Post-Quantum Zero-Knowledge Proofs for Accumulators with Applications to Ring Signatures from Symmetric-Key Primitives

David Derler[1], Sebastian Ramacher[1], and Daniel Slamanig[2]

[1] IAIK, Graz University of Technology, Graz, Austria
[2] AIT Austrian Institute of Technology, Vienna, Austria
firstname.lastname@tugraz.at, firstname.lastname@ait.ac.at

Abstract. In this paper we address the construction of privacy-friendly cryptographic primitives for the post-quantum era and in particular accumulators with zero-knowledge membership proofs and ring signatures. This is an important topic as it helps to protect the privacy of users in online authentication or emerging technologies such as cryptocurrencies. Recently, we have seen first such constructions, mostly based on assumptions related to codes and lattices. We, however, ask whether it is possible to construct such primitives without relying on structured hardness assumptions, but solely based on symmetric-key primitives such as hash functions or block ciphers. This is interesting because the resistance of latter primitives to quantum attacks is quite well understood.

In doing so, we choose a modular approach and firstly construct an accumulator (with one-way domain) that allows to efficiently prove knowledge of (a pre-image of) an accumulated value in zero-knowledge. We, thereby, take care that our construction can be instantiated solely from symmetric-key primitives and that our proofs are of sublinear size. Latter is non trivial to achieve in the symmetric setting due to the absence of algebraic structures which are typically used in other settings to make these efficiency gains. Regarding efficient instantiations of our proof system, we rely on recent results for constructing efficient non-interactive zero-knowledge proofs for general circuits. Based on this building block, we then show how to construct logarithmic size ring signatures solely from symmetric-key primitives. As constructing more advanced primitives only from symmetric-key primitives is a very recent field, we discuss some interesting open problems and future research directions. Finally, we want to stress that our work also indirectly impacts other fields: for the first time it raises the requirement for collision resistant hash functions with particularly low AND count.

Keywords: post-quantum cryptography, privacy-preserving cryptography, provable security, accumulator, zero-knowledge for circuits

1 Introduction

The design of cryptographic schemes that remain secure in the advent of powerful quantum computers has become an important topic in recent years. Although it is hard to predict when quantum computers will be powerful enough to break factoring and discrete logarithm based cryptosystems, it is important to start the transition to post-quantum cryptography early enough to eventually not end up in a rush. This is underpinned by the NIST post-quantum cryptography standardization project[3], which aims at identifying the next generation of public key encryption, key exchange and digital signature schemes basing their security on conjectured quantum hard problems. Apart from these fundamental schemes, there are many other valuable schemes which would nicely complement a post-quantum cryptographic toolbox. In this paper we are interested in privacy-friendly cryptographic primitives for the post-quantum era and in particular accumulators with zero-knowledge membership proofs and ring signatures. Such schemes help to protect the privacy of users, and significantly gained importance due to recent computing trends such as Cloud computing or the Internet of Things (IoT). Examples where privacy-enhancing protocols are already widely deployed today are remote attestation via direct anonymous attestation (DAA) [BCC04] as used by the Trusted Platform Module (TPM)[4], privacy-friendly online authentication within Intel's Enhanced Privacy ID (EPID) [BL07], or usage within emerging technologies such as cryptocurrencies to provide privacy of transactions.[5]

Let us now briefly discuss the primitives we construct in this paper. An accumulator scheme [Bd93] allows to represent a finite set as a succinct value called the accumulator. For every element in the accumulated set, one can efficiently compute a so called witness to certify its membership in the accumulator. However, it should be computationally infeasible to find a witness for non-accumulated values. We are interested in accumulators supporting efficient zero-knowledge membership proofs. Ring signature schemes [RST01] allow a member of an ad-hoc group $\mathcal{R}$ (the so called ring), defined by the member's public keys, to anonymously sign a message on behalf of $\mathcal{R}$. Such a signature attests that some member of $\mathcal{R}$ produced the signature, but the actual signer remains anonymous.

