, parse $vk_i = (A_i, (A_{j,b}^{(i)})_{(j,b)\in[\ell]\times\{0,1\}}, y_i)$.
3. Parse $\sigma = (x_j^{(i)})_{i\in[N], j\in\{0\}\cup[\ell]}$.
4. If $vk \notin R$, output $\perp$ and halt.
5. Define $i^* \in [N]$ be such that $vk_{i^*} = vk$.
6. For $(i, j) \in [N] \times \{0\} \cup [\ell]$ other than the pair $(i^*, 0)$, invoke algorithm ExplainDist to sample random coins $\rho_j^{(i)}$ that produce output $x_j^{(i)}$ under the discrete Gaussian sampling algorithm.
7. Invoke algorithm ExplainCond to sample random coins $\rho_0^{(i^*)}$ that produce output $x_0^{(i^*)}$ under the conditional random sampling algorithm using trapdoor $T$.
8. Output $(\rho_j^{(i)})$.

**Theorem 6.2.** *Under the* $\mathsf{SIS}_{q,m',\beta}$ *assumption,* BK-RS *is a unclaimable ring signature scheme satisfying a weak notion of unforgeability in which the challenge is sampled at random at the beginning of the experiment.*

*sketch.* Completeness, anonymity and unforgeability are proven in [BK10]. It remains to show that the scheme is unclaimable. Consider the experiment described in Definition 3.12. The components of the signature corresponding to matrices with no trapdoor are distributed identically on the two sides of the experiment. By the correctness of algorithm ExplainDist, the components of $\rho_1$ and $\rho_2$ corresponding to identities other than $vk_1, vk_2$ are distributed statistically close, jointly with $S$ and the corresponding components of the signature. It remains to consider the portions of the signature corresponding to identities $vk_1$ and $vk_2$. But Lemma 6.1 implies that the distribution of vectors $(x_0^{(1)}, x_0^{(2)})$ is statistically close, regardless of which trapdoor was used to sample. By the correctness of algorithm ExplainCond, the corresponding components of $\rho_1$ and $\rho_2$ are also statistically close, even conditioned on the other values in the experiment. The conclusion follows. □

### 6.3 Unclaimability for the full ring signature scheme of [BK10]

The ring signature scheme just described satisfies a weak notion of unforgeability, in which the message on which a signature must be forged is sampled at random by the challenger and sent to the forger in the beginning of the experiment. To achieve full unforgeability, [BK10] proceed through a sequence of four reductions to construct schemes satsifying successively stronger notions of unforgeability. We now provide a brief overview of these reductions and describe how to modify the ExtractRandomness algorithm for each scheme.

The first modified scheme appends a description of the ring to the message to be signed. This only affects that message, so the ExtractRandomness algorithm is unchanged and is simply invoked on a different message.

The second modification is the most complicated, and introduces a variant of chameleon hash functions. A chameleon hash function $h$ is sampled during Gen and is included as part of the verification key $vk$. During the Sign algorithm, randomness $r$ is sampled from some distribution,[22] and a value $y = h(m, r)$ is computed, where $m$ is the message to be signed and $h$ is the hash function

[22] The distribution over which $r$ is generated is uniform over the set of vectors with bounded $\ell_2$ norm, i.e., $\{r \in \mathbb{Z}_q^{m'} : 0 < \|r\|_2 \leq \beta\}$ for some $m', \beta$.

corresponding to the lexicographically first identity in the ring. The previous signature scheme is invoked on message $y$, and the signature is augmented to include the randomness $r$ as well. Observe that the only randomness to explain is the choice of $r$ and the randomness used in the invocation of the previous signature scheme. Consequently the only modification to ExtractRandomness is that it now must also provide random coins resulting in a particular choice of the vector $r$, which is straightforward.

The third modification simply computes a signature under the previous scheme of every prefix of the message to be signed, and outputs a list of these $|m|$ signatures as its signature. We can invoke the previous ExtractRandomness algorithm for the previous scheme on each of these $|m|$ messages. The final modification produces a random pad $\alpha$ as part of the Gen algorithm, and computes the signature on the exclusive or of the original message with the pad corresponding to the lexicographically first identity in the ring. This is the full ring signature scheme of [BK10]. As above, this only affects the message to be signed, and so the ExtractRandomness algorithm is simply invoked on a different message.

Given the ExtractRandomness algorithm for the weakly-unforgeable ring signature scheme in the previous section, the modifications we have just described yield a ExtractRandomness algorithm for the fully-unforgeable ring signature scheme of [BK10]. It is not difficult to see that this scheme satisfies Definition 3.12 and is an unclaimable ring signature scheme. Consequently, we obtain the following theorem.

