# Compact Ring Signatures from Learning With Errors

Rohit Chatterjee[1], Sanjam Garg[2,3], Mohammad Hajiabadi[4], Dakshita Khurana[5], Xiao Liang[1], Giulio Malavolta[6], Omkant Pandey[1], and Sina Shiehian[1,2]

[1] Stony Brook University
[2] University of California, Berkeley
[3] NTT Research
[4] University of Waterloo
[5] University of Illinois Urbana-Champaign
[6] Max Planck Institute for Security and Privacy

**Abstract.** Ring signatures allow a user to sign a message on behalf of a "ring" of signers, while hiding the true identity of the signer. As the degree of anonymity guaranteed by a ring signature is directly proportional to the size of the ring, an important goal in cryptography is to study constructions that minimize the size of the signature as a function of the number of ring members.

In this work, we present the first compact ring signature scheme (i.e., where the size of the signature grows logarithmically with the size of the ring) from the (plain) learning with errors (LWE) problem. The construction is in the standard model and it does not rely on a common random string or on the random oracle heuristic. In contrast with the prior work of Backes *et al.* [EUROCRYPT'2019], our scheme does not rely on bilinear pairings, which allows us to show that the scheme is post-quantum secure assuming the quantum hardness of LWE.

At the heart of our scheme is a new construction of compact and statistically witness indistinguishable ZAP arguments for NP ∩ coNP, that we show to be sound based on the plain LWE assumption. Prior to our work, statistical ZAPs (for all of NP) were known to exist only assuming *sub-exponential* LWE. We believe that this scheme might find further applications in the future.

## 1 Introduction

In a *ring signature* scheme, (introduced in [44]) a user can sign a message with respect to a *ring* of public keys. The ring can be arbitrarily chosen by the signer and the verification keys that populate the ring can be sampled locally by each user, i.e., no central coordination is required. No user or entity should be able to tell which user in the ring actually produced a given signature — a property referred to as *anonymity*. This is complemented with the standard notion of *unforgeability* for signatures, which in this case requires that no user outside a specified ring should be able to produce valid signatures on behalf of this ring. A salient feature is the *online* or *setup-free* generation of ring signatures, which

requires that signatures can be generated without any prior interaction between members of the ring. Ring signatures and their variants have found natural applications related to whistleblowing, authenticating leaked information, and more recently to cryptocurrencies [47,37].

There is a sizeable body of work [29,39,5,2] that construct ring signatures under various definitions and hardness assumptions. As the degree of anonymity guaranteed by the ring signature is directly proportional to the size of the ring, an important property of ring signatures becomes *compactness*, which requires that signatures only have a logarithmic (or lower) dependence on the size of the ring. Recently, the work of [2] provided a compact ring signature construction in the plain model (i.e., not needing a common random string or a setup). Their construction assumes the existence of the following: noninteractive witness indistinguishable proofs or NIWIs (which are known only from bilinear pairing based assumptions or indistinguishability obfuscation), somewhere perfectly binding (SPB) hashes, public key encryption with oblivious public key generation and pseudorandom ciphertexts, and a standard signature scheme. While most of these primitives are known under a variety of cryptographic assumptions including LWE, unfortunately, we currently do not know any constructions of NIWI proofs from LWE (please see the technical overview for related discussion).

This leads to the natural question of whether NIWIs are necessary to construct compact ring signatures in the plain model. The looming threat of quantum computers makes this question particularly pressing, since we would lose our only candidate construction to quantum attacks (due to the reliance on bilinear maps). We therefore ask the following open question:

*Can we obtain compact (logarithmic size) ring signatures from the hardness of standard learning with errors (LWE) problem?*

### 1.1 Our Results

Our main contribution resolves the open problem stated above. In other words, we obtain a ring signature construction from *plain* or *standard* LWE, i.e., only assuming polynomial hardness of the LWE problem with polynomial modulus-to-noise ratio. Our result is obtained as follows:

**ZAPs for NP $\cap$ coNP.** The first key step to our construction of ring signatures is realization of a new argument system that we call *relaxed ZAPs for extended* $\mathsf{NP} \cap \mathsf{coNP}$. These are akin to ZAPs but with a few additional restrictions, and can also be viewed as a generalization of (non-adaptive) ZAPs for languages in $\mathsf{NP} \cap \mathsf{coNP}$. We show how to construct these ZAPs from the plain (polynomially-hard) LWE. This is in contrast with the known constructions of ZAPs for NP [4,25,31] that assume *subexponential* hardness of LWE. Our ZAP construction also enjoys several other attractive properties such as statistical witness indistinguishability and proof compactness; which we expect will make them useful in other application contexts. We defer further exposition to our technical overview.

**Compact ring signatures from LWE.** Next, we show that our notion of relaxed ZAPs, along with SPB hash schemes and a special public key encryption scheme, is sufficient to construct compact ring signatures. All these components have constructions from plain LWE. Thus, we obtain the *first* construction of compact ring signatures in the plain model from purely post-quantum assumptions in the literature. In addition, we investigate security in the *fully quantum* setting, where the adversary can query the signing oracle on a superposition of messages. Towards this goal, we give a construction that retains unforgeability and anonymity in this setting.

### 1.2 Background

**Fiat-Shamir transformation, trapdoor $\Sigma$-protocols, and correlation intractability.** A trapdoor $\Sigma$-protocol [14] for a language $L$ is a 3-move (honest verifier) zero-knowledge protocol between a prover and a verifier, where the prover tries to convince the verifier about the veracity of a statement $x$. The protocol is instantiated with an *extractable* commitment scheme where there is an extraction trapdoor $td$ which allows extracting the plaintexts in the commitments. In the first move of the protocol, the prover commits to a string $a$ and sends this commitment $\mathsf{Com}(a)$ to the verifier. Next, in the second move, the verifier sends a challenge $b$ to the prover. In the final round, based on the challenge $b$, the prover computes the final message and sends it to the verifier. The distinctive property of trapdoor sigma protocols is that for a false statement $x$, there is at most one *bad challenge* $b^*$ which lets a malicious prover to successfully complete its proof, and this bad challenge is efficiently computable given $a$. Consequently, given the extraction trapdoor $td$, the bad challenge can be efficiently computed from $\mathsf{Com}(a)$.

The Fiat-Shamir transformation [20] can convert a trapdoor $\Sigma$-protocol (indeed any $\Sigma$-protocol) to a noninteractive protocol in the random oracle model. The way it works is that the prover evaluates a hash function on its first message to compute the challenge $b$, and then proceeds to send the full proof generated using this challenge to the verifier. To argue soundness, notice that since the bad challenge is unique and the hash function is modeled as a random oracle, a malicious prover has a negligible chance of finding a first message such that the hash of it equals the bad challenge $b^*$.

A line of work [14,42] builds a special type of hash functions called *correlation intractable* (CI) hash functions, to securely instantiate the Fiat-Shamir heuristic in the plain model for trapdoor $\Sigma$-protocols, thus turning them into noninteractive protocols in the CRS model. In particular, the LWE-based CI hash functions in [42] allow building noninteractive zero knowledege protocols for all of NP from the LWE assumption. Informally, a hash function is CI for a class of circuits if for any circuit $C$ in the class, given a random hash key $k$, it is hard to find an input $z$ whose image under the hash function equals $C(z)$. We can securely replace CI hash functions with random oracles when we apply the Fiat-Shamir transformation to trapdoor $\Sigma$-protocols. To see this, notice that since in trapdoor

$\Sigma$-protocols the bad challenge is efficiently computable given the prover's first message, a malicious prover that can find a first message which gets mapped to the bad challenge is breaking the correlation intractability of the hash function.

**Somewhere perfectly binding hash.** A somewhere perfectly binding (SPB) hash is similar to a Merkle tree [34]: it can compress a large database of $N$ records into a small digest. In a SPB hash with private local openings, binding holds perfectly for a single hidden index $i \in [N]$. In more detail, the SPB hash key generation algorithm takes as input a binding index $i \in [N]$ and outputs a pair of keys $(hk, shk)$. The hash key $hk$ can be used to generate a digest. The secret key $shk$, can be used to generate a short opening for the $i$th record in the database. Perfect binding says that a valid opening for the $i$th location of the database uniquely determines the value of that location. Also, the hash key $hk$ is index hiding, i.e, it computationally hides the binding index $i$.

**Compact ring signatures of [2].** The construction of [2] is based on four ingredients: a noninteractive witness indistinguishable argument system $\mathsf{NIWI}$, a public key encryption scheme $\mathsf{PKE}$, a standard signature scheme $\mathsf{Sig}$, and a somewhere perfectly binding hash scheme $\mathsf{SPB}$. In this scheme, each verification key consists of two components: a uniformly chosen public key $pk$ (not generated through the key generation algorithm of $\mathsf{PKE}$) and a standard signature verification key $vk$. The signing key consists of $sk$, the corresponding signing key for $vk$. To sign a message $m$ using signing key $sk_i$ corresponding to verification key $\mathsf{VK_i} = (vk_i, pk_i)$, and on behalf of ring $\mathsf{R} = (\mathsf{VK}_1, \cdots, \mathsf{VK}_\ell)$, the signer

- first generates a standard signature $\sigma$ for message $m$ as $\sigma \leftarrow \mathsf{Sig.Sign}(sk_i, m)$,
- encrypts $\sigma$ under $pk_i$ with random coins $r$ to get ciphertext $c \leftarrow \mathsf{PKE.Enc}(pk_i, \sigma; r)$,
- generates a binding SPB key pair for index $i$, $(hk, shk) \leftarrow \mathsf{SPB.Gen}(i)$,
- hashes the ring $\mathsf{R}$ with $hk$ to get a short digest $h := \mathsf{SPB.Hash}(hk, \mathsf{R})$,
- creates an opening for the $i$th location $\tau \leftarrow \mathsf{SPB.Open}(hk, shk, \mathsf{R}, i)$,
- generates a NIWI proof $\pi$ which using the short opening $\tau$ proves existence of a verification key $\mathsf{VK}_i = (vk_i, pk_i)$ in the ring $\mathsf{R}$ such that under $pk_i$, $c$ encrypts a valid signature under verifiable with $vk_i$,
- finally, it publishes $(\pi, c, hk)$ as the signature for message $m$.

To prove unforgeability, we switch to a hybrid where we generate each $pk_i$ with a corresponding secret key $sk_i$. Consequently, perfect binding of $\mathsf{SPB}$ and soundness of $\mathsf{NIWI}$ imply that given a forgery $(\pi, c, hk)$ for ring $R$, there exists a $sk_i$ such that $\mathsf{PKE.Dec}(sk_i, c)$ is valid forgery against $\mathsf{Sig}$. Proving anonymity involves techniques similar to [36].

### 1.3 Technical Overview

**ZAP instead of NIWI.** As already mentioned, the issue that prevents [2] from basing their construction solely on LWE is their reliance on NIWIs. Our starting point is the observation of [6] which proposes using two-message public

coin argument systems, known in the literature as ZAPs [18], instead of NIWIs. Namely, we can add a ZAP first message $\rho$ to each verification key. The signer now picks the lexicographically smallest verification key $\mathsf{VK}_1 = (vk_1, pk_1, \rho_1) \in \mathsf{R}$ in the ring and uses $\rho_1$ to generate a proof. Using LWE-based ZAPs constructed in [4,25,31], this approach gives us LWE based compact ring signatures. However, there is a major caveat: none of the LWE-based ZAPs mentioned above are based on polynomial hardness assumptions. They all need super-polynomial hardness of LWE (in fact subexponential hardness if the goal is to achieve conventional $\lambda$ bits of security). Therefore, using lattice based ZAPs generically seems to be unsatisfactory as the resulting construction would rely on qualitatively stronger assumptions.

**ZAPs for $\mathsf{NP} \cap \mathsf{coNP}$.** Our next insight is that we may not need ZAPs for all of $\mathsf{NP}$. Assume that in the forgery $\Sigma^* = (\pi^*, c^*, hk^*)$, the ciphertext $c^*$ is guaranteed to be encrypted under one of the public keys $pk_i$. In the unforgeability game we can generate $pk_i$s with corresponding secrets $sk_i$. In this case, given $sk_i$, checking that $\Sigma^*$ is a forgery can be done efficiently, i.e., $sk_i$ is a witness for the fact that $\Sigma^*$ is not a valid signature. Therefore, ZAPs for $\mathsf{NP} \cap \mathsf{coNP}$ might suffice for our application.