For ring signatures there is a known approach to construct them from accumulators and non-interactive zero-knowledge proof systems in the random oracle model. The main technical hurdle in the post-quantum setting is to find accumulators, and, more importantly, compatible proof systems under suitable assumptions. Only recently, Libert et al. in [LLNW16] showed that it is possible to instantiate this approach in the post-quantum setting and provided the first post-quantum accumulator from lattices. This combined with suitable non-interactive variants of Σ-protocols yields post-quantum ring signatures in the random oracle model (ROM). However, this does not give rise to a construction of ring signatures from symmetric-key primitives such as hash functions or block ciphers, as we pursue in this paper. The main technical tools we use in our construction are recent results from zero-knowledge proof systems for general circuits [GMO16, CDG+17], and our techniques are inspired by recent approaches to construct post-quantum signature schemes based on these proof systems [CDG+17]. We note that there are also post-quantum ring signature candidates from problems related to codes [MCG08] and multivariate cryptogra-

[3] `https://csrc.nist.gov/groups/ST/post-quantum-crypto/`
[4] `https://trustedcomputinggroup.org/tpm-library-specification/`
[5] `https://getmonero.org/resources/moneropedia/ringsignatures.html`

phy [MP17]. However, they all have size linear in the number of ring members, whereas we are only interested in sublinear ones. Additionally, former schemes are proven secure in weaker security models.

Contribution. Our contributions are as follows:

- We present the first post-quantum accumulator (with one-way domain) together with efficient zero-knowledge proofs of (a pre-image of) an accumulated value, which solely relies on assumptions related to symmetric-key primitives. That is, we do not require any structured hardness assumptions. Our proofs are of sublinear size in the number of accumulated elements and can be instantiated in both, the ROM as well as the quantum accessible ROM (QROM). Besides being used as an important building block in this paper, such accumulators are of broader interest. In particular, such accumulators with efficient zero-knowledge membership proofs have many other applications beyond this work, e.g., membership revocation [BCD+17] or anonymous cash such as Zerocoin [MGGR13]. We also note that the only previous construction of post-quantum accumulators with efficient zero-knowledge membership proofs in [LLNW16] relies on hardness assumptions on lattices.
- We use our proposed accumulator to construct ring signatures of sublinear size. Therefore, we prove an additional property—simulation-sound extractability—of the proof system (ZKB++ [CDG+17]) we are using. This then allows us to rigorously prove the security of our ring signature construction in the strongest model of security for ring signatures due to Bender et al. [BKM09]. Consequently, we propose a construction of sublinear size ring signatures solely from symmetric-key primitives.
- We present a selection of symmetric-key primitives that can be used to instantiate our ring signature construction and evaluate the practicality of our approach. In particular, we present signature sizes for rings of various sizes when instantiating the one-way function and hash function using LowMC [ARS+15, ARS+16]. Finally, we present some interesting directions for future research within this very recent domain.

2 Preliminaries

Notation. Let $x \xleftarrow{R} X$ denote the operation that picks an element uniformly at random from a finite set X and assigns it to x. We assume that all algorithms run in polynomial time and use $y \leftarrow \mathsf{A}(x)$ to denote that y is assigned the output of the potentially probabilistic algorithm A on input x and fresh random coins. For algorithms representing adversaries we use calligraphic letters, e.g., $\mathcal{A}$. We assume that every algorithm outputs a special symbol $\perp$ on error. We write $\Pr[\Omega : \mathcal{E}]$ to denote the probability of an event $\mathcal{E}$ over the probability space Ω. A function $\epsilon : \mathbb{N} \to \mathbb{R}^+$ is called negligible if for all $c > 0$ there is a k_0 such that $\epsilon(k) < 1/k^c$ for all $k > k_0$. In the remainder of this paper, we use ϵ to denote such a negligible function. Finally, we define $[n] \coloneqq \{1, \ldots, n\}$.

2.1 Zero-Knowledge Proofs and Σ-Protocols

Σ-Protocols. Let $L \subseteq \mathsf{X}$ be an **NP**-language with witness relation R so that $L = \{x \mid \exists w : R(x, w) = 1\}$. A Σ-protocol for language L is defined as follows.