**Theorem 6.3.** *Under the* $\mathsf{SIS}_{q,m',\beta}$ *assumption, the ring signature scheme of [BK10] combined with the* ExtractRandomness *algorithm described above is an unclaimable ring signature scheme.*

## Acknowledgements

We are grateful to Yael Tauman Kalai for helpful feedback on an earlier draft. Both authors' research was supported by the following grants: NSF MACS (CNS-1413920), DARPA IBM (W911NF-15-C-0236), Simons Investigator award agreement dated June 5th, 2012, and the Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant agreement CCF-0939370. Adam Sealfon was additionally supported by a DOE CSGF fellowship, DARPA/NJIT Palisade 491512803, Sloan/NJIT 996698, and MIT/IBM W1771646.

## References

[Ajt99] Miklós Ajtai. "Generating hard instances of the short basis problem". In: *ICALP*. 1999.

[BLO18] Carsten Baum, Huang Lin, and Sabine Oechsner. *Towards Practical Lattice-Based One-Time Linkable Ring Signatures*. Cryptology ePrint Archive 2018/107. 2018.

[BKM09] Adam Bender, Jonathan Katz, and Ruggero Morselli. "Ring Signatures: Stronger Definitions, and Constructions without Random Oracles". In: *J. Cryptology* 22.1 (2009), pp. 114–138.

[Bit17] Nir Bitansky. "Verifiable Random Functions from Non-interactive Witness-Indistinguishable Proofs". In: *TCC*. 2017.

[Boo+15] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. "Short Accountable Ring Signatures Based on DDH". In: *ESORICS*. 2015.

[BK10] Zvika Brakerski and Yael Tauman Kalai. "A Framework for Efficient Signatures, Ring Signatures and Identity Based Encryption in the Standard Model." In: *IACR Cryptology ePrint Archive 2010/086* (2010).

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. "Deniable encryption". In: *CRYPTO*. 1997.

[CPP18] Ran Canetti, Sunoo Park, and Oxana Poburinnaya. "Fully Bideniable Interactive Encryption". In: *IACR Cryptology ePrint Archive* 2018 (2018), p. 1244. URL: https://eprint.iacr.org/2018/1244.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. "Bonsai Trees, or How to Delegate a Lattice Basis". In: *EUROCRYPT*. 2010.

[CH91] David Chaum and Eugène van Heyst. "Group Signatures". In: *EUROCRYPT*. 1991.

[DN07] Cynthia Dwork and Moni Naor. "Zaps and Their Applications". In: *SIAM J. Comput.* 36.6 (2007).

[FS07] Eiichiro Fujisaki and Koutarou Suzuki. "Traceable Ring Signature". In: *PKC*. 2007.

[GPV07] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. "Trapdoors for Hard Lattices and New Cryptographic Constructions". In: *IACR Cryptology ePrint Archive 2007/432* (2007).

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. "Trapdoors for hard lattices and new cryptographic constructions". In: *STOC*. 2008.

[GO92] Shafi Goldwasser and Rafail Ostrovsky. "Invariant Signatures and Non-Interactive Zero-Knowledge Proofs are Equivalent (Extended Abstract)". In: *CRYPTO*. 1992.

[GHKW17] Rishab Goyal, Susan Hohenberger, Venkata Koppula, and Brent Waters. "A Generic Approach to Constructing and Proving Verifiable Random Functions". In: *TCC*. 2017.

[Ish+16] Ai Ishida, Keita Emura, Goichiro Hanaoka, Yusuke Sakai, and Keisuke Tanaka. "Group Signature with Deniability: How to Disavow a Signature". In: *CANS*. 2016.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014. ISBN: 9781466570269.

[LNWX17] San Ling, Khoa Nguyen, Huaxiong Wang, and Yanhong Xu. "Lattice-Based Group Signatures: Achieving Full Dynamicity with Ease". In: *ACNS*. 2017.

[LSW06] Joseph K. Liu, Willy Susilo, and Duncan S. Wong. "Ring Signature with Designated Linkability". In: *IWSEC*. 2006.

[LWW04] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. "Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups (Extended Abstract)". In: *ACISP*. 2004.

[Man00] R. Mankiewicz. *The story of mathematics*. The story of mathematics. Princeton University Press, 2000. URL: https://books.google.com/books?id=JXxrpwAACAAJ.

[Mel+13] Carlos Aguilar Melchor, Slim Bettaieb, Xavier Boyen, Laurent Fousse, and Philippe Gaborit. "Adapting Lyubashevsky's Signature Schemes to the Ring Signature Setting". In: *AFRICACRYPT*. 2013.

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. "Verifiable Random Functions". In: *FOCS*. 1999.

[MP12] Daniele Micciancio and Chris Peikert. "Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller". In: *EUROCRYPT*. 2012.