It turns out that for $\mathsf{NP} \cap \mathsf{coNP}$ we can build ZAPs based on the polynomial hardness of LWE. We will now describe a ZAP protocol for any arbitrary language $L \in \mathsf{NP} \cap \mathsf{coNP}$. The ZAP that we describe here is constructed by following the general framework of converting a $\Sigma$-protocol to a noninteractive protocol using a CI hash function [14,42]. More specifically, we describe a two round commitment scheme and use it to instantiate [14,42]. Our commitment scheme is defined with respect to the complement language $\overline{L}$. To commit to a bit $b$ and generate the second commitment message, the sender specifies a statement $x$. The receiver can recover the committed bit, if, when generating the first message it specified a (non-)witness $\overline{w}$ for the fact that $x \in \overline{L}$. If $x \notin \overline{L}$, the committed bit is hidden. The commitment scheme works as follows:

- The receiver sends an arbitrary bitstring $\overline{w}$ via the first message of a statistically sender private (1 out of 2) OT, $\mathsf{OT1}(\overline{w})$.
- The sender garbles the following circuit: on input $\overline{w}$, if $\overline{w}$ is a witness for $x \in \overline{L}$, output $b$, otherwise, output 0. The sender sends the garbled circuit along with a OT second message containing the labels $\mathsf{OT2}(\{\mathsf{lbl}_{i,0}, \mathsf{lbl}_{i,1}\})$.

We instantiate [14,42] for language $L$ with our two round *witness extractable commitment* for $\overline{L}$, to get a ZAP for $\mathsf{NP} \cap \mathsf{coNP}$: the verifier sends the commitment first message along with a key for a CI hash function, the prover uses the hash key and the commitment scheme to proceed as in [14]. Recall that to apply the transformation of [14,42], we have to make sure that when $x \notin L$, the commitments used in the underlying $\Sigma$-protocol are extractable given the extraction trapdoor. If $x \notin L$, the verifier can (undetectably) switch to generating the first commitment message using a non-witness $\overline{w_x}$ for $x \in \overline{L}$ and this will let the soundness proof go through.

Unfortunately, when trying to use our $\mathsf{NP} \cap \mathsf{coNP}$ ZAPs to build ring signatures we encounter two issues:

1. The ZAPs that we need for our ring signatures need to be adaptively sound. However, the ZAPs that we just constructed are only selectively sound. This is because, in the soundness reduction we have to switch to a commitment first message that depends on a non-witness $\overline{w}$ for the statement $x$ (depends on a $\overline{w}$ such that $(x, \overline{w}) \in \overline{R}$ ).
2. We assumed the ciphertext $c^*$ in the forgery is a valid encryption under one of the public keys of the ring. This may not be true. In particular, $\overline{L}$ might not be in $\mathsf{NP}$.

The first issue seems relatively easy to resolve. Our ZAP construction already achieves a weak notion of adaptivity that is sufficient for our purpose: as long as the non-witness $\overline{w}$ is fixed in advance, the cheating prover cannot come up with a valid proof for a statement $x$ where $(x, \overline{w}) \in \overline{R}$. In our case, the non-witnesses which are the secret keys corresponding to $pk_i$s are clearly fixed in advance.

**Extending the complement language.** For the second issue, our solution is to *extend* $\overline{L}$ to a language $\widetilde{L} \in \mathsf{NP}$ and use a witness extractable commitment for this extended language $\widetilde{L}$. In more detail, given a forgery $\Sigma^*$ for a ring $\mathsf{R}^*$ of size $\ell$, we define statement $x = (\Sigma^*, \mathsf{R}^*)$. For a witness $\widetilde{w} = (sk_1, \cdots, sk_\ell)$ we say that $(x, \widetilde{w}) \in \widetilde{L}$, if each $sk_i$ is a valid secret key for $pk_i$, and decrypting $c^*$ with any $sk_j$ does not yield a valid standard signature corresponding to $vk_j$. Accordingly, in the unforgeability game, we can generate each public key $pk_i$ with a corresponding secret $sk_i$, and put these secret keys inside the witness extractable commitment first message. With this approach we encounter a new problem: the size of the commitment first message, and consequently the verifier's first message in the ZAP scheme, needs to scale at least linearly with the maximum number of members in a ring. However, the number of members in a ring can be arbitrarily large.

**One secret key, multiple public keys.** To overcome this problem we use a public key cryptosystem which can generate multiple public keys having a single secret key. Using such a public key scheme, the first commitment message now only needs to hold one short secret key. Luckily, public key cryptosystems with this property already exist in the literature. In particular, Regev's cryptosystem [43] already has the ability to generate multiple pseudorandom public keys having a single uniformly chosen secret key. This cryptosystem also has another appealing feature: it has sparse valid public keys. In other words, a randomly chosen public key does not have a corresponding secret key (except with negligible probability). In the ring signature context, this sparseness property will be helpful in proving the anonymity of the scheme. Specifically, given a signature $\Sigma$ for a ring $\mathsf{R}$, if at least one verification key in $\mathsf{R}$ is generated honestly, or in particular if at least one public key $pk_i \in \mathsf{R}$ is chosen uniformly at random, then $(\Sigma, \mathsf{R}) \notin \widetilde{L}$, and therefore, the commitment scheme is hiding.

**Compactness through malicious circuit private FHE.** The ZAP that we have constructed so far seems to satisfy the soundness and witness indistinguishibility properties that we need in the application to our ring signature scheme. However, upon a closer look, it seems that our resulting ZAP scheme suffers from a major flaw. Namely, the size of our ring signatures is linear in the size of the ring. This is because in the witness extractable commitment scheme, each commitment contains a garbling of a circuit that computes on each verification key in the ring separately. Specifically, while the size of the circuit for checking membership in $L$ is independent of the size of the ring (thanks to the properties of SPB hashing), circuits for checking membership in $\widetilde{L}$, and even the relevant statements, have size that depend on the ring. It seems that to overcome this, we need some form of a *compact* witness extractable commitment. Our final idea is to use a fully homomorphic encryption scheme [21] to build such a compact witness extractable commitment. The construction is as follows:

- The receiver generates a FHE key pair $(\mathsf{FHE}.pk, \mathsf{FHE}.sk)$ and sends a ciphertext
  $ct \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE}.pk, \widetilde{w})$ encrypting an arbitrary string $\widetilde{w}$.
- The sender homomorphically evaluates the following circuit on $ct$: on input $\widetilde{w}$, if $\widetilde{w}$ is a witness for $x \in \widetilde{L}$, output $b$, otherwise, output 0. The sender sends the evalauted ciphertext $ct_{eval}$.

Observe now that the compactness of the commitment scheme follows from the compactness of the FHE scheme. Clearly, if the receiver encrypts a non-witness for $x$, it can recover the committed bit $b$ using the secret key $\mathsf{FHE}.sk$. For this commitment scheme to be hiding, it is sufficient that in the FHE scheme, the evaluated ciphertext hides the circuit that has been evaluated on it, even if the initial FHE ciphertext and public key are maliciously generated. Fortunately, FHE schemes satisfying the aforementioned *malicious circuit privacy* have already been constructed from LWE [38,11].

### 1.4 Related Existing Work

The initial construction of ring signatures by Rivest, Shamir and Kalai [44] is in the random oracle model. Several subsequent constructions [1,9,27] were also given in the ROM. Constructions in the standard model were first obtained concurrently by Chow, Liu, Wei and Yuen [16] and Bender, Katz and Morselli [6]. Several works also construct schemes in the CRS model [46,10,45]. Brakerski and Kalai [12] gave a construction based on the SIS problem in the standard model, and there are subsequent lattice-based constructions [5,47] that give more practical constructions (these works actually construct *linkable* ring signatures). Park and Sealfon [39] give certain constructions based on SIS and others based on verifiable random functions that satisfy new and interesting definitions of repudiability and claimability that they develop. All of these constructions give ring signatures linear in the ring size.

Dodis et al [17] gave the first sublinear size ring signatures in the ROM. Since then, constructing logarithmic size ring signatures in the ROM been the

focus of many works [26,28,29,19,7,32]. In the CRS model, [15,23,24,30] build sublinear ring signatures under various assumptions. In the plain model, Backes et al [3] construct sublinear ring signatures using a primitive called signatures with *flexible* public key, and Malavolta and Schroder [33] construct constant size ring signatures under a knowledge of exponent assumption, which is unfalsifiable. Finally, as mentioned, Backes et al [2] construct the first logarithmic size ring signatures in the plain model under standard and falsifiable assumptions, namely DDH or LWE along with NIWI proofs.

## 2 Preliminaries

We denote the security parameter by $\lambda$. For any $\ell \in \mathbb{N}$, we denote the set of the first $\ell$ positive integers by $[\ell]$. For a set $S$, $x \leftarrow S$ denotes sampling a uniformly random element $x$ from $S$.

### 2.1 Learning With Errors

We recall the Learning With Errors (LWE) problem, and its hardness based on worst-case lattice problems.

For a positive integer dimension $n$ and modulus $q$, and an error distribution $\chi$ over $\mathbb{Z}$, the LWE distribution and decision problem are defined as follows. For an $\mathbf{s} \in \mathbb{Z}^n$, the LWE distribution $A_{\mathbf{s},\chi}$ is sampled by choosing a uniformly random $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ and an error term $e \leftarrow \chi$, and outputting $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e) \in \mathbb{Z}_q^{n+1}$.

**Definition 1.** *The decision-LWE$_{n,q,\chi}$ problem is to distinguish, with non-negligible advantage, between any desired (but polynomially bounded) number of independent samples drawn from $A_{\mathbf{s},\chi}$ for a single $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and the same number of* uniformly random *and independent samples over $\mathbb{Z}_q^{n+1}$.*

A standard instantiation of LWE is to let $\chi$ be a *discrete Gaussian* distribution over $\mathbb{Z}$ with parameter $r = 2\sqrt{n}$. A sample drawn from this distribution has magnitude bounded by, say, $r\sqrt{n} = \Theta(n)$ except with probability at most $2^{-n}$, and hence this tail of the distribution can be entirely removed. For this parameterization, it is known that LWE is at least as hard as *quantumly* approximating certain "short vector" problems on $n$-dimensional lattices, in the worst case, to within $\tilde{O}(q\sqrt{n})$ factors [43,41]. Classical reductions are also known for different parameterizations [40,13].

Fix a dimension $n = \text{poly}(\lambda)$. For the rest of this paper, when we refer to hardness of LWE, we mean hardness of LWE with polynomial modulus-to-noise ratio against polynomial sized adversaries, i.e., polynomial hardness of LWE$_{n,q,\chi}$ where, $q$ is a polynomial in $n$, and $\chi$ is the error distribution described in the previous paragraph.

### 2.2 Correlation Intractable Hash Functions

We borrow the following the definition of CI hash functions from [42] verbatim.

**Definition 2.** *Let $\mathcal{C} = \{\mathcal{C}_\lambda\}$ be a family of circuits, i.e., a set of circuits for each $\lambda$. A hash function family $\mathsf{Hash} = (\mathsf{Gen}, \mathsf{Eval})$ is* correlation intractable (CI) for $\mathcal{C}$ *if for every non-uniform polynomial-size adversary $\mathcal{A} = \{\mathcal{A}_\lambda\}$ there exists a negligible function $\nu(\lambda)$ such that for every $C \in \mathcal{C}_\lambda$,*

$$\Pr_{\substack{k \leftarrow \mathsf{Hash.Gen}(1^\lambda) \\ x \leftarrow \mathcal{A}_\lambda(k)}} [\mathsf{Hash.Eval}(k, x) = C(x)] \leq \nu(\lambda) \ . \tag{1}$$

We will also use CI hash construction of [42].

**Theorem 1 ([42]).** *Assuming hardness of LWE, there exists a polynomial $m = m(\lambda)$ such that, for every family of polynomial sized circuits $\mathcal{C}$ with output size at least $m$ bits, there exists a hash function family which is CI for $\mathcal{C}$. Furthermore, the key generation algorithm in this hash function family simply outputs a uniformly random key from its key space.*

### 2.3 Public Key Encryption

Similar to [2], we need a public key encryption scheme which has pseudorandom ciphertexts and public keys. For our application, we also require additional properties.