Definition 1 (Σ-Protocol). *A Σ-protocol for language L is an interactive three-move protocol between a PPT prover* $\mathsf{P} = (\mathsf{Commit}, \mathsf{Prove})$ *and a PPT verifier* $\mathsf{V} = (\mathsf{Challenge}, \mathsf{Verify})$*, where* P *makes the first move and transcripts are of the form* $(\mathsf{a}, \mathsf{e}, \mathsf{z}) \in \mathsf{A} \times \mathsf{E} \times \mathsf{Z}$*, where* a *is output by* Commit*,* e *is output by* $\mathsf{Challenge}$ *and* z *is output by* Prove*. Additionally, Σ protocols satisfy the following properties*

Completeness. *For all security parameters κ, and for all $(x, w) \in R$, it holds that*

$$\Pr[\langle \mathsf{P}(1^\kappa, x, w), \mathsf{V}(1^\kappa, x)\rangle = 1] = 1.$$

s-Special Soundness. *There exists a PPT extractor* E *so that for all x, and for all sets of accepting transcripts* $\{(\mathsf{a}, \mathsf{e}_i, \mathsf{z}_i)\}_{i\in[s]}$ *with respect to x where* $\forall i, j \in [s], i \neq j : \mathsf{e}_i \neq \mathsf{e}_j$*, generated by any algorithm with polynomial runtime in κ, it holds that*

$$\Pr\left[w \leftarrow \mathsf{E}(1^\kappa, x, \{(\mathsf{a}, \mathsf{e}_i, \mathsf{z}_i)\}_{i\in[s]}) \,:\, (x, w) \in R\right] \geq 1 - \epsilon(\kappa).$$

Special Honest-Verifier Zero-Knowledge. *There exists a PPT simulator* S *so that for every $x \in L$ and every challenge* $\mathsf{e} \in \mathsf{E}$*, it holds that a transcript* $(\mathsf{a}, \mathsf{e}, \mathsf{z})$*, where* $(\mathsf{a}, \mathsf{z}) \leftarrow \mathsf{S}(1^\kappa, x, \mathsf{e})$ *is computationally indistinguishable from a transcript resulting from an honest execution of the protocol.*

The s-special soundness property gives an immediate bound for soundness: if no witness exists then (ignoring a negligible error) the prover can successfully answer at most to $(s-1)/t$ challenges, where $t = |\mathsf{E}|$ is the size of the challenge space. In case this value is too large, it is possible to reduce the soundness error using ℓ-fold parallel repetition of the Σ-protocol. Furthermore, it is also well known that one can easily express conjunctions and disjunctions of languages proven using Σ-protocols. For the formal details refer to [Dam10, CDS94].

Non-Interactive ZK Proof Systems. Now, we recall a standard definition of non-interactive zero-knowledge proof systems. Therefore, let L be an **NP**-language with witness relation R so that $L = \{x \mid \exists\ w : R(x, w) = 1\}$.

Definition 2 (Non-Interactive Zero-Knowledge Proof System). *A non-interactive proof system* Π *is a tuple of algorithms* (Setup, Proof, Verify)*, defined as:*

$\mathsf{Setup}(1^\kappa)$: *This algorithm takes a security parameter κ as input, and outputs a common reference string* crs.

$\mathsf{Proof}(\mathsf{crs}, x, w)$: *This algorithm takes a common reference string* crs*, a statement x, and a witness w as input, and outputs a proof π.*

$\mathsf{Verify}(\mathsf{crs}, x, \pi)$: *This algorithm takes a common reference string* crs*, a statement x, and a proof π as input, and outputs a bit $b \in \{0, 1\}$.*

We require the properties *completeness*, *adaptive zero-knowledge*, and *simulation-sound extractability* as defined below.