[MR07] Daniele Micciancio and Oded Regev. "Worst-Case to Average-Case Reductions Based on Gaussian Measures". In: *SIAM J. Comput.* 37.1 (2007), pp. 267–302.

[Mon] Monero. *Monero: Private Digital Currency*. https://www.getmonero.org.

[Ngu05] Lan Nguyen. "Accumulators from Bilinear Pairings and Applications". In: *Topics in Cryptology - CT-RSA*. 2005.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. "How to Leak a Secret". In: *ASIACRYPT*. 2001.

[SW14] Amit Sahai and Brent Waters. "How to use indistinguishability obfuscation: deniable encryption, and more". In: *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*. Ed. by David B. Shmoys. ACM, 2014, pp. 475–484. ISBN: 978-1-4503-2710-7. DOI: 10.1145/2591796.2591825. URL: https://doi.org/10.1145/2591796.2591825.

[SS10] Sven Schäge and Jörg Schwenk. "A CDH-Based Ring Signature Scheme with Short Signatures and Public Keys". In: *FC*. 2010.

[XY04] Shouhuai Xu and Moti Yung. "Accountable Ring Signatures: A Smart Card Approach". In: *Smart Card Research and Advanced Applications VI, IFIP 18th World Computer Congress, TC8/WG8.8 & TC11/WG11.2 Sixth International Conference on Smart Card Research and Advanced Applications (CARDIS), 22-27 August 2004, Toulouse, France*. Ed. by Jean-Jacques Quisquater, Pierre Paradinas, Yves Deswarte, and Anas Abou El Kalam. Vol. 153. IFIP. Kluwer/Springer, 2004, pp. 271–286. ISBN: 1-4020-8146-4. DOI: 10.1007/1-4020-8147-2_18. URL: https://doi.org/10.1007/1-4020-8147-2_18.

# A Definitions for repudiable-and-claimable ring signatures

**Definition A.1** (Repudiable-and-claimable ring signature)**.** A *repudiable-and-claimable ring signature scheme* is a ring signature scheme with an additional quadruple of algorithms

$$(\mathsf{Repudiate}, \mathsf{VerRepud}, \mathsf{Claim}, \mathsf{VerClaim}) \ ,$$

satisfying the five properties of *correctness* (Definition 2.2), *repudiability* (Definition A.2) *claimability* (Definition A.3), *anonymity* (Definition A.4), and *unforgeability* (Definition A.5).

The syntax of $(\mathsf{Repudiate}, \mathsf{VerRepud})$ and $(\mathsf{Claim}, \mathsf{VerClaim})$ are as defined in Definitions 3.3 and 3.9 respectively.

**Definition A.2** (Repudiability of repudiable-and-claimable ring signatures)**.** A ring signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfies *repudiability* if equipped with algorithms

$$(\mathsf{Repudiate}, \mathsf{VerRepud}, \mathsf{Claim}, \mathsf{VerClaim})$$

such that for any (possibly adversarial) PPT signing algorithm $\mathcal{A}_{\mathsf{Sign}}$, there exists a negligible function $\varepsilon$ such that (3) and (4) (from Definition 3.3) are satisfied when $\mathcal{O} = \{\mathsf{OSign}, \mathsf{ORpd}, \mathsf{OClaim}\}$.

**Definition A.3** (Claimability of repudiable-and-claimable ring signatures)**.** A ring signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ satisfies *claimability* if equipped with algorithms

$$(\mathsf{Repudiate}, \mathsf{VerRepud}, \mathsf{Claim}, \mathsf{VerClaim})$$

such that conditions 1, 2, and 3 of Definition 3.9 hold when $\mathcal{O} = \{\mathsf{OSign}, \mathsf{ORpd}\}$.

**Definition A.4** (Anonymity of repudiable-and-claimable ring signatures)**.** A repudiable-and-claimable ring signature scheme

$$(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Repudiate}, \mathsf{VerRepud}, \mathsf{Claim}, \mathsf{VerClaim}))$$

satisfies anonymity against

$$\{\text{adversarially chosen keys, attribution attacks, full key exposure}\}$$

if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $(\{\mathsf{OSign}, \mathsf{ORpd}, \mathsf{OClaim}\}, \varnothing, \alpha)$-anonymous (Definition 2.3) for, respectively,

$$\alpha \in \{2, 1, 0\} \ .$$

Moreover, the claimable ring signature satisfies the *adaptive* variants of the above anonymity definitions if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $(\{\mathsf{OSign}, \mathsf{ORpd}, \mathsf{OClaim}\}, \{\mathsf{OSign}, \mathsf{ORpd}, \mathsf{OClaim}\}, \alpha)$-anonymous for $\alpha \in \{2, 1, 0\}$ respectively.