**Definition 3.** *A public key encryption scheme is a tuple of PPT algorithms $\mathsf{PKE} = (\mathsf{GenWithKey}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Valid})$, with the following interfaces, where for each security parameter $\lambda \in \mathbb{N}$, $\mathsf{PK}_\lambda$, $\mathsf{SK}_\lambda$ and $\mathsf{CT}_\lambda$ are three sets where the uniform distribution is efficiently sampleable,*

- $\mathsf{GenWithKey}(sk)$*, on input a secret key $sk \in \mathsf{SK}_\lambda$ outputs a public key $pk \in \mathsf{PK}_\lambda$.*
- $\mathsf{Enc}(pk, b)$*, on input a public key $pk$, and a message $b \in \{0,1\}$, outputs a ciphertext $ct \in \mathsf{CT}_\lambda$.*
- $\mathsf{Dec}(sk, ct)$*, on input a secret key $sk$ and a ciphertext $ct$, outputs a bit $b$.*
- $\mathsf{Valid}(pk, sk)$*, on input a public $pk$ and a secret key $sk$, either accepts or rejects.*

*We consider the following properties:*

1. Completeness*: for any $\lambda \in \mathbb{N}$, any key pair $(pk, sk)$ such that $\mathsf{Valid}(pk, sk)$ accepts, and any message $b$,*

$$\Pr[\mathsf{Dec}(sk, ct) = b] = 1,$$

*where $ct \leftarrow \mathsf{Enc}(pk, m)$. Furthermore, for any $\lambda \in \mathbb{N}$,*

$$\Pr_{\substack{sk \leftarrow \mathsf{SK}_\lambda \\ pk \leftarrow \mathsf{GenWithKey}(sk)}} [\mathsf{Valid}(pk, sk) \ \textit{accepts}] = 1.$$

2. Sparseness of valid public keys*: for any $\lambda \in \mathbb{N}$,*

$$\Pr_{pk \leftarrow \mathsf{PK}_\lambda} [\exists sk : \mathsf{Valid}(pk, sk) \ \textit{accepts}] = \mathrm{negl}(\lambda).$$

3. Injectivity of key generation*: for any $sk$,*

$$\Pr_{pk \leftarrow \mathsf{GenWithKey}(sk)}[\exists sk' \neq sk : \mathsf{Valid}(pk, sk') \ \textit{accepts}] = \mathrm{negl}(\lambda).$$

4. Pseudorandomness of public keys*: for every $Q = \mathrm{poly}(\lambda)$, the following two distributions*
   - *first, samples a uniformly random secret key $sk \leftarrow \mathsf{SK}_\lambda$, then outputs $\{pk_i \leftarrow \mathsf{GenWithKey}(sk)\}_{i\in[Q]}$*
   - *outputs $L$ uniformly random public keys $\{pk_i \leftarrow \mathsf{PK}_\lambda\}_{i\in[Q]}$,*

   *are computationally indistinguishable.*
5. Closeness of ciphertexts to uniform*: for every message $b$, the output of the following two distributions*
   - *first, samples a uniformly random public key $pk \leftarrow \mathsf{PK}_\lambda$, then, outputs $(pk, \mathsf{Enc}(pk, b))$,*
   - *first, samples a uniformly random public key $pk \leftarrow \mathsf{PK}_\lambda$, then, chooses a uniformly random ciphertext $ct \leftarrow \mathsf{CT}_\lambda$ and outputs $(pk, ct)$,*

   *are statistically indistinguishable.*

Consider Regev's public key cryptosystem [43]. For some appropriate parameters $n = \mathrm{poly}(\lambda), q = \mathrm{poly}(n), m \geq 2n \log q$, $B \ll q/4$, a secret key in this scheme is a vector $\mathbf{s} \in \mathbb{Z}_q^n$ and valid public keys for secret $\mathbf{s}$ are generated as

$$pk := \mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}} \\ \mathbf{s}^t \bar{\mathbf{A}} + \mathbf{e}^t \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times m},$$

where, $\bar{\mathbf{A}} \leftarrow \mathbb{Z}_q^{n\times m}$ and $\mathbf{e}$ is chosen from some $B$-bounded distribution. For this cryptosystem we define the following validity check algorithm

- $\mathsf{Valid}(pk = \mathbf{A}, sk = \mathbf{s})$: Accept iff $|(\mathbf{s}^t, -1) \cdot \mathbf{A}|_\infty \leq B$.

**Theorem 2.** *Assuming hardness of LWE, there exists a public key encryption scheme satisfying all the properties in Definition 3.*

*Proof.* We briefly argue that equipped with algorithm $\mathsf{Valid}$, Regev's cryptosystem satisfies all the properties in Definition 3. When $q$ is a prime number, injectivity of key generation is an implication of Lemma 5.3 in [22]. The rest of the properties are already established in [43,35].

### 2.4 Blum's Raw Protocol

Here, we formally define and state the properties of the raw version of Blum's sigma protocol [8]. In this abstraction, the prover does not use any commitment or encryption scheme to hide its first message and therefore, the protocol does not satisfy a conventional zero knowledge property. Using a commitment scheme, this protocol can be converted into an honest verifier zero knowledge protocol.

**Definition 4.** *Let $L \in \mathsf{NP}$ be a language with a corresponding relation $R$. Blum's raw protocol for $L$, is a tuple of PPT algorithms*
$\Pi = (\mathsf{P1}, \mathsf{P2}, \mathsf{V}, \mathsf{BadChallenge}, \mathsf{Sim})$ *with the following interfaces*

– $\mathsf{P1}(x, w)$, *on input a statement witness pair* $(x, w) \in L$, *it outputs a first message* $a$, *two substrings* $(a_0, a_1)$ *of a corresponding to two subsets of indices* $S_0, S_1$, *and an internal state* $\zeta$. *We implicitly assume* $(a_0, a_1)$ *uniquely determine the subsets* $S_0, S_1$.
– $\mathsf{P2}(a, b, \zeta)$, *on input first message* $a$, *a* challenge *bit* $b \in \{0, 1\}$, *and internal state* $\zeta$, *it outputs a second message* $c$.
– $\mathsf{V}(x, b, a_b, c)$, *on input an instance* $x$, *a challenge bit* $b \in \{0, 1\}$, *an* opening $a_b$, *and a response* $c$, *it either accepts or rejects.*
– $\mathsf{BadChallenge}(a)$, *on input a first message* $a$, *it outputs a badchallenge bit* $b \in \{0, 1\}$.
– $\mathsf{Sim}(b, x)$, *takes as input a challenge bit* $b$, *and an instance* $x$, *and outputs two strings* $a_b$ *and* $c$.

*These algorithms satisfy the following properties:*

1. Completeness*: for any* $(x, w) \in L$, *any* $\ell \in \mathbb{N}$, *and any* $b \in \{0, 1\}$,

$$\Pr[\mathsf{V}(x, b, a_b, c) \text{ accepts }] = 1,$$

   *where,* $(a, (a_0, a_1), \zeta) \leftarrow \mathsf{P1}(x, w)$, *and* $c \leftarrow \mathsf{P2}(a, b, \zeta)$.
2. Soundness*: if* $x \notin L$, *then, for any two subsets of indices* $S_0, S_1$, *bit* $b \neq \mathsf{BadChallenge}(a)$, *and any* $c$,

$$\Pr[\mathsf{V}(x, b, a_b, c) \text{ rejects }] = 1,$$

   *where,* $a_b$ *denotes a subset of* $a$ *specified by indices in* $S_b$.
3. Zero knowledge*: for any* $b \in \{0, 1\}$ *and any* $(x, w) \in R$ *the following two distributions,*
   – *outputs* $(b, a_b, c)$, *where* $(a, (a_0, a_1), \zeta) \leftarrow \mathsf{P1}(x, w)$, *and* $c \leftarrow \mathsf{P2}(a, b, \zeta)$
   – *outputs* $(b, \mathsf{Sim}(b, x))$
   *are identical.*

Blum's raw protocol exists unconditionally for any language $L \in \mathsf{NP}$ [8].

### 2.5 Maliciously Circuit Private FHE

We review the definition of maliciously circuit private FHE.

**Definition 5 ([38]).** *A maliciously circuit private leveled FHE scheme is a tuple of algorithms*
$\mathsf{FHE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Sim})$, *where, except for* $\mathsf{Sim}$ *the rest of the algorithms are PPT, having the following interfaces*

– $\mathsf{Gen}(1^\lambda, 1^d)$, *given a security parameter* $\lambda \in \mathbb{N}$ *and a depth parameter* $d \in \mathbb{N}$, *outputs a public and private key pair* $(pk, sk)$.
– $\mathsf{Enc}(pk, b)$, *given a public key* $pk$ *and a message* $b \in \{0, 1\}$, *outputs a ciphertext* $ct \in \{0, 1\}^{\ell_{ct}(\lambda, d)}$.

- $\mathsf{Eval}((ct_1, \cdots, ct_k), C; r)$, *given* $k$ *ciphertexts* $ct_1, \cdots, ct_k$, *a boolean circuit* $C : \{0,1\}^k \to \{0,1\}$, *and random coins* $r$, *outputs an evaluated ciphertext* $ct_{eval} \in \{0,1\}^{\ell_{eval}}$.
- $\mathsf{Dec}(sk, ct)$, *given a secret key* $sk$ *and a ciphertext* $ct$, *outputs a bit* $b \in \{0,1\}$.
- $\mathsf{Sim}(pk^*, (ct_1^*, \cdots, ct_k^*), b)$, *on input a (not necessarily honestly generated) public-key* $pk^*$ *and* $k$ *(not necessarily honestly generated) ciphertexts* $ct_1^* \in \{0,1\}^{\ell_{ct}(\lambda,d)}, ..., ct_k^* \in \{0,1\}^{\ell_{ct}(\lambda,d)}$, *and a bit* $b$, *outputs a simulated ciphertext* $ct_{sim}$.

*We consider FHE schemes that satisfy the following properties:*

1. Completeness*: for every* $\lambda, d \in \mathbb{N}$, *every circuit* $C : \{0,1\}^k \to \{0,1\}$ *of depth at most* $d$ *and every input* $m \in \{0,1\}^k$,

$$\Pr[\mathsf{Dec}(sk, ct_{eval}) = C(m)] = 1,$$

   *where,* $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda, 1^d)$, $ct_i \leftarrow \mathsf{Enc}(pk, m_i)$ *for every* $i \in [\ell]$, *and* $ct_{eval} \leftarrow \mathsf{Eval}((ct_1, \cdots, ct_\ell), C)$.
2. Compactness*: there exists fixed polynomials* $\ell_{eval} = \ell_{eval}(\lambda, d)$ *and* $\ell_{rand} = \ell_{rand}(\lambda, d)$, *such that evaluated ciphertexts have size* $\ell_{eval}(\lambda, d)$ *and the size of the randomness used in* $\mathsf{Eval}$ *algorithm is* $\ell_{rand}(\lambda, d)$, *i.e., the size of evaluated ciphertexts and the size of randomness in the evaluation algorithm only depend on the depth of the circuit being evaluated.*
3. Pseudorandomness of public keys*: the public key* $pk$ *output by the* $\mathsf{Gen}$ *algorithm is pseudorandom.*
4. Pseudorandomness of ciphertexts*: for every non-uniform polynomial-size adversary* $\mathcal{A}$, *every* $d \in \mathbb{N}$ *and every* $b \in \{0,1\}$, *the probabilities*

$$\Pr[\mathcal{A}(pk, ct) = 1], \tag{2}$$

   *in the following two experiments differ by only* $\mathrm{negl}(\lambda)$*:*
   - *in experiment 1,* $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda, 1^d)$, $ct \leftarrow \mathsf{Enc}(pk, b)$
   - *in experiment 2,* $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda, 1^d)$, $ct \leftarrow \{0,1\}^{\ell_{ct}(\lambda,d)}$
5. Malicious circuit privacy*: for every (not necessarily honestly generated) public-key* $pk^*$, *every* $k \in \mathbb{N}$, *and every* $k$ *(not necessarily honestly generated) ciphertexts* $ct_1^* \in \{0,1\}^{\ell_{ct}(\lambda,d)}, ..., ct_k^* \in \{0,1\}^{\ell_{ct}(\lambda,d)}$, *there exists a* $m^* \in \{0,1\}^k$ *such that, for every circuit* $C : \{0,1\}^k \to \{0,1\}$ *of depth at most* $d$,

$$\mathsf{Eval}((ct_1^*, \cdots, ct_k^*), C) \stackrel{s}{\approx} \mathsf{Sim}(pk^*, (ct_1^*, \cdots, ct_k^*), C(m^*))$$

   .

**Theorem 3 ([38,11]).** *Assuming hardness of LWE, there exists a maliciously circuit private leveled FHE, where the size of evaluated ciphertexts and secret keys only depend on the security parameter* $\lambda$.

### 2.6 Somewhere Perfectly Binding Hash

We next define the notion of somewhere perfectly binding hash or SPB schemes, which are very similar to somewhere statistically binding hash schemes (as can be surmised, the only change from the latter is that here we expect the somewhere binding property to hold with probability 1). As in the scheme of [2], we will only define and employ a slightly weaker primitive denoted as *somewhere perfectly binding hash with private local openings*, which is what we will need for our signature scheme as well. We direct the reader to [2] for further details. The definition is essentially identical to that in [2], and is as follows.