Definition 3 (Completeness). *A non-interactive proof system* $\mathsf{\Pi}$ *is* complete*, if for every adversary* $\mathcal{A}$ *it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ (x, w) \leftarrow \mathcal{A}(\mathsf{crs}), \\ \pi \leftarrow \mathsf{Proof}(\mathsf{crs}, x, w)\end{array} \ :\ \begin{array}{r}\mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \\ \vee\ \ (x, w) \notin R\end{array}\right] \approx 1.$$

Definition 4 (Adaptive Zero-Knowledge). *A non-interactive proof system* $\mathsf{\Pi}$ *is* adaptively zero-knowledge*, if there exists a PPT simulator* $S = (\mathcal{S}_1, \mathcal{S}_2)$ *such that for every PPT adversary* $\mathcal{A}$ *there is a negligible function* $\epsilon(\cdot)$ *such that*

$$\left|\begin{array}{l}\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa)\ :\ \mathcal{A}^{\mathcal{P}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) = 1\right]\ - \\ \Pr\left[(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\kappa)\ :\ \mathcal{A}^{\mathcal{S}(\mathsf{crs},\tau,\cdot,\cdot)}(\mathsf{crs}) = 1\right]\end{array}\right| \leq \epsilon(\kappa),$$

where, τ *denotes a simulation trapdoor. Thereby,* $\mathcal{P}$ *and* $\mathcal{S}$ *return* $\perp$ *if* $(x, w) \notin R$ *or* $\pi \leftarrow \mathsf{Proof}(\mathsf{crs}, x, w)$ *and* $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, x)$*, respectively, otherwise.*

Definition 5 (Simulation-Sound Extractability). *An adaptively zero-knowledge non-interactive proof system* $\mathsf{\Pi}$ *is simulation-sound extractable, if there exists a PPT extractor* $\mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$ *such that for every adversary* $\mathcal{A}$ *it holds that*

$$\left|\begin{array}{l}\Pr\left[(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\kappa)\ :\ \mathcal{A}(\mathsf{crs}, \tau) = 1\right]\ - \\ \Pr\left[(\mathsf{crs}, \tau, \xi) \leftarrow \mathcal{E}_1(1^\kappa)\ :\ \mathcal{A}(\mathsf{crs}, \tau) = 1\right]\end{array}\right| = 0,$$

and for every PPT adversary $\mathcal{A}$ *there is a negligible function* $\varepsilon_2(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}(\mathsf{crs}, \tau, \xi) \leftarrow \mathcal{E}_1(1^\kappa), \\ (x^\star, \pi^\star) \leftarrow \mathcal{A}^{\mathcal{S}(\mathsf{crs},\tau,\cdot)}(\mathsf{crs}), \\ w \leftarrow \mathcal{E}_2(\mathsf{crs}, \xi, x^\star, \pi^\star)\end{array} \ :\ \begin{array}{c}\mathsf{Verify}(\mathsf{crs}, x^\star, \pi^\star) = 1\ \wedge \\ (x^\star, \pi^\star) \notin \mathcal{Q}_{\mathsf{S}}\ \ \wedge\ \ (x^\star, w) \notin R\end{array}\right] \leq \varepsilon_2(\kappa),$$

where $\mathcal{S}(\mathsf{crs}, \tau, x) := \mathcal{S}_2(\mathsf{crs}, \tau, x)$ *and* $\mathcal{Q}_{\mathsf{S}}$ *keeps track of the queries to and answers of* $\mathcal{S}$*.*

The Fiat-Shamir Transform. The Fiat-Shamir transform [FS86] is a frequently used tool to convert Σ-protocols $\langle \mathsf{P}, \mathsf{V} \rangle$ to their non-interactive counterparts. Essentially, the transform removes the interaction between P and V by using a RO $H : \mathsf{A} \times \mathsf{X} \rightarrow \mathsf{E}$ to obtain the challenge e.[6] That is, one uses a PPT algorithm $\mathsf{Challenge}'(1^\kappa, \mathsf{a}, x)$ which obtains $\mathsf{e} \leftarrow H(\mathsf{a}, x)$ and returns e. Then, the prover can locally obtain the challenge e *after* computing the initial message a. Starting a verifier $\mathsf{V}' = (\mathsf{Challenge}', \mathsf{Verify})$ on the same initial message a will then yield the same challenge e. More formally, we obtain the non-interactive PPT algorithms $(\mathsf{P}_H, \mathsf{V}_H)$ indexed by the used RO:

$\mathsf{P}_H(1^\kappa, x, w)$: Start P on $(1^\kappa, x, w)$, obtain the first message a, answer with $\mathsf{e} \leftarrow H(\mathsf{a}, x)$, and finally obtain z. Returns $\pi \leftarrow (\mathsf{a}, \mathsf{z})$.