**Definition A.5** (Unforgeability of repudiable-and-claimable ring signatures)**.** A repudiable-and-claimable ring signature scheme

$$(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, (\mathsf{Repudiate}, \mathsf{VerRepud}, \mathsf{Claim}, \mathsf{VerClaim}))$$

is *unforgeable* if $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is $\{\mathsf{ORpd}, \mathsf{OClaim}\}$-unforgeable (Definition 2.7).

# B Completing the proof of Lemma 4.12

*Proof (continued).* It remains to show that R-RS satisfies (4). Suppose not, for the sake of contradiction. Then there is a PPT algorithm $\mathcal{A}_{\mathsf{S\&R}}$ such that the following probability is non-negligible:

$$\Pr\left[\begin{array}{l} (vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k) \\ (\sigma, R', m, \{\xi_{vk}\}_{vk \in R' \setminus R}) \leftarrow \mathcal{A}^{\mathcal{O}}_{\mathsf{S\&R}}(R) \\ \forall vk \in R' \setminus R,\ b_{vk} \leftarrow \mathsf{VerRepud}(R', vk, \sigma, \xi_{vk}) \\ b' \leftarrow \mathsf{Verify}(R', \sigma, m) \end{array} : \begin{array}{l} R' \cap R \neq \varnothing \wedge \bigwedge\limits_{vk \in R' \setminus R} b_{vk} = 1 \\ \wedge b' = 1 \wedge Q \cap \{(\cdot, m, R')\} = \varnothing \end{array}\right] \quad (27)$$

where $R = \{vk_1, \ldots, vk_N\}$, $\mathcal{O} = \{\mathsf{OSign}, \mathsf{ORpd}\}$, and $Q$ is the set of queries made to oracle $\mathsf{OSign}$.

Based on $\mathcal{A}_{\mathsf{S\&R}}$, we construct another adversary $\mathcal{A}'$ to Parallel VRF Game for $2N$ keys, as follows. $\mathcal{A}'$ first samples

$$(vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k)$$

and parses each $vk_i$ as

$$vk_i = (\vec{vk}^{\,i}_{\mathsf{VRF}} = (vk^{i,1}_{\mathsf{VRF}}, vk^{i,2}_{\mathsf{VRF}}, vk^{i,3}_{\mathsf{VRF}}, vk^{i,4}_{\mathsf{VRF}}), \rho_i, \vec{\alpha}_i) \ .$$

Next, $\mathcal{A}'$ obtains $2N$ verification keys from the VRF challenger. For notational convenience in the rest of the proof, let these $2N$ keys be denoted by $\{vk_{\mathsf{VRF}}^{i,3,*}, vk_{\mathsf{VRF}}^{i,4,*}\}_{i\in[N]}$. Let the corresponding $2N$ secret keys be denoted by $\{sk_{\mathsf{VRF}}^{i,3,*}, sk_{\mathsf{VRF}}^{i,4,*}\}_{i\in[N]}$.[23]

Define $vk_i^*$ as

$$vk_i^* = (\vec{vk}_{\mathsf{VRF}}^{\,i} = (vk_{\mathsf{VRF}}^{i,1}, vk_{\mathsf{VRF}}^{i,2}, vk_{\mathsf{VRF}}^{i,3,*}, vk_{\mathsf{VRF}}^{i,4,*}), \rho_i, \vec{\alpha}_i) \ .$$

That is, $vk_i^*$ is identical to $vk_i$ except that $vk_{\mathsf{VRF}}^{i,3}$ and $vk_{\mathsf{VRF}}^{i,4}$ are replaced by $vk_{\mathsf{VRF}}^{i,3,*}, vk_{\mathsf{VRF}}^{i,4,*}$.

Let $R^* = \{vk_1^*, \ldots, vk_N^*\}$. $\mathcal{A}'$ then runs $\mathcal{A}_{\mathsf{S\&R}}$ on input $R^*$, answering its oracle queries as follows.

- On query $(i'', m'', R'')$ to $\mathsf{OSign}$: $\mathcal{A}'$ runs the honest signing algorithm $\mathsf{R\text{-}RS.Sign}$ on input $(R'' \cup \{vk_{i''}^*\}, sk_{i''}, m'')$, with the following modification: in step 6, instead of using $sk_{i''}$ to generate $y_3, \tau_3$, and $y_4, \tau_4$, $\mathcal{A}'$ invokes the corresponding VRF oracle for keys $vk_{\mathsf{VRF}}^{i'',3,*}, vk_{\mathsf{VRF}}^{i'',4,*}$.
- On query $(i'', \sigma'', R'')$ to $\mathsf{ORpd}$: $\mathcal{A}'$ runs the honest repudiation algorithm $\mathsf{R\text{-}RS.Repudiate}$ on input $(R'' \cup \{vk_{i''}^*\}, sk_{i''}, \sigma'')$. (As noted above, $sk_{\mathsf{VRF}}^3$ and $sk_{\mathsf{VRF}}^4$ are not used in algorithm $\mathsf{R\text{-}RS.Repudiate}$, so $\mathcal{A}'$ does not need to invoke the VRF oracle here.)