**Definition 6 (SPB Hash).** *A somewhere perfectly binding hash scheme with private local openings, denoted by* $\mathsf{SPB}$*, consists of a tuple of probabilistic polynomial time algorithms* $(\mathsf{Gen}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Verify})$*, with the following syntax:*

- $\mathsf{Gen}(1^\lambda, n, \mathsf{ind})$*, given a security parameter* $\lambda$*, a database size* $n$*, and index* $\mathsf{ind}$ *as input, outputs a hash and secret key pair* $(hk, shk)$*.*
- $\mathsf{Hash}(\mathsf{hk}, \mathsf{db})$*, given a hash key* $hk$ *and database* $\mathsf{db}$ *as input, outputs a hash value* $h$*.*
- $\mathsf{Open}(hk, shk, \mathsf{db}, \mathsf{ind})$*, given a hash key* $hk$*, secret key* $shk$*, database* $\mathsf{db}$ *and index* $\mathsf{ind}$ *as input, outputs a witness* $\tau$*.*
- $\mathsf{Verify}(hk, h, \mathsf{ind}, x, \tau)$*, given as input a hash key* $hk$*, a hash value* $h$*, an index* $\mathsf{ind}$*, a value* $x$ *and a witness* $\tau$*, either accepts or rejects.*

*To maintain clarity, we will not explicitly specify the block size of databases as input to* $\mathsf{Gen}$*, but assume that this is clear from the specific usage and hardwired into the algorithm. We ask that the* $\mathsf{SPB}$ *scheme satisfies the following properties:*

1. Correctness: *for all* $\lambda \in \mathbb{N}$*,* $n = \mathrm{poly}(\lambda)$*, all databases* $\mathsf{db}$ *of size* $n$*, and all indices* $\mathsf{ind} \in [n]$*, we have,*

$$\Pr[\mathsf{Verify}(hk, h, \mathsf{ind}, \mathsf{db}_{\mathsf{ind}}, \tau) \text{ accepts }] = 1,$$

   *where,* $(hk, shk) \leftarrow \mathsf{Gen}(1^\lambda, n, \mathsf{ind})$*,* $h \leftarrow \mathsf{Hash}(hk, \mathsf{db})$ *and* $\tau \leftarrow \mathsf{Open}(hk, shk, \mathsf{db}, \mathsf{ind})$*.*
2. Efficiency: *any hash keys* $\mathsf{hk}$ *and witnesses* $\tau$ *corresponding to size* $n$ *databases, are of size* $\log(n) \cdot \mathrm{poly}(\lambda)$*. Further, for size* $n$ *databases,* $\mathsf{Verify}$ *can be computed by a circuit of size* $\log(n) \cdot \mathrm{poly}(\lambda)$*.*
3. Somewhere perfectly binding: *for all* $\lambda, n \in \mathbb{N}$*, all databases* $\mathsf{db}$ *of size* $n$*, all indices* $\mathsf{ind} \in [n]$*, all purported hashing keys* $hk$*, all purported witnesses* $\tau$*, all* $h$ *in the support of* $\mathsf{Hash}(hk, \mathsf{db})$*, if* $\mathsf{Verify}(hk, h, \mathsf{ind}, x, \tau)$ *accepts, then* $x = \mathsf{db}_{\mathsf{ind}}$*.*
4. Index hiding, *for any* $n \in \mathbb{N}$ *and any* $\mathsf{ind}_0, \mathsf{ind}_1 \in [n]$*,*

$$\{hk : (hk, shk) \leftarrow \mathsf{Gen}(1^\lambda, n, \mathsf{ind}_0)\} \overset{c}{\approx} \{hk : (hk, shk) \leftarrow \mathsf{Gen}(1^\lambda, n, \mathsf{ind}_1)\}$$

**Theorem 4 ([2]).** *Assuming hardness of LWE, there exists a SPB hash.*

### 2.7 Ring Signatures

We review the definition of compact ring signatures as presented in [2].

**Definition 7 (Compact Ring Signature [2]).** *A compact ring signature scheme* $\mathsf{RS}$ *is described by a triple of PPT algorithms* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *that have the following interface:*

- $\mathsf{Gen}(1^\lambda, N)$*, given a security parameter* $\lambda$ *and the maximum number of members in a ring* $N$*, outputs a verification and signing key pair* $(\mathsf{VK}, \mathsf{SK})$.
- $\mathsf{Sign}(\mathsf{SK}, m, \mathsf{R})$*, given a secret key* $\mathsf{SK}$*, a message* $m \in \mathcal{M}_\lambda$*. and a list of verification keys (interpreted as a ring)* $\mathsf{R} = (\mathsf{VK}_1, \cdots, \mathsf{VK}_\ell)$ *as input, outputs a ring signature* $\Sigma$.
- $\mathsf{Verify}(\mathsf{R}, m, \Sigma)$*, given a ring* $\mathsf{R} = (\mathsf{VK}_1, \ldots, \mathsf{VK}_\ell)$*, message* $m \in \mathcal{M}_\lambda$ *and ring signature* $\Sigma$ *as input, either accepts or rejects.*

*Further, we require that the ring signature satisfies the following properties:*

1. Completeness: *for all* $\lambda \in \mathbb{N}$*,* $N \in \mathbb{N}$*,* $\ell \leq N$*,* $i \in [\ell]$ *and* $m \in \mathcal{M}_\lambda$*, we have:*

$$\Pr[\mathsf{RS.Verify}(\mathsf{R}, m, \Sigma) \text{ accepts}] = 1,$$

   *where,* $(\mathsf{VK}_\mathsf{i}, \mathsf{SK}_\mathsf{i}) \leftarrow \mathsf{Gen}(1^\lambda, N)$*,* $\Sigma \leftarrow \mathsf{Sign}(\mathsf{SK}_\mathsf{i}, m, \mathsf{R})$ *(where* $\mathsf{R} = (\mathsf{VK}_1, \ldots, \mathsf{VK}_\ell)$*).*
2. Unforgeability: *for any* $N \in \mathbb{N}$*, and any* $Q = \mathrm{poly}(\lambda)$*, any PPT adversary* $\mathcal{A}$ *has negligible advantage in the following game:*

   **Experiment** $\mathsf{RS}{-}\mathsf{Forge}^\mathsf{Q}(\mathcal{A})$**:** *This experiment is run by a challenger that proceeds as follows:*
   - *For each* $i \in [Q]$*, the challenger generates key pairs* $(\mathsf{VK}_\mathsf{i}, \mathsf{SK}_\mathsf{i}) \leftarrow \mathsf{Gen}(1^\lambda, N; r_i)$*, and stores these key pairs along with their corresponding randomness. It then sets* $\mathcal{VK} = \{\mathsf{VK}_1, \ldots, \mathsf{VK}_\mathsf{Q}\}$ *and initializes* $\mathcal{C} = \emptyset$.
   - *The challenger sends* $\mathcal{VK}$ *to* $\mathcal{A}$.
   - $\mathcal{A}$ *can now make the following two kinds of queries: signing queries* $\mathsf{sign}$ *to get signatures signed by a particular entity on a particular message with respect to a ring of its choice, and corruption queries* $\mathsf{corrupt}$ *to corrupt a particular entity. The challenger responds as follows:*
     - *Signing query* $(\mathsf{sign}, i, m, \mathsf{R})$*: The challenger first checks if* $\mathsf{VK}_i \in \mathsf{R}$*. If so, it computes* $\Sigma \leftarrow \mathsf{Sign}(\mathsf{SK}_\mathsf{i}, m, \mathsf{R})$ *and sends this to* $\mathcal{A}$*. It also keeps a list of all such queries made by* $\mathcal{A}$.
     - *Corruption query* $(\mathsf{corrupt}, i)$*: The challenger adds* $\mathsf{VK}_i$ *to the set* $\mathcal{C}$ *and returns the randomness* $r_i$ *to* $\mathcal{A}$.
   - *Finally,* $\mathcal{A}$ *outputs a forgery attempt, namely a purported ring signature* $\Sigma^*$ *with respect to a ring* $\mathsf{R}^*$ *and message* $m^*$*. The challenger checks if:*
     - $\mathsf{R}^* \subseteq \mathcal{VK} \setminus \mathcal{C}$*,*
     - $\mathcal{A}$ *never made a signing query with respect to* $m^*$ *and* $\mathsf{R}^*$ *(together, i.e. of the form* $(\mathsf{sign}, \cdot, m^*, \mathsf{R}^*)$*), and*
     - $\mathsf{Verify}(\mathsf{R}^*, m^*, \Sigma^*)$ *accepts.*

     *If so, the challenger outputs 1, otherwise, it outputs 0.*

*The* advantage *of the adversary* $\mathcal{A}$ *is defined to be* $\mathbf{Adv}_{\mathsf{RS-Forge}^Q}(\mathcal{A}) = \Pr[\mathsf{RS-Forge}^Q(\mathcal{A}) = 1]$.

3. Anonymity: *for all* $Q = \mathrm{poly}(\lambda)$, *any PPT adversary* $\mathcal{A}$ *has negligible advantage in the following game:*

   **Experiment** $\mathsf{RS-Anon}^Q(\mathcal{A})$**:** *This experiment is run by a challenger that proceeds as follows:*
   - *For each* $i \in [Q]$, *the challenger generates key pairs* $(\mathsf{VK_i}, \mathsf{SK_i}) \leftarrow \mathsf{Gen}(1^\lambda; r_i)$. *It sends these key pairs along with their corresponding randomness to* $\mathcal{A}$.
   - *Eventually* $\mathcal{A}$ *sends a challenge to the challenger of the form* $(\mathsf{R}, m, i_0, i_1)$. *We stress that* $\mathsf{R}$ *might have keys that are not generated by the challenger in the previous step. In particular, it might contain maliciously generated keys. The challenger checks if* $\mathsf{VK}_{i_0}, \mathsf{VK}_{i_1} \in \mathsf{R}$. *If so, it first samples a uniform bit* $b \leftarrow \{0,1\}$, *then computes* $\Sigma \leftarrow \mathsf{Sign}(\mathsf{SK}_{i_b}, m, \mathsf{R})$, *and sends this to* $\mathcal{A}$.
   - $\mathcal{A}$ *sends back its guess bit* $b'$. *The challenger outputs 1 if* $b' = b$, *otherwise it outputs 0.*

   *The* advantage *of the adversary* $\mathcal{A}$ *in this game is defined as*

$$\mathbf{Adv}_{\mathsf{RS-Anon}^Q}(\mathcal{A}) = |\Pr[\mathsf{RS-Anon}^Q(\mathcal{A}) = 1] - \frac{1}{2}|.$$

4. Compactness: *the size of a signature is upper-bounded by a polynomial in* $\lambda$ *and* $\log N$.

We mention that the unforgeability and anonymity properties defined in Definition 7 correspond respectively to the notions of *unforgeability with insider corruption* and *anonymity with respect to full key exposure* presented in [6].

## 3 Compact Witness Extractable Commitments

In this section we define witness extractable commitments. A witness extractable commitment for a language $L \in \mathsf{NP}$ with corresponding $\mathsf{NP}$ relation $R$ is a two round commitment protocol. The receiver's message is generated using a witness $w$, and the sender's commitment to a bit $b$ is generated using a statement $x$. Informally speaking, the bit $b$ can be efficiently extracted when $(x, w) \in R$, however, when $x \notin L$, $b$ is statistically hidden.

For our application in this paper, we are interested in witness extractable commitments that are compact. This means that the size of a commitment second message does not depend on the size of the $\mathsf{NP}$ verifier circuit (except for maybe its depth).

### 3.1 Definition

**Definition 8.** *Fix a language* $L \in \mathsf{NP}$. *By* $R$ *and* $\{C_{n,\ell} : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}\}_{n,\ell \in N}$ *denote the* $\mathsf{NP}$ *relation and the* $\mathsf{NP}$ *verification circuit corresponding*

*to $L$ respectively. Also, let $d = d(n, \ell)$ be the depth of $C_{n,\ell}$. When it is clear from the context, we use $d$ instead of $d(n, \ell)$. A* witness extractable commitment scheme *for $L$, is a tuple of PPT algorithms* $(\mathsf{Com1}, \mathsf{Com2}, \mathsf{Verify}, \mathsf{Extract})$ *having the following interfaces:*

- $\mathsf{Com1}(1^\lambda, 1^D, w)$*, given a security parameter $\lambda$, circuit depth upper bound $D$, and a witness $w \in \{0,1\}^\ell$, outputs the commitment first message* $\mathsf{com1} \in \{0,1\}^{\ell_{com1}=\ell_{com1}(\lambda,D,\ell)}$ *and a string* $\mathsf{st} \in \{0,1\}^{\ell_{st}}$ *representing the internal state.*
- $\mathsf{Com2}(\mathsf{com1}, x, b; r)$*, given a commitment first message* $\mathsf{com1}$*, a statement $x$, a bit $b$ to commit, and randomness $r \in \{0,1\}^{\ell_r}$, outputs a commitment* $\mathsf{com2} \in \{0,1\}^{\ell_{com2}}$*.*
- $\mathsf{Verify}(\mathsf{com1}, \mathsf{com2}, x, b, r)$*, given a commitment transcript* $\mathsf{com1}, \mathsf{com2}$*, a statement $x$, a bit $b$, and random coins $r$, it either accepts or rejects.*
- $\mathsf{Extract}(\mathsf{com2}, \mathsf{st})$*, given a commitment* $\mathsf{com2}$ *and internal state* $\mathsf{st}$*, outputs a bit $b$.*

*We consider the following properties:*

1. Completeness*: for every $\lambda \in \mathbb{N}$, bit $b$, every statement $x$, every witness $w$, every $D \geq d$,*

$$\Pr[\mathsf{Verify}(\mathsf{com1}, \mathsf{com2}, x, b, r) \text{ accepts }] = 1,$$

   *where,* $\mathsf{com1} \leftarrow \mathsf{Com1}(1^\lambda, 1^D, w)$*, $r \leftarrow \{0,1\}^{\ell_r}$, and* $\mathsf{com2} \leftarrow \mathsf{Com2}(\mathsf{com1}, x, b; r)$*.*
2. Statistical hiding*: if $x \notin L$, then, for any $\ell \in \mathbb{N}$, any $D \geq d(|x|, \ell)$, and any sequence of strings* $\{\mathsf{com1}_\lambda \in \{0,1\}^{\ell_{com1}(\lambda,D,\ell)}\}_{\lambda\in\mathbb{N}}$*,*

$$\mathsf{com1}_\lambda, \mathsf{Com2}(\mathsf{com1}_\lambda, x, 0) \overset{s}{\approx} \mathsf{com1}_\lambda, \mathsf{Com2}(\mathsf{com1}_\lambda, x, 1) \tag{3}$$

3. Pseudorandomness of first message*: for any $w$ and any $D$,*

$$\mathsf{Com1}(1^\lambda, 1^D, w) \overset{c}{\approx} U_{\ell_{com1}} \tag{4}$$

4. Extractability*: if $(x, w) \in R$, then, for any bit $b$, any $D \geq d$, any* $\mathsf{com1}, \mathsf{st}$ *in the support of* $\mathsf{Com1}(1^\lambda, 1^D, w)$*, any randomness $r$, and any* $\mathsf{com2}$ *such that* $\mathsf{Verify}(\mathsf{com1}, \mathsf{com2}, x, b, r)$ *accepts,*

$$\Pr[\mathsf{Extract}(\mathsf{com2}, \mathsf{st}) = b] = 1. \tag{5}$$

5. Compactness*: the parameters $\ell_{com2}$,$\ell_r$ and $\ell_{st}$ are upper-bounded by some language-independent fixed polynomials in $\lambda$ and $D$. In particular, they are independent of the size of the* NP *verifier circuit $C_{n,\ell}$ and the size of the statement $x$.*

### 3.2 Construction

We will use the maliciously circuit private FHE scheme $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval})$ of [38].