Let $(\sigma, \xi, m, R')$ be the output of $\mathcal{A}_{\mathsf{S\&R}}$. $\mathcal{A}'$ parses $\sigma = (\vec{\pi}, \vec{y}, \varphi)$ and $\vec{y} = (y_1, \ldots, y_4)$, then submits $(R', m, \varphi)$ to the VRF challenger and, for each $i \in [N]$, receives responses $y_{3,i}, y_{4,i}$ corresponding to . If there exists any index $i \in [N]$ such that $y_{3,i} = y_3$ or $y_{4,i} = y_4$, $\mathcal{A}'$ outputs 0. Otherwise, $\mathcal{A}'$ outputs a random bit. Let us now consider the behavior of $\mathcal{A}'$ in the two cases where the VRF challenger's bit $b$ is equal to 0 and equal to 1.

**Case $b = 0$.** In this case, the view of $\mathcal{A}_{\mathsf{S\&R}}$ is identical to the view in (27), so by assumption $\mathcal{A}_{\mathsf{S\&R}}$ wins the game described in (27) with non-negligible probability. Note that whenever $\mathcal{A}_{\mathsf{S\&R}}$ wins the game, the condition $Q \cap \{(\cdot, m, R')\} \neq \varnothing$ in (27) implies that $\mathcal{A}$ has not previously made an oracle query on the VRF challenge message $(R', m, \varphi)$ during the query phase. Let us suppose that $\mathcal{A}_{\mathsf{S\&R}}$ wins the game described in (27) and consider the implications.

By definition, if $\mathsf{R\text{-}RS.Verify}$ accepts (with non-negligible probability) on input $(R', \sigma, m)$, then

$$\forall i \in [|R'|], \ \mathsf{ZAP.Verify}_L(\rho_i, \pi_i, (R', m, \varphi, \vec{y})) = 1 \ . \tag{28}$$

$R' \cap R \neq \varnothing$ from our assumption that $\mathcal{A}_{\mathsf{S\&R}}$ wins the game described in (27), and therefore we have that at least one $\rho_{i'}$ corresponding to some $vk_{i'} \in R' \cap R$ is honestly generated. Thus, the soundness of the ZAP holds w.r.t. this $\rho_{i'}$, and (28) implies that $(R', m, \varphi, \vec{y}) \in L$. Moreover, by the definition of $L$,

$$(R', m, \varphi, \vec{y}) \in L \Rightarrow (b_3 \vee b_4), \tag{29}$$

where $b_3, b_4$ are as defined in Definition 4.6. Expanding the definitions of $b_3, b_4$, we have that the right-hand side of (29) implies:

$$\begin{gathered} \exists \eta \in \{3,4\}, i^* \in [|R'|], \tau_1, \ldots, \tau_4, \gamma \quad \text{s.t.} \quad \forall i' \in [|R'|], j' \in [M], \\ \mathsf{VRF.Verify}(vk_{\mathsf{VRF}}^{i^*,\eta,*}, (R', m, \varphi), y_\eta, \tau_\eta; \alpha_{j'}^{i'} \oplus \gamma) = 1 \ . \end{gathered} \tag{30}$$

Then applying Corollary 4.11 (again setting the algorithm $\mathcal{V}$ to be $\mathsf{VRF.Verify}$): (30) implies *either*

$$\begin{gathered} \exists \eta \in \{3,4\} \text{ and } \tau \text{ s.t.} \\ \Pr\left[\mathsf{VRF.Verify}(vk_{\mathsf{VRF}}^{i^*,\eta,*}, (R', m, \varphi), y_\eta, \tau) = 1\right] \text{ is overwhelming,} \end{gathered} \tag{31}$$

[23] Note that $\{sk_{\mathsf{VRF}}^{i,3,*}, sk_{\mathsf{VRF}}^{i,4,*}\}_{i\in[N]}$ are generated by the VRF challenger and not accessible by $\mathcal{A}'$.

*or* a negligible probability event occurred. By the complete and unique provability of the VRF, (31) implies that

$$y_3 = \mathsf{VRF.Eval}(sk^{i,3,*}_{\mathsf{VRF}}, (R', m, \varphi)) \text{ or } y_4 = \mathsf{VRF.Eval}(sk^{i,4,*}_{\mathsf{VRF}}, (R', m, \varphi)) \ . \tag{32}$$

Chaining together the implications, we obtain that (32) holds with all but negligible probability conditonioned on the non-negligible-probability event of $\mathcal{A}_{\mathsf{S\&R}}$ winning the game described in (27).

Finally, by definition of Parallel VRF Game, when $b = 0$, for all $i \in [N]$,

$$y_{3,i} = \mathsf{VRF.Eval}(sk^{i,3,*}_{\mathsf{VRF}}, (R', m, \varphi)) \text{ and } y_{4,i} = \mathsf{VRF.Eval}(sk^{i,4,*}_{\mathsf{VRF}}, (R', m, \varphi)) \ .$$

It follows that conditioned on $b = 0$, there is a non-negligible probability that

$$\exists i^* \in [N] \text{ s.t. } y_3 = y_{3,i^*} \text{ or } y_4 = y_{4,i^*} \ . \tag{33}$$

Recall that (33) is the trigger condition for $\mathcal{A}'$ to output 0. Therefore, when $b = 0$, $\mathcal{A}'$ outputs 0 with non-negligible probability (and outputs a random bit the rest of the time).

**Case $b = 1$.** In this case, $y'_3$ and $y'_4$ are uniformly random and independent of the rest of the experiment, so with overwhelming probability, they will be distinct from $y_{3,i}$ and $y_{4,i}$ for every $i \in [N]$. Consequently, in this case, with all but negligible probability $\mathcal{A}$ outputs a random bit.

Thus, $\mathcal{A}'$ wins Parallel VRF Game with non-negligible probability. This contradicts the security of the VRF. We therefore conclude that R-RS satisfies (4). The lemma follows. □

# C Anonymity of R-RS

*Lemma* 4.14. R-RS is satisfies adaptive anonymity against adversarially chosen keys (Definition 3.4).

*Proof.* Suppose that this is not the case. Then there exists some $N = \mathsf{poly}(k)$ and PPT $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ such that the following probability is non-negligibly greater than $1/2$, where $I$ is the set of queries to the corruption oracle and $\mathcal{O} = \{\mathsf{OSign}, \mathsf{ORpd}\}$:

$$\Pr\left[\begin{array}{ll} (vk_1, sk_1), \ldots, (vk_N, sk_N) \leftarrow \mathsf{Gen}(1^k) & \\ ((m^*, i_0^*, i_1^*, R^*), \mathfrak{s}) \leftarrow \mathcal{A}_1^{\mathcal{O},\mathsf{Corr}}(vk_1, \ldots, vk_N) & \\ b \leftarrow \{0,1\} & : \ b' = b \wedge \{i_0^*, i_1^*\} \cap I = \varnothing \\ \sigma \leftarrow \mathsf{Sign}(R^* \cup \{vk_{i_0^*}, vk_{i_1^*}\}, sk_{i_b^*}, m^*) & \\ b' \leftarrow \mathcal{A}_2^{\mathcal{O},\mathsf{Corr}}(\mathfrak{s}, \sigma) & \end{array}\right] .$$

We proceed via a sequence of hybrids.

**Hybrid 1.** The honest experiment, with $b = 0$.

**Hybrid 2.** Identical to the above, but in the generation of signature $\sigma = (\vec{\pi}, (y_1, \ldots, y_4))$, $y_2$ and $y_4$ are generated at random while $y_1$ and $y_3$ are still generated using the VRFs for party $i_0^*$.

By the security of the VRF, this hybrid is indistinguishable from Hybrid 1.

**Hybrid 3.** Identical to the above, but in the generation of signature $\sigma = (\vec{\pi}, (y_1, \ldots, y_4))$, $y_2$ and $y_4$ are generated using the VRFs for party $i_1^*$ rather than $i_0^*$. That is, parsing $sk_{i_1^*} = ((sk_{\mathsf{VRF}}^{i_1^*,1}, \ldots, sk_{\mathsf{VRF}}^{i_1^*,4}), vk_{i_1^*})$, we set

$$y_2 = \mathsf{VRF.Eval}(sk_{\mathsf{VRF}}^{i_1^*,2}, (R, m, \varphi))$$

and

$$y_4 = \mathsf{VRF.Eval}(sk_{\mathsf{VRF}}^{i_1^*,4}, (R, m, \varphi)).$$

By the security of the VRF, this hybrid is indistinguishable from Hybrid 2.