**Construction 1.** Here we use the same notation as in Definition 8.

- $\mathsf{Com1}(1^\lambda, 1^D, w)$: run $(\mathsf{FHE.pk}, \mathsf{FHE.sk}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^{D+1})$. Output $\mathsf{com1} \leftarrow \mathsf{FHE.Enc}(\mathsf{FHE.pk}, w)$. Keep $\mathsf{st} := \mathsf{FHE.sk}$ as internal state. Notice that for any circuit $C$ of depth $D$, the circuit constructed in Figure 1 has depth $D+1$.
- $\mathsf{Com2}(\mathsf{com1}, x, b)$: on input first message $\mathsf{com1}$ and bit $b$, output

$$\mathsf{com2} \leftarrow \mathsf{FHE.Eval}(\mathsf{com1}, G_{x,b}; r),$$

  where, $G_{x,b}$ is the circuit defined below and $r$ represents the random coins used in the FHE evaluation algorithm.
- $\mathsf{Extract}(\mathsf{com2}, \mathsf{st} = \mathsf{FHE.sk})$: output $\mathsf{FHE.Dec}(\mathsf{FHE.sk}, \mathsf{com2})$.
- $\mathsf{Verify}(\mathsf{com1}, \mathsf{com2}, x, b, r)$: accept iff $\mathsf{com2}$ is equal to $\mathsf{FHE.Eval}(\mathsf{com1}, G_{x,b}; r)$.

**procedure** $G_{x,b}(w)$
  **if** $C(x, w) = 1$ **then**
    Output $b$
  **else**
    Output 0

**Fig. 1:** Description of $G_{x,b}$

Extractability and completeness of Construction 1 follow immediately from completeness of FHE. Compactness also follows from compactness of FHE. In fact, using the FHE in [38], both $\ell_{st}$ and $\ell_{com2}$ only depend on $\lambda$. Finally, pseudorandomness of FHE ciphertext and public keys imply pseudorandomness of the first message in Construction 1.

**Theorem 5.** *If* FHE *is maliciously circuit private, then, the commitment scheme in Construction 1 is statistically hiding.*

## 4 Compact Relaxed ZAPs for Extended NP ∩ coNP

In this section, we define and construct a 2-round public-coin argument system. Our argument system can be viewed as a generalization of ZAPs for NP∩coNP. To describe this generalization, first we introduce the notion of *super-complement* of a language. A super-complement of a language is an extension of the complement of that language. Notably, the complement of a language is a super-complement of it.

**Definition 9 (Super-Complement).** *Let $L$, $\widetilde{L}$ be two languages where the elements of $\widetilde{L}$ are represented as pairs of bit strings. We say $\widetilde{L}$ is a* super-complement *of $L$, if*

$$\widetilde{L} \subseteq (\{0,1\}^* \setminus L) \times \{0,1\}^*,$$

*i.e., $\widetilde{L}$ is a super complement of $L$ if for any $x = (x_1, x_2)$,*

$$x \in \widetilde{L} \Rightarrow x_1 \notin L.$$

Our argument system is defined for two $\mathsf{NP}$ languages $L, \widetilde{L}$, where, $\widetilde{L}$ is a super-complement of $L$. Notice that, while the complement of $L$ might not be in $\mathsf{NP}$, however we require that $\widetilde{L} \in \mathsf{NP}$. The language $\widetilde{L}$ is used to define the soundness property. Namely, producing a proof for a statement $x = (x_1, x_2) \in \widetilde{L}$, should be hard. We also use the fact that $\widetilde{L} \in \mathsf{NP}$ to mildly strengthen the soundness property. In more detail, instead of having selective soundness where the statement $x \in \widetilde{L}$ is fixed in advance, now, we fix a non-witness $\widetilde{w}$ and let the statement $x$ be adaptively chosen by the malicious prover from all statements which have $\widetilde{w}$ as a witness to their membership in $\widetilde{L}$.

For our application to compact ring signatures, we further require the size of the proofs to be compact with respect to $\widetilde{L}$. Roughly speaking, this means that size of a proof for a statement $x = (x_1, x_2)$ only depends on the size of $x_1$.

### 4.1 Definition

**Definition 10.** *Let $L, \widetilde{L} \in \mathsf{NP}$ be two languages such that $\widetilde{L}$ is a super complement of $L$. By $R$ and $\widetilde{R}$ denote the $\mathsf{NP}$ relations corresponding to $L$ and $\widetilde{L}$ respectively. Let $\{C_{n,\ell}\}_{n,\ell \in N}$ and $\{\widetilde{C}_{n,\ell}\}_{n,\ell \in N}$ be the $\mathsf{NP}$ verification circuits for $L$ and $\widetilde{L}$ respectively. Let $\widetilde{d} = \widetilde{d}(n,\ell)$ be the depth of $\widetilde{C}_{n,\ell}$. A* compact relaxed ZAP *for $L, \widetilde{L}$ is a tuple of PPT algorithms $(\mathsf{V}, \mathsf{P}, \mathsf{Verify})$ having the following interfaces (where $1^n, 1^\lambda$ are implicit inputs to $\mathsf{P}$, $\mathsf{Verify}$):*

- $\mathsf{V}(1^\lambda, 1^n, 1^{\widetilde{\ell}}, 1^{\widetilde{D}})$*, given a security parameter $\lambda$, statement length $n$ for $L$, witness length $\widetilde{\ell}$ for $\widetilde{L}$, and $\mathsf{NP}$ verifier circuit depth upper-bound $\widetilde{D}$ for $\widetilde{L}$, outputs a first message $\rho$.*
- $\mathsf{P}(\rho, x = (x_1, x_2), w)$*, given a string $\rho$, a statement $(x_1 \in \{0,1\}^n, x_2)$, and a witness $w$ such that $(x_1, w) \in R$, outputs a proof $\pi$.*
- $\mathsf{Verify}(\rho, x = (x_1, x_2), \pi)$*, given a string $\rho$, a statement $x$, and a proof $\pi$, either accepts or rejects.*

*We consider the following properties:*

1. Completeness*: for every $(x_1, w) \in L$, every $x_2 \in \{0,1\}^*$, every $\widetilde{\ell} \in \mathbb{N}$, every $\widetilde{D} \geq \widetilde{d}(|x_1| + |x_2|, \widetilde{\ell})$, and every $\lambda \in \mathbb{N}$,*

$$\Pr[\mathsf{Verify}(\rho, x = (x_1, x_2), \pi) \text{ accepts }] = 1,$$

*where, $\rho \leftarrow \mathsf{V}(1^\lambda, 1^{|x_1|}, 1^{\widetilde{\ell}}, 1^{\widetilde{D}})$ and $\pi \leftarrow \mathsf{P}(\rho, x, w)$.*

2. Public coin: $\mathsf{V}(1^\lambda, 1^n, 1^{\widetilde{\ell}}, 1^{\widetilde{D}})$ *simply outputs a uniformly random string.*
3. Selective non-witness and adaptive statement soundness*: for every non-uniform polynomial-size "cheating" prover* $P^* = \{P^*_\lambda\}$ *there exists a negligible function* $\nu(\lambda)$ *such that for any* $n, \widetilde{D} \in \mathbb{N}$ *and any non-witness* $\widetilde{w} \in \{0,1\}^*$,

$$\Pr_{\substack{\rho \leftarrow \mathsf{V}(1^\lambda, 1^n, 1^{|\widetilde{w}|}, 1^{\widetilde{D}}) \\ (x=(x_1,x_2), \pi^*) \leftarrow P^*_\lambda(\rho)}} [\mathsf{Verify}(\rho, x, \pi^*) \text{ accepts} \wedge \widetilde{D} \geq \widetilde{d}(|x|, |\widetilde{w}|) \wedge (x, \widetilde{w}) \in \widetilde{R}] \leq \nu(\lambda). \tag{6}$$

4. Statistical witness indistinguishability*: for every (possibly unbounded) "cheating" verifier* $V^* = (V^*_1, V^*_2)$*, and every* $n, \widetilde{\ell}, \widetilde{D} \in \mathbb{N}$ *the probabilities*

$$\Pr[V^*_2(\rho, x, \pi, \zeta) = 1 \wedge (x, w) \in \mathcal{R} \wedge (x, w') \in \mathcal{R}]$$

*in the following two experiments differ only by* $\mathrm{negl}(\lambda)$*:*
   - *in experiment 1,* $(\rho, x, w, w', \zeta) \leftarrow V^*_1(1^\lambda, 1^n, 1^{\widetilde{\ell}}, 1^{\widetilde{D}}), \pi \leftarrow \mathsf{P}(\rho, x, w)$
   - *in experiment 2,* $(\rho, x, w, w', \zeta) \leftarrow V^*_1(1^\lambda, 1^n, 1^{\widetilde{\ell}}, 1^{\widetilde{D}}), \pi \leftarrow \mathsf{P}(\rho, x, w')$

5. Compactness*: bit-size of proof* $\pi$ *is a fixed polynomial in* $n, \widetilde{\ell}, \widetilde{D}, |C|$ *and* $\lambda$*. In particular, it is independent of the size of* $\widetilde{C}$ *and* $x_2$*.*

### 4.2 Construction

For languages $L$, $\widetilde{L}$, we give the tuple of algorithms $(\mathsf{V}, \mathsf{P}, \mathsf{Verify})$ that make up our relaxed ZAP scheme. In the construction we will use the following ingredients:

- A witness extractable commitment $\mathsf{Com} = (\mathsf{Com1}, \mathsf{Com2}, \mathsf{Verify}, \mathsf{Extract})$ for $\widetilde{L}$. We denote the sizes of the first commitment message, second commitment message, the internal state output by $\mathsf{Com1}$, and the randomness for $\mathsf{Com2}$, by $\ell_{com1}$ ,$\ell_{com2}$, $\ell_{st}$ and $\ell_r$ respectively.
- Blum's raw protocol $\Pi = (\mathsf{P1}, \mathsf{P2}, V, \mathsf{BadChallenge}, \mathsf{Sim})$ for $L$. We denote the size of the first and second prover messages by $\ell_{p1}$ and $\ell_{p2}$ respectively. For any $\ell \in N$, $\Pi^\ell$ means repeating the protocol $\Pi$, $\ell$ times in parallel and interpreting the inputs to the algorithms accordingly. When it is clear from the context, we drop the superscript $\ell$.
- A hash family $\mathsf{Hash}^{n,\ell_{rep}} = (\mathsf{Gen}, \mathsf{Eval})$ that for any $n \in \mathbb{N}$, and any polynomial $\ell_{rep} = \ell_{rep}(\lambda)$ that is larger than the polynomial $m(\lambda)$ in Theorem 1, is CI for a circuit family $\mathcal{C}^{n,\ell_{rep}}$ which we will define next. The circuit family is defined as

$$\mathcal{C}^{n,\ell_{rep}} = \{\mathcal{C}^{n,\ell_{rep}}_\lambda\}_{\lambda \in \mathbb{N}},$$

where for each $\lambda \in \mathbb{N}$,

$$\mathcal{C}^{n,\ell_{rep}}_\lambda = \{f_{st} : \{0,1\}^{\ell_{rep} \cdot \ell_{p1} \cdot \ell_{com2}} \to \{0,1\}^{\ell_{rep}}\}_{st \in \{0,1\}^{\ell_{st}(\lambda)}},$$

where, $f_{st}$ is defined as

$$f_{st}(x) = \Pi^{\ell_{rep}}.\mathsf{BadChallenge}(\mathsf{Com.Extract}(x, st)),$$

i.e., $f_{st}$ extracts a message from the input commitment and outputs the *bad challenge* corresponding to that message. We will drop the indices $n, \ell_{rep}$ when they are clear from the context.

**Construction 2.** Let $\ell_{rep} = \ell_{rep}(\lambda)$ be a polynomial that is larger than the polynomial $m(\lambda)$ in Theorem 1.

- $\mathsf{V}(1^\lambda, 1^n, 1^{\widetilde{\ell}}, 1^{\widetilde{D}})$: first, pick a uniformly random commitment first message $\mathsf{com1} \leftarrow \{0,1\}^{\ell_{com1}(\lambda,\widetilde{D},\widetilde{\ell})}$, then, generate CI-hash key $k \leftarrow \mathsf{Hash}^{n,\ell_{rep}}.\mathsf{Gen}(1^\lambda)$. Output $\rho := (\mathsf{com1}, k)$.
- $\mathsf{P}(\rho = (\mathsf{com1}, k), x = (x_1, x_2), w)$: first, compute

$$(\mathbf{a}, \{a_{i,b}\}_{i\in[\ell_{rep}],b\in\{0,1\}}, \zeta) \leftarrow \Pi^{\ell_{rep}}.\mathsf{P1}(x_1, w),$$

  and then send

$$\pi = (\mathsf{com2} = \mathsf{Com.Com2}(\mathsf{com1}, x, \mathbf{a}; \mathbf{r}), I = \mathsf{Hash.Eval}(k, \mathsf{com2}), \mathbf{c} = \Pi.\mathsf{P2}(\mathbf{a}, I, \zeta), \mathbf{r}_I, \mathbf{a}_I)$$

  to the verifier, where, $\mathbf{a}_I = \{a_{i,I_i}\}_{i\in[\ell_{rep}]}$ and $\mathbf{r}_I$ denotes the subset of randomness used to commit to $\mathbf{a}_I$. Also, $\mathsf{com2}_I$ denotes the chunks of $\mathsf{com2}$ which commit to $\mathbf{a}_I$.
- $\mathsf{Verify}(\rho, x = (x_1, x_2), \pi)$: parse $\pi = (\mathsf{com2},\ I,\ \mathbf{c},\ \mathbf{r}_I,\ \mathbf{a}_I)$. Accept iff both $\Pi.\mathsf{V}(x, I, \mathbf{a}_I, \mathbf{c})$ and $\mathsf{Com.Verify}(\mathsf{com1}, \mathsf{com2}_I, x, \mathbf{a}_I, \mathbf{r}_I)$ accept.

Completeness of Construction 2 follows directly from the completeness of $\Pi$ and $\mathsf{Com}$. It is also public coin because the CI hash keys are uniform. Compactness also follows from the compactness of $\mathsf{Com}$, namely, it follows from the fact that $\ell_{com2}$, $\ell_{st}$ and $\ell_r$ may only polynomially depend on the depth $\widetilde{D}$ of $\widetilde{C}$ and not its size.

**Theorem 6.** *The protocol described in Construction 2 satisfies selective non-witness adaptive statement soundness.*

**Theorem 7.** *The protocol described in Construction 2 is statistically witness indsitinguishable.*

## 5 Compact LWE-based Ring Signature Scheme

In this section, we present our compact ring signature scheme. First, we briefly list the ingredients in our construction:

- A standard signature scheme $\mathsf{Sig}$= ($\mathsf{Gen}$, $\mathsf{Sign}$, $\mathsf{Verify}$) with $\mathsf{EUF-CMA}$ security.
- A public key encryption scheme $\mathsf{PKE}$ = ($\mathsf{GenWithKey}$, $\mathsf{Enc}$, $\mathsf{Dec}$, $\mathsf{Valid}$) as defined in Definition 3.
- A somewhere perfectly binding hash function $\mathsf{SPB}$ = ($\mathsf{Gen}$, $\mathsf{Hash}$, $\mathsf{Open}$, $\mathsf{Verify}$) with private local openings.
- A compact relaxed ZAP scheme $\mathsf{ZAP}$ = ($\mathsf{V}$, $\mathsf{P}$, $\mathsf{Verify}$) as described in section 4.

Next, we define the languages $L$ and $\widetilde{L}$ that we will instantiate our relaxed ZAP construction for. The language $L$ is identical to the language $\mathcal{L}$ used in the

ring signature construction of [2]. For a statement $y_1 = (m, c, hk, h)$ and witness $w = (\mathsf{VK} = (vk, pk, \rho), i, \tau, \sigma, r_c)$, define relations $R_1, R_2$ and $R_3$ as follows:

$$(y_1, w) \in R_1 \Leftrightarrow \mathsf{SPB.Verify}(hk, h, i, \mathsf{VK}, \tau) \text{ accepts}$$
$$(y_1, w) \in R_2 \Leftrightarrow \mathsf{PKE.Enc}(pk, (\sigma, vk); r_c) = c$$
$$(y_1, w) \in R_3 \Leftrightarrow \mathsf{Sig.Verify}(vk, m, \sigma) \text{ accepts}$$

Next, define the relation $R'$ as

$$R' := R_1 \cap R_2 \cap R_3.$$

Let $L'$ be the language corresponding to $R'$. For statements of the form $(m, c_1, c_2, hk_1, hk_2, h_1, h_2)$, define the language $L$ as

$$L = \{(m, c_1, c_2, hk_1, hk_2, h_1, h_2) | (m, c_1, hk_1, h_1) \in L' \vee (m, c_2, hk_2, h_2) \in L'\}.$$

Now, we define the language $\widetilde{L}$ and prove that it is a super-complement of $L$. Let $x_2 = \mathsf{R} = (\mathsf{VK}_1, \ldots, \mathsf{VK}_\ell)$, $y = (y_1, x_2)$, and $\widetilde{w} = s$. Define the following relations:

$$(y, \widetilde{w}) \in R_4 \Leftrightarrow \forall j \in [\ell] : \mathsf{PKE.Valid}(pk_j, s) \text{ accepts} \wedge h = \mathsf{SPB.Hash}(hk, \mathsf{R})$$
$$(y, \widetilde{w}) \in R_5 \Leftrightarrow \mathsf{PKE.Dec}(s, c) = (\sigma, vk) \wedge \mathsf{Sig.Verify}(vk, m, \sigma) \text{ accepts}$$
$$\wedge\, \exists \mathsf{VK} \in \mathsf{R} : \mathsf{VK} = (vk, pk, \rho) \text{ for some } pk \text{ and } \rho$$

where, for each $j \in [\ell]$, $pk_j$ is the public key in $\mathsf{VK}_j$. Let $L_4, L_5$ be the languages corresponding to $R_4, R_5$ respectively.

Define further the relation $\widehat{R}$ according to

$$\widehat{R} := R_4 \setminus R_5,$$

and let $\widehat{L}$ be the corresponding language. Finally, for statements of the form $x = (x_1 = (m, c_1, c_2, hk_1, hk_2, h_1, h_2), x_2 = \mathsf{R})$, let $\widetilde{L}$ be the language given by

$$\widetilde{L} = \{(m, c_1, c_2, hk_1, hk_2, h_1, h_2, \mathsf{R}) | (m, c_1, hk_1, h_1, \mathsf{R}) \in \widehat{L} \wedge (m, c_2, hk_2, h_2, \mathsf{R}) \in \widehat{L}\}.$$

Given the properties of the SPB and PKE we can quickly prove the following lemma.

**Lemma 1.** *If* SPB *is somewhere perfectly binding and* PKE *is complete,* $\widetilde{L}$ *is a super-complement of* $L$*.*

We will employ the relaxed ZAP scheme for the languages $L$ and $\widetilde{L}$.

### 5.1 Construction

**Construction 3.** Let $\widetilde{D} = \widetilde{D}(\lambda, N)$ be the maximum depth of the NP verifier circuit for language $\widetilde{L}$ restricted to statements where the the ring has at most $N$ members, and the security parameter corresponding to SPB hash keys and values and PKE ciphertext is $\lambda$. By $n = n(\lambda, \log N)$ denote the maximum size of the statements of language $L$ where the ring has at most $N$ members and the security parameter is $\lambda$. Recall that for security parameter $\lambda$, secret keys in PKE have size $\widetilde{\ell} = \ell_{sk}(\lambda)$. We now describe our ring signature construction:

– $\mathsf{Gen}(1^\lambda, N)$:
  - sample signing and verification keys $(vk, sk) \leftarrow \mathsf{Sig.Gen}(1^\lambda)$,
  - sample $pk$ uniformly from the keyspace of $\mathsf{PKE}$,
  - compute the first message $\rho \leftarrow \mathsf{ZAP.V}(1^\lambda, 1^n, 1^{\widetilde{\ell}}, 1^{\widetilde{D}})$ for the relaxed ZAP scheme,
  - output the verification key $\mathsf{VK} = (vk, pk, \rho)$ and signing key $\mathsf{SK} = (sk, vk, pk, \rho)$.
– $\mathsf{Sign}(\mathsf{SK}, m, R = (\mathsf{VK}_1, \ldots, \mathsf{VK}_\mathsf{l}))$:
  - parse $\mathsf{SK} = (sk, vk, pk, \rho)$,
  - compute $\sigma \leftarrow \mathsf{Sig.Sign}(sk, m)$,
  - let $\mathsf{VK} = \mathsf{VK}_\mathsf{i} \in \mathsf{R}$ be the verification key corresponding to $\mathsf{SK}$,
  - sample hash keys $(hk_1, shk_1) \leftarrow \mathsf{SPB.Gen}(1^\lambda, |\mathsf{R}|, i)$, and compute the hash $h_1 \leftarrow \mathsf{SPB.Hash}(\mathsf{hk}_1, \mathsf{R})$,
  - compute the opening $\tau_1 \leftarrow \mathsf{SPB.Open}(hk_1, shk_1, \mathsf{R}, i)$ to position $i$,
  - compute $c_1 \leftarrow \mathsf{PKE.Enc}(pk, (\sigma, vk); r_{c_1})$
  - sample hash keys $(hk_2, shk_2) \leftarrow \mathsf{SPB.Gen}(1^\lambda, |\mathsf{R}|, i)$ and compute the hash $h_2 \leftarrow \mathsf{SPB.Hash}(\mathsf{hk}_2, \mathsf{R})$,
  - sample $c_2$ randomly from the ciphertext space of $\mathsf{PKE}$,
  - let $\mathsf{VK}_1 = (vk_1, pk_1, \rho_1)$ denote the lexicographically smallest member of $R$ (as a string; note that this is necessarily unique).,
  - fix statement $x_1 = (m, c_1, c_2, hk_1, hk_2, h_1, h_2)$, witness $w = (vk, pk, i, \tau_1, \sigma, r_{c_1})$, and statement $x_2 = \mathsf{R}$,
  - Compute $\pi \leftarrow \mathsf{ZAP.P}(\rho_1, x = (x_1, x_2), w)$,
  - output $\Sigma = (c_1, hk_1, c_2, hk_2, \pi)$.
– $\mathsf{Verify}(\Sigma, m, \mathsf{R})$:
  - identify the lexicographically smallest verification key $\mathsf{VK}_1$ in $\mathsf{R}$,
  - compute $h'_1 = \mathsf{SPB.Hash}(hk_1, \mathsf{R})$,
  - compute $h'_2 = \mathsf{SPB.Hash}(hk_2, \mathsf{R})$,
  - fix $x_1 = (m, c_1, c_2, hk_1, hk_2, h'_1, h'_2)$, and $x_2 = \mathsf{R}$,
  - determine $\rho_1$ in $\mathsf{VK}_1$,
  - compute and output $\mathsf{ZAP.Verify}(\rho_1, x, \pi)$.

Completeness of Construction 3 follows by the completeness of $\mathsf{SPB}$ and $\mathsf{ZAP}$. For compactness, notice that $\widetilde{D}$ is upper-bounded by a polynomial in $\lambda$ and $\log N$, and therefore, since Construction 2 is compact, Construction 3 is also compact.

### 5.2 Unforgeability

Here, we prove that our ring signature scheme possesses the unforgeability property as defined in Definition 7. The proof strategy is as follows: we leverage the selective non-witness adaptive statement soundness of $\mathsf{ZAP}$ to conclude that there must be a valid signature $\sigma$ in the forgery attempt, and essentially try to obtain this signature with significantly high probability so that we can devise a reduction to the existential unforgeability of $\mathsf{Sig}$.