**Hybrid 4.** Identical to the above, but in the generation of signature $\sigma$, the ZAPs in the $\mathsf{Sign}$ procedure are proven with respect to the witnesses for party $i_1^*$ rather than $i_0^*$. That is, parsing $sk_{i_1^*} = ((sk_{\mathsf{VRF}}^{i_1^*,1}, \ldots, sk_{\mathsf{VRF}}^{i_1^*,4}), vk_{i_1^*})$, in the invocation of $\mathsf{Sign}$ to generate $\sigma$, let $\tau_2' = \mathsf{VRF.Prove}(sk_{\mathsf{VRF}}^{i_1^*,2}, (R, m, \varphi))$ and $\tau_4' = \mathsf{VRF.Prove}(sk_{\mathsf{VRF}}^{i_1^*,4}, (R, m, \varphi))$. For each $i \in [N]$, in step 7 of $\mathsf{Sign}$, generate $\pi_i$ by invoking

$$\mathsf{ZAP.Prove}(\rho_i, \mathsf{stmt}, (i_1^*, \bot, \tau_2', \bot, \tau_4', \gamma))$$

instead of using the witness for $i_0^*$.

By the witness indistinguishability property of the ZAP, this hybrid is indistinguishable from Hybrid 3.

**Hybrid 5.** Identical to the above, but in the generation of signature $\sigma = (\vec{\pi}, (y_1, \ldots, y_4))$, $y_1$ and $y_3$ are generated at random while $y_2$ and $y_4$ are still generated using the VRFs for party $i_1^*$.

By the security of the VRF, this hybrid is indistinguishable from Hybrid 4.

**Hybrid 6.** Identical to the above, but in the generation of signature $\sigma = (\vec{\pi}, (y_1, \ldots, y_4))$, $y_1$ and $y_3$ (as well as $y_2$ and $y_4$) are now generated using the VRFs for party $i_1^*$.

By the security of the VRF, this hybrid is indistinguishable from Hybrid 5.

**Hybrid 7.** The honest experiment, with $b = 1$.

By the witness indistinguishability property of the ZAP, this hybrid is indistinguishable from Hybrid 6. □

# D Explaining randomness of discrete Gaussian samples

Our construction requires "explaining" algorithms $\mathsf{ExplainDist}$ and $\mathsf{ExplainCond}$. $\mathsf{ExplainDist}$ must, on input $(x, y)$, output $\rho$ distributed according to the conditional distribution

$$\left\{ \rho \;\middle|\; \mathsf{SampleDist}(x; \rho) = y \right\}.$$

Similarly, $\mathsf{ExplainCond}$ must, on input $(x, y)$, output $\rho$ distributed according to

$$\left\{ \rho \;\middle|\; \mathsf{SampleCond}(x; \rho) = y \right\}.$$

In this appendix, we outline how this "explaining randomness" is able to be efficiently computed according to the required distribution.

In our underlying instantiation, as in the original construction of [BK10]:

- SampleDist is instantiated by the function $\mathsf{SampleD}(B, s, c)$ defined in [GPV07, §4.2].[24]
- SampleCond is instantiated by the function $\mathsf{SampleISIS}(A, T, s, u)$ defined in [GPV07, §5.3.2].

We first recall the basis randomization technique of [CHKP10]. In the following, tildes denote Gram-Schmidt orthogonalization.

**Lemma D.1** (Lemma 3.3 in [CHKP10])**.** *There exists a probabilistic polynomial-time algorithm* $\mathsf{RandBasis}(T, s)$ *that takes as input a basis* $T$ *of an* $m'$*-dimensional integer lattice and a parameter* $s \geq \|\tilde{T}\| \cdot \omega(\sqrt{\log n})$*, and outputs a basis* $T'$ *for the same lattice, such that:*

1. *With overwhelming probability,* $\|\mathsf{RandBasis}(T, s)\| \leq s \cdot \sqrt{m'}$
2. *For any pair of bases* $T_1, T_2$ *for the same lattice and any* $s \geq \max(\|\tilde{T_1}\|, \|\tilde{T_2}\|) \cdot \omega(\sqrt{\log n})$*, the outputs of* $\mathsf{RandBasis}(T_1, s)$ *and* $\mathsf{RandBasis}(T_2, s)$ *have negligible statistical distance.*

We now describe the relatively simple algorithms SampleD and SampleISIS in order to outline how one can efficiently compute randomness to explain any particular output. SampleISIS invokes SampleD, and SampleD in turn invokes a simpler algorithm $\mathsf{Sample}\mathbb{Z}$ which is also defined in [GPV07]. We describe these three algorithms in the order they are listed in the preceding sentence.

**Definition D.2.** SampleISIS takes as input $(A, T, s, u)$ where $A$ and $T$ are matrices, $s$ is a Gaussian parameter, and $u$ is a vector.