**Theorem 8.** *Construction 3 is unforgeable, assuming* Sig *is* EUF−CMA *secure,* PKE *has injective key generation and pseudorandom public keys,* SPB *is somewhere perfectly binding, and* ZAP *satisfies selective non-witness adaptive statement soundness.*

*Proof.* We start by considering a PPT adversary $\mathcal{A}$ that participates in the unforgeability game. Let $Q = \text{poly}(\lambda)$ be an upper bound on the number of key queries made by $\mathcal{A}$.

We proceed with a hybrid argument to set up our reduction to the unforgeability of Sig. Consider the following hybrids:

**Hybrid $H_0$:** This is just the standard unforgeability game. In particular, for all $i \in [Q]$, the challenger in the game generates $pk_i$ by sampling an element uniformly from the keyspace of PKE.

**Hybrid $H_1$:** In this experiment, the only difference is that, the challenger first picks a uniformly random secret key $sk_{\mathsf{PKE}}$ for PKE, and then generates the corresponding public keys for the adversary using this, namely $pk_i \leftarrow \mathsf{PKE.GenWithKey}(sk_{\mathsf{PKE}})$, for all $i \in [Q]$. The challenger now stores $sk_{\mathsf{PKE}}$.

**Lemma 2.** *Assuming* PKE *has pseudorandom public keys,* $H_0 \overset{c}{\approx} H_1$.

*Proof.* Let $\mathcal{A}$ be a PPT adversary attempting to distinguish $H_0$ and $H_1$. We use $\mathcal{A}$ to build an adversary $\mathcal{A}'$ having the same advantage against the pseudorandomness of public keys of PKE. Here, $\mathcal{A}'$ is either given $\{pk_i \leftarrow \mathsf{GenWithKey}(sk)\}_{i\in[Q]}$ for a $sk$ chosen uniformly at random or $\{pk_i \leftarrow \mathsf{PK}_\lambda\}_{i\in[Q]}$. We define $\mathcal{A}'$ to proceed exactly as in $H_0$ but using the public keys that is given to it as input. Clearly, if $pk_i$s are chosen with a single uniformly chosen secret key, then, the view of $\mathcal{A}$ is identical to $H_1$, whereas, if $pk_i$s are chosen uniformly at random, the view of $\mathcal{A}$ is identical to $H_0$.

Now, we will proceed to show that unforgeability holds in $H_1$. Consider the adversary's forgery attempt $(\Sigma^* = (c_1^*, hk_1^*, c_2^*, hk_2^*, \pi^*), m^*, \mathsf{R}^*)$. Define $x_1^*$ as the statement corresponding to $\Sigma^*$ as $x_1^* = (m^*, c_1^*, c_2^*, hk_1^*, hk_2^*, h_1^*, h_2^*)$, where $h_1^* = \mathsf{SPB.Hash}(hk_1^*, \mathsf{R}^*)$ and $h_2^* = \mathsf{SPB.Hash}(hk_2^*, \mathsf{R}^*)$. Let $\mathsf{VK}_1^* = (vk_1^*, pk_1^*, \rho_1^*)$ be the lexicographically smallest verification key in $\mathsf{R}^*$.

Our next step is to show that if $\pi^*$ is a valid proof for $x^* = (x_1^*, x_2^* = \mathsf{R}^*)$ under $\rho_1^*$, then, with overwhelming probability, $x^* \notin \widetilde{L}$.

**Lemma 3.** *In $H_1$, assuming* ZAP *satisfies selective non-witness adaptive statement soundness, and* PKE *has injective key generation,*

$$\Pr[x^* \in \widetilde{L} \wedge \mathsf{ZAP.Verify}(\rho_1^*, x^*, \pi^*) \text{ accepts }] = \text{negl}(\lambda).$$

*Proof.* It is enough to show that for each $j \in [Q]$,

$$\Pr[x^* \in \widetilde{L} \wedge \mathsf{ZAP.Verify}(\rho_j, x^*, \pi^*) \text{ accepts }] = \text{negl}(\lambda),$$

where $\rho_j$ denotes the ZAP first message corresponding to the $j$th verification key $\mathsf{VK}_j$ generated in the game.

Let $\mathcal{A}$ be an adversary attempting to output a forgery such that $x^* \in \widetilde{L} \wedge \mathsf{ZAP.Verify}(\rho_j, x^*, \pi^*)$ accepts. We build an adversary $\mathcal{A}'$ against the selective non-witness adaptive statement soundness of $\mathsf{ZAP}$ for languages $L$ and $\widetilde{L}$ with fixed non-witness $\widetilde{w} = sk_{\mathsf{PKE}}$. The algorithm $\mathcal{A}'$ proceeds as follows:

- on input $\mathsf{ZAP}$ first message $\widehat{\rho}$, it sets $\rho_j = \widehat{\rho}$ and then proceeds exactly as $H_1$.
- upon recieving the forgery attempt $\Sigma^*$ from $\mathcal{A}$, it constructs the corresponding $x^*$ and $\pi^*$, and outputs $(x^*, \pi^*)$.

To finish the proof of this lemma, we observe that if $x^* \in \widetilde{L}$, then, except with negligible probability, $(x^*, sk_{\mathsf{PKE}}) \in \widetilde{R}$. This is because, if $x^* \in \widetilde{L}$, then, by definition of $\widetilde{L}$, there exists a non-witness $\widetilde{w}^*$ such that,

$$\forall (vk_i^*, pk_i^*, \rho_i^*) \in \mathsf{R}^* : \mathsf{PKE.Valid}(pk_i^*, \widetilde{w}^*) \text{ accepts},$$

and since $\mathsf{PKE}$ has injective key generation, it follows that except with negligible probability, $\widetilde{w}^* = sk_{\mathsf{PKE}}$.

In the next lemma we show that if $x^* \notin \widetilde{L}$, then, by decrypting $c_1^*$ or $c_2^*$, we can find a forgery for $\mathsf{Sig}$.

**Lemma 4.** *In $H_1$, assuming $\mathsf{Sig}$ is $\mathsf{EUF{-}CMA}$ secure, $\mathsf{PKE}$ has injective key generation, and $\mathsf{SPB}$ is somewhere perfectly binding,*

$$\Pr[x^* \notin \widetilde{L}] = \mathrm{negl}(\lambda).$$

*Proof.* Let $\mathcal{A}$ be an adversary attempting to output a forgery such that $x^* \notin \widetilde{L}$. We build an algorithm $\mathcal{A}'$ against the $\mathsf{EUF{-}CMA}$ security of $\mathsf{Sig}$. The algorithm $\mathcal{A}'$ proceeds as follows,

- on input $\widehat{vk}$, first picks an index $j \leftarrow [Q]$ uniformly at random, and then sets $vk_j = \widehat{vk}$. It then proceeds as in $H_1$,
- when $\mathcal{A}$ sends a signing query, if it is using keys from a party other than the $j$th party, it proceeds as in $H_1$, otherwise, it uses the $\mathsf{EUF{-}CMA}$ game's signing oracle to obtain a signature for the $j$th party and then continues exactly as in $H_1$.
- if $\mathcal{A}$ tries to corrupt the $j$th party, $\mathcal{A}'$ aborts.
- upon recieving the forgery attempt $\Sigma^*$ from $\mathcal{A}$, it decrypts $c_1^*$ using $sk_{\mathsf{PKE}}$ to recover $\sigma_1^*$.
  If $\mathsf{Sig.Verify}(vk_j, m^*, \sigma_1^*)$ accepts, it sets $\widehat{\sigma} := \sigma_1^*$. Otherwise, it decrypts $c_2^*$ with $sk_{\mathsf{PKE}}$ to recover $\sigma_2^*$, and sets $\widehat{\sigma} = \sigma_2^*$. It outputs $(m^*, \widehat{\sigma})$.

To finish the proof, we show that with probability at least

$$\frac{1}{Q}(\Pr[x^* \notin \widetilde{L}] - \mathrm{negl}(\lambda)),$$

$(m^*, \widehat{\sigma})$ is a valid forgery for key $vk_j$. Without loss of generality assume that

$$(m^*, c_1^*, hk_1^*, h_1^*, \mathsf{R}^*) \notin \widehat{L}.$$

Observe that due to the way $H_1$ generates the public keys and also by definition of $h_1^*$,

$$(m^*, c_1^*, hk_1^*, h_1^*, \mathsf{R}^*) \in L_4.$$

Therefore, by definition of $\widehat{L}$, there exists a string $\widetilde{w}$ such that,

$$((m^*, c_1^*, hk_1^*, h_1^*, \mathsf{R}^*), \widetilde{w}) \in R_5.$$

By an argument similar to the one presented in Lemma 3, it follows that except with negligible probability $\widetilde{w} = sk_{\mathsf{PKE}}$. Consequently, (i) $\mathsf{PKE.Dec}(sk_{PKE}, c_1^*) = (\sigma^*, vk^*)$, (ii) due to the somewhere perfectly binding property of $\mathsf{SPB}$, there exists $\mathsf{VK}^* = (vk^*, pk^*, \rho^*)$ such that $\mathsf{VK}^* \in \mathsf{R}^*$, and finally (iii) $\mathsf{Sig.Verify}(vk^*, m^*, \sigma^*)$ accepts. We conclude that the adversary uses a verification key $\mathsf{VK}^* \in \mathsf{R}^*$ and that $c_1^*$ encrypts (among other things) a signature $\sigma^*$ that is valid for the forgery message $m^*$ w.r.t. key $vk^*$. Since index $j$ is chosen uniformly at random, $vk_j = vk^*$ with probability $1/Q$.

Lemma 3 and Lemma 4 show that any efficient adversary has negligible chance of winning the $\mathsf{RS{-}FORGE}$ game in hybrid $H_1$. We observe that winning the $\mathsf{RS{-}FORGE}$ game is an event that can be efficiently tested, therefore, by Lemma 2 no efficient adversary can win the $\mathsf{RS{-}FORGE}$ game in hybrid $H_0$, i.e., Construction 3 is unforgeable.

### 5.3 Anonymity

We now prove that our construction satisfies anonymity. Recall that this corresponds to an experiment where the adversary recieves the secret keys and randomness of all the existing parties, and then recieves a challenge signature created using the keys of one of two possible parties (of course, the challenge ring may also include parties that were created by the adversary). Our task is to show that the adversary cannot distinguish between a signature created by party $i_0$ and one created by party $i_1$ (for any distinct $i_0, i_1$). We will do this using a sequence of hybrids. Our strategy will be roughly as follows: we start with a signature produced using the signing key of party $i_0$. First, we switch $c_2$ to valid encryptions of a signature under $vk_{i_1}$ (along with $vk_{i_1}$) and $hk_2$ to a valid $\mathsf{SPB}$ hash key to the index for $\mathsf{VK}_{i_1}$ in the ring respectively. Next, we switch the witness used in $\pi$ to use these values (instead of $c_1$ and $hk_1$). Then, we change $c_1$ to valid encryption of a signature under $vk_{i_1}$ and a valid $\mathsf{SPB}$ hash key to the index for $\mathsf{VK}_{i_1}$ in the ring respectively. Finally, we change $c_2$ to a junk ciphertext, as in the honest signing algorithm. The final hybrid just outputs a signature using the keys for party $i_1$, and thus we only have to show that the adversary cannot detect any of the individual changes outlined above.

**Theorem 9.** *Assume* PKE *has close to uniform ciphertexts and sparse valid public keys as described in Definition 3,* SPB *is index hiding, and* ZAP *is statistically witness indistinguishable. Then the ring signature scheme in Construction 3 satisfies the anonymity property described in Definition 7.*

*Proof.* Let $\mathcal{A}$ be a PPT adversary participating in the anonymity game. Let $Q = \text{poly}(\lambda)$ be an upper bound on the number of key queries made by $\mathcal{A}$. Suppose that the adversary's eventual challenge is $(\mathsf{R}, m, i_0, i_1)$. Let $t_0, t_1$ be the indices of $\mathsf{VK}_{i_0}$, $\mathsf{VK}_{i_1}$ in $\mathsf{R}$ respectively. Denote by $\rho$ the ZAP first message corresponding to the lexicographically smallest $\mathsf{VK}$ in $\mathsf{R}$. As pointed out, it suffices to show that a signature prepared using $\mathsf{SK}_{i_0}$ is indistinguishable from one prepared using $\mathsf{SK}_{i_1}$, even when $\mathcal{A}$ has all the keys $\mathsf{VK}_1, \cdots, \mathsf{VK}_Q$ and the randomness used in creating them. We do so using the following hybrids:

**Hybrid** $H_0$**:** This hybrid simply runs the anonymity game honestly as the challenger, and sends an honest signature generated using $\mathsf{SK}_{i_0}$, namely $\Sigma = (c_1, hk_1, c_2, hk_2, \pi)$, as the challenge to the adversary.

**Hybrid** $H_1$**:** The only change in this hybrid is that it samples $hk_2$ in the signature with index $t_1$, i.e. $(hk_2, shk_2) \leftarrow \mathsf{SPB.Gen}(1^\lambda, |\mathsf{R}|, t_1)$.