1. Let $T' \leftarrow \mathsf{RandBasis}(T, s)$.
2. By Gaussian elimination, choose an arbitrary $t$ such that $At = u \pmod q$.
3. Sample $v \leftarrow \mathsf{SampleD}(T', s, -t)$.
4. Output $e = t + v$.

**Claim D.3.** There is an efficient algorithm for ExplainCond if there is an efficient algorithm for ExplainDist.

*Proof.* Essentially, the claim follows from the fact that Step 3 is the only randomized step in SampleISIS. Recall that ExplainCond needs to correctly sample the randomness distribution of SampleCond (i.e., SampleISIS) conditioned on a particular input and output. Given the input, $t$ (Step 2) can be computed deterministically. Finally, given the output $e$ and $t$, it remains only to explain the randomness of SampleD conditioned on input $(T, s, -t)$ and output $e - t$. □

**Definition D.4.** SampleD takes as input $(B, s, c)$ where $B = (b_1, \ldots, b_n)$ is a matrix describing a lattice basis, $s$ is a Gaussian parameter, and $c$ is a vector.

1. Let $v_n := 0$ (the zero vector) and $c_n := c$. For $i = n, \ldots, 1$:
   (a) Let $c'_i := \langle c_i, \tilde{b_i} \rangle / \langle \tilde{b_i}, \tilde{b_i} \rangle \in \mathbb{R}$ and $s'_i = s / \|\tilde{b_i}\|_2 > 0$.
   (b) Sample $z_i \leftarrow \mathsf{Sample}\mathbb{Z}(s'_i, c'_i)$.
   (c) Let $c_{i-1} := c_i - z_i b_i$ and let $v_{i-1} := v_i + z_i b_i$.
2. Output $v_0$.

**Claim D.5.** There is an efficient algorithm for ExplainDist if there is an efficient algorithm to "explain $\mathsf{Sample}\mathbb{Z}$" — i.e., an efficient algorithm that, on input $(s, c, x)$, samples from the following distribution:

$$\left\{ \rho \;\middle|\; \mathsf{Sample}\mathbb{Z}(s, c; \rho) = x \right\}. \tag{34}$$

[24] We cite the full version here for the detailed description of the function; the conference version is [GPV08].

*Proof (sketch).* Each output of $\mathsf{SampleDist}$ on a given input induces a unique set of $z_i$s — in other words, as noted in [GPV07], there is a bijective correspondence between the random choices of the $z_i$s and the output. After recovering these $z_i$s, it remains only to explain the randomness of $\mathsf{Sample}\mathbb{Z}$ conditioned on inputs $(s'_i, c'_i)$ and outputs $z_i$ for each $i$. □

**Definition D.6.** $\mathsf{Sample}\mathbb{Z}$ takes input $(s, c)$ where $s$ is a Gaussian parameter and $c \in \mathbb{R}$. In the following, $t(n) \in \omega(\sqrt{\log(n)})$ is some fixed function, say, $t(n) = \log(n)$; $\mathsf{Ber}(p)$ denotes the Bernoulli distribution with probability $p$ of outputting 1; and $\rho_s(\cdot)$ is the Gaussian measure, defined as $\rho_s(x) = \exp(-\pi\|x\|_2^2/s^2)$.

1. Let $X = \mathbb{Z} \cap [c - s \cdot t(n), c + s \cdot t(n)]$ and sample uniformly $x \leftarrow X$.
2. Sample $b \leftarrow \mathsf{Ber}(\rho_s(x - c))$.
3. If $b = 1$, output $x$. Else, go back to step 1.

**Claim D.7.** There is an efficient algorithm that, on input $(s, c, x)$, samples from the distribution described in (34).

*Proof (sketch).* $\mathsf{Sample}\mathbb{Z}$ consists of rejection sampling based on a biased coinflip, where the bias depends on the present sample $x$. The randomness $\rho$ to explain a particular output of $\mathsf{Sample}\mathbb{Z}$ can be thought to consist of a series of "rejected" samples $x_1, \ldots, x_{\ell-1}$ followed by the "accepted" sample $x_\ell$ (which must be equal to $x$). The length $\ell$ of this sequence follows a geometric distribution parametrized by the *expected* bias $\beta$ over the entire domain $X$. That is,

$$\beta = \frac{1}{|X|} \sum_{x \in X} \rho_s(x - c) \ .$$

Once a value of $\ell$ is sampled from the appropriately parametrized geometric distribution, it remains only to sample the "failed attempts" $x_1, \ldots, x_{\ell-1}$. This can be done by the following procedure. For each $i \in [\ell - 1]$:

1. Sample uniformly $x' \leftarrow X$.
2. Sample $b \leftarrow \mathsf{Ber}(\rho_s(x' - c))$.
3. If $b = 0$, then set $x_i := x'$ and continue. Else, go back to Step 1.

□