**Hybrid** $H_2$**:** The only difference between this hybrid and $H_1$ is that here, instead of sampling $c_2$ uniformly from the PKE ciphertext space, it generates $c_2$ as $c_2 \leftarrow \mathsf{PKE.Enc}(pk_{i_1}, (\sigma', vk_{i_1}); r_{c_2})$, where, $\sigma' \leftarrow \mathsf{Sig.Sign}(sk_{i_1}, m)$.

**Hybrid** $H_3$**:** This hybrid works exactly like the previous one, except that it uses a witness corresponding to $(c_2, hk_2)$ to generate the proof $\pi$. Namely, it computes witness $w' = (vk_{i_2}, pk_{i_2}, t_1, \tau_2', \sigma', r_{c_2})$, where, $\tau_2' = \mathsf{SPB.Open}(hk_2, shk_2, \mathsf{R}, t_1)$, and proof $\pi$ is generated as $\pi \leftarrow \mathsf{ZAP.P}(\rho, x, w')$.

**Hybrid** $H_4$**:** This hybrid is similar to $H_3$, except that it now computes $c_1$ by sampling it uniformly from the ciphertext space of PKE.

**Hybrid** $H_5$**:** This hybrid works exactly like the previous, with the only difference being that it generates $hk_1$ with respect to index $t_1$, i.e., $(hk_1, shk_1) \leftarrow \mathsf{SPB.Gen}(1^\lambda, |\mathsf{R}|, t_1)$.

**Hybrid** $H_6$**:** It is identical to $H_5$, except that here, $c_1 \leftarrow \mathsf{PKE.Enc}(pk_{i_1}, (\sigma', vk_{i_1}); r_{c_1})$ where $\sigma' \leftarrow \mathsf{Sig.Sign}(sk_{i_1}, m)$.

**Hybrid** $H_7$**:** The only change in this hybrid is that, it uses a witness corresponding to $(c_1, hk_1)$ to generate the ZAP proof. Namely, it computes $w'' = (vk, pk, t_1, \tau_1', \sigma, r_{c_1},)$, where $\tau_1' = \mathsf{SPB.Open}(hk_1, shk_1, \mathsf{R}, t_1)$, and $\pi = \mathsf{ZAP.P}(\rho, x, w'')$, and uses this in $\Sigma$.

**Hybrid** $H_8$**:** This hybrids works exactly like the previous, except that it now computes $c_2$ by sampling it uniformly from the ciphertext space of Enc. Notice that this hybrid corresponds to generating the signature using $\mathsf{SK}_{i_1}$.

**Lemma 5.** *Assuming* SPB *is index hiding,* $H_0 \overset{c}{\approx} H_1$.

*Proof.* Let $\mathcal{A}$ be an adversary attempting to distinguish $H_0$ and $H_1$. We use $\mathcal{A}$ to build an adversary $\mathcal{A}'$ having the same advantage against the index hiding property of SPB. $\mathcal{A}'$ runs $\mathcal{A}$ and interacts with it exactly like $H_0$, till the point where $\mathcal{A}$ sends its challenge $(\mathsf{R}, m, i_0, i_1)$. At this point, $\mathcal{A}'$ sends $(t_0, t_1, |\mathsf{R}|)$ to its index hiding challenger. $\mathcal{A}'$ then receives a SPB hash key $hk^*$, which is either $\mathsf{SPB.Gen}(1^\lambda, |\mathsf{R}|, t_0)$ or $\mathsf{SPB.Gen}(1^\lambda, |\mathsf{R}|, t_1)$. It uses $hk^*$ as the key $hk_2$ for generating the challenge signature $\Sigma$ for $\mathcal{A}$. If $hk^*$ is generated for index $t_0$ then $\mathcal{A}$'s view is identical to its view in $H_0$. Otherwise, if $hk^*$ corresponds to $t_1$, $\mathcal{A}$'s view is identical to its view in $H_1$.

**Lemma 6.** *If* PKE *has close to uniform ciphertexts,* $H_1 \overset{s}{\approx} H_2$.

*Proof.* This follows directly from the definition of close to uniform ciphertexts property described in Definition 3.

**Lemma 7.** *If* ZAP *is statistically witness indistinguishable, and* PKE *has sparse valid public keys,* $H_2 \overset{s}{\approx} H_3$.

*Proof.* At least two of the public-keys in $\mathsf{R}$, $pk_{i_0}$ and $pk_{i_1}$ are generated uniformly at random. Consequently, since PKE has sparse valid public keys, except with negligible probability,

$$\nexists sk : (\forall (vk, pk, \rho) \in \mathsf{R} : \mathsf{PKE.Valid}(pk, sk) \text{ accepts }).$$

Thus, $x_1 \notin \widetilde{L}$, and consequently, the lemma follows from the definition of witness indistinguishability described in Definition 3.

**Lemma 8.** *If* PKE *has close to uniform ciphertexts,* $H_3 \overset{s}{\approx} H_4$.

*Proof.* This follows directly from the definition of close to uniform ciphertexts property described in Definition 3.

**Lemma 9.** *Assuming* SPB *is index hiding,* $H_4 \overset{c}{\approx} H_5$.

*Proof.* The proof of the lemma is almost identical to Lemma 5.

**Lemma 10.** *If* PKE *has close to uniform ciphertexts,* $H_5 \overset{s}{\approx} H_6$.

*Proof.* This follows directly from the definition of close to uniform ciphertexts property described in Definition 3.

**Lemma 11.** *If* ZAP *is statistically witness indistinguishable, and* PKE *has sparse valid public keys,* $H_6 \overset{s}{\approx} H_7$.

*Proof.* The proof for this lemma is very similar to Lemma 5.3 and we won't repeat it.

**Lemma 12.** *If* PKE *has close to uniform ciphertexts,* $H_7 \overset{s}{\approx} H_8$.

*Proof.* This follows directly from the definition of close to uniform ciphertexts property described in Definition 3.

This completes the proof of Theorem 9

## 6 Acknowledgments

We thank anonymous reviewers for pointing out issues in Definition 4 and Definition 8 in an earlier version of this work.

Omkant Pandey is supported in part by DARPA SIEVE Award HR00112020026, NSF grants 1907908 and 2028920, and a Cisco Research Award.

Sanjam Garg is supported in part by DARPA under Agreement No. HR00112020026, AFOSR Award FA9550-19-1-0200, NSF CNS Award 1936826, and research grants by the Sloan Foundation and Visa Inc.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government, DARPA, AFOSR, NSF, Cisco, Sloan Foundation, or Visa Inc.

## References

1. Abe, M., Ohkubo, M., Suzuki, K.: 1-out-of-n signatures from a variety of keys. In: ASIACRYPT. pp. 415–432 (2002)
2. Backes, M., Döttling, N., Hanzlik, L., Kluczniak, K., Schneider, J.: Ring signatures: Logarithmic-size, no setup - from standard assumptions. In: EUROCRYPT. pp. 281–311 (2019)
3. Backes, M., Hanzlik, L., Kluczniak, K., Schneider, J.: Signatures with flexible public key: Introducing equivalence classes for public keys. In: ASIACRYPT. pp. 405–434 (2018)
4. Badrinarayanan, S., Fernando, R., Jain, A., Khurana, D., Sahai, A.: Statistical ZAP arguments. In: EUROCRYPT. pp. 642–667 (2020)
5. Baum, C., Lin, H., Oechsner, S.: Towards practical lattice-based one-time linkable ring signatures. In: ICICS. pp. 303–322 (2018)
6. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. In: TCC. pp. 60–79 (2006)
7. Beullens, W., Katsumata, S., Pintore, F.: Calamari and falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In: ASIACRYP. pp. 464–492 (2020)
8. Blum, M.: How to prove a theorem so no one else can claim it. In: Proceedings of the International Congress of Mathematicians. pp. 1444–1451 (1987)
9. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: EUROCRYPT. pp. 416–432 (2003)
10. Boyen, X., Haines, T.: Forward-secure linkable ring signatures. In: ACISP. pp. 245–264 (2018)
11. Brakerski, Z., Döttling, N.: Two-message statistically sender-private OT from LWE. In: TCC. pp. 370–390 (2018)
12. Brakerski, Z., Kalai, Y.T.: A framework for efficient signatures, ring signatures and identity based encryption in the standard model. Cryptology ePrint Archive, Report 2010/086 (2010), https://eprint.iacr.org/2010/086
13. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: STOC. pp. 575–584 (2013)
14. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D., Wichs, D.: Fiat-Shamir: From practice to theory. In: STOC (2019), to appear

15. Chandran, N., Groth, J., Sahai, A.: Ring signatures of sub-linear size without random oracles. In: ICALP. pp. 423–434 (2007)
16. Chow, S.S.M., Wei, V.K., Liu, J.K., Yuen, T.H.: Ring signatures without random oracles. In: ASIACCS. pp. 297–302 (2006)
17. Dodis, Y., Kiayias, A., Nicolosi, A., Shoup, V.: Anonymous identification in ad hoc groups. In: EUROCRYPT. pp. 609–626 (2004)
18. Dwork, C., Naor, M.: Zaps and their applications. SIAM J. Comput. **36**(6), 1513–1543 (2007)
19. Esgin, M.F., Zhao, R.K., Steinfeld, R., Liu, J.K., Liu, D.: Matrict: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In: CCS. pp. 567–584 (2019)
20. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: CRYPTO. pp. 186–194 (1986)
21. Gentry, C.: A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University (2009), http://crypto.stanford.edu/craig
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC. pp. 197–206 (2008)
23. Ghadafi, E.: Sub-linear blind ring signatures without random oracles. In: IMACC. pp. 304–323 (2013)
24. González, A.: A ring signature of size $\Theta$(sqrt[3]{n}) without random oracles (2017), https://eprint.iacr.org/2017/905
25. Goyal, V., Jain, A., Jin, Z., Malavolta, G.: Statistical zaps and new oblivious transfer protocols. In: EUROCRYPT. pp. 668–699 (2020)
26. Groth, J., Kohlweiss, M.: One-out-of-many proofs: Or how to leak a secret and spend a coin. In: EUROCRYPT. pp. 253–280 (2015)
27. Herranz, J., Sáez, G.: Forking lemmas for ring signature schemes. In: INDOCRYPT. pp. 266–279 (2003)
28. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In: EUROCRYPT. pp. 1–31 (2016)
29. Libert, B., Peters, T., Qian, C.: Logarithmic-size ring signatures with tight security from the DDH assumption. In: ESORICS. pp. 288–308 (2018)
30. Libert, B., Nguyen, K., Peters, T., Yung, M.: One-shot fiat-shamir-based nizk arguments of composite residuosity in the standard model. Cryptology ePrint Archive, Report 2020/1334 (2020), https://eprint.iacr.org/2020/1334
31. Lombardi, A., Vaikuntanathan, V., Wichs, D.: 2-message publicly verifiable WI from (subexponential) LWE. Cryptology ePrint Archive, Report 2019/808 p. 808 (2019), https://eprint.iacr.org/2019/808
32. Lyubashevsky, V., Nguyen, N.K., Seiler, G.: SMILE: set membership from ideal lattices with applications to ring signatures and confidential transactions. In: CRYPTO (2021), to appear
33. Malavolta, G., Schröder, D.: Efficient ring signatures in the standard model. In: ASIACRYPT. pp. 128–157. Lecture Notes in Computer Science (2017)
34. Merkle, R.C.: A digital signature based on a conventional encryption function. In: CRYPTO. pp. 369–378 (1987)
35. Micciancio, D.: Duality in lattice cryptography. In: Public Key Cryptography (2010), invited talk
36. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC. pp. 427–437 (1990)
37. Noether, S.: Ring signature confidential transactions for monero (2015), https://eprint.iacr.org/2015/1098

38. Ostrovsky, R., Paskin-Cherniavsky, A., Paskin-Cherniavsky, B.: Maliciously circuit-private FHE. In: CRYPTO. pp. 536–553 (2014)
39. Park, S., Sealfon, A.: It wasn't me! - repudiability and claimability of ring signatures. In: CRYPTO. pp. 159–190 (2019)
40. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: STOC. pp. 333–342 (2009)
41. Peikert, C., Regev, O., Stephens-Davidowitz, N.: Pseudorandomness of Ring-LWE for any ring and modulus. In: STOC. pp. 461–473 (2017)
42. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: CRYPTO. pp. 89–114 (2019)
43. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 1–40 (2009), preliminary version in STOC 2005
44. Rivest, R., Tauman, A.S.Y.: How to leak a secret. In: ASIACRYPT. pp. 552–565 (2001)
45. Schäge, S., Schwenk, J.: A cdh-based ring signature scheme with short signatures and public keys. In: Financial Cryptography and Data Security. pp. 129–142 (2010)
46. Shacham, H., Waters, B.: Efficient ring signatures without random oracles. In: PKC. pp. 166–180 (2007)
47. Torres, W.A.A., Steinfeld, R., Sakzad, A., Liu, J.K., Kuchta, V., Bhattacharjee, N., Au, M.H., Cheng, J.: Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (lattice ringct v1. 0). In: Australasian Conference on Information Security and Privacy. pp. 558–576. Springer (2018)