[6] This is a stronger variant of FS (cf. [FKMV12, BPW12]). The original weaker variant of the FS transform does not include the statement x in the challenge computation.

$\mathsf{V}_H(1^\kappa, x, \pi)$: Parse π as (a, z). Start V' on $(1^\kappa, x)$, send a as first message to V'. When V' outputs e, reply with z and output 1 if V' accepts and 0 otherwise.

One can obtain a non-interactive proof system satisfying the properties above by applying the Fiat-Shamir transform to any Σ-protocol where the min-entropy α of the commitment a sent in the first phase is so that $2^{-\alpha}$ is negligible in the security parameter κ and the challenge space E is exponentially large in the security parameter. Formally, $\mathsf{Setup}(1^\kappa)$ fixes a RO $H : \mathsf{A} \times \mathsf{X} \to \mathsf{E}$, sets $\mathsf{crs} \leftarrow (1^\kappa, H)$ and returns crs. The algorithms Proof and Verify are defined as follows: $\mathsf{Proof}(\mathsf{crs}, x, w) \coloneqq \mathsf{P}_H(1^\kappa, x, w)$, $\mathsf{Verify}(\mathsf{crs}, x, \pi) \coloneqq \mathsf{V}_H(1^\kappa, x, \pi)$.

Signatures via Fiat-Shamir. The Fiat-Shamir (FS) transform can elegantly be used to convert (canonical) identification schemes into adaptively secure signature schemes. The basic idea is similar to above, but slightly differs regarding the challenge generation, i.e., one additionally includes the message upon generating the challenge. Note that in the context of the stronger variant of the FS transform we rely on, one can simply modify the language so that the statements additionally include the message to be signed. This is because our variant of the FS transform includes the statement upon challenge generation, which is why extending the statement by the message also implicitly means including the message in the challenge generation. We will not make this language change explicit in the following, but implicitly assume that the language is changed if a message is included as the last parameter of the statement to be proven.

The Unruh Transform. Similar to FS, Unruh's transform [Unr12, Unr15, Unr16] allows one to construct NIZK proofs and signature schemes from Σ-protocols. In contrast to the FS transform, Unruh's transform can be proven secure in the QROM (quantum random oracle model), strengthening the security guarantee against quantum adversaries. At a high level, Unruh's transform works as follows: given Σ-protocol, the prover repeats the first phase of the Σ-protocol t times and for each of those runs produces responses for M randomly selected challenges. All those responses are permuted using a random permutation G. Querying the random oracle on all first rounds all permuted responses then determines the responses to publish for each round.

2.2 Efficient NIZK Proof Systems for General Circuits

ZKB++ [CDG+17], an optimized version of ZKBoo [GMO16], is a proof system for zero-knowledge proofs over arbitrary circuits. ZKBoo and ZKB++ build on the MPC-in-the-head paradigm by Ishai et al. [IKOS09], which roughly works as follows. The prover simulates all parties of a multiparty computation protocol (MPC) implementing the joint evaluation of some function, say $y =$ SHA-256(x), and computes commitments to the states of all players. The verifier then randomly corrupts a subset of the players and checks whether those players did the computation correctly.

ZKBoo generalizes the idea of [IKOS09] by replacing MPC with circuit decompositions. There the idea is to decompose the circuit into three shares, where revealing the wire values of two shares does not leak any information about the wire values on the input of the circuit. The explicit formulas for circuit decomposition can be found in [GMO16] for ZKBoo and in [CDG+17] for ZKB++. Multiplication gates induce some dependency between the individual shares which is why the wire values on the output of the multiplication gates needs to be stored in the transcripts. Hence, the transcripts grow linearly in the number of multiplication gates. Due to space limitations we do not include further details on ZKB++ and refer the reader to [CDG+17] for the details.

3 PQ Accumulators & ZK Membership Proofs

Our goal is to come up with an accumulator and associated efficient zero-knowledge membership proof system, which remains secure in the face of attacks by a quantum attacker. The first building block we, thus, require for our constructions are accumulators which can be proven secure under an assumption which is believed to resist attacks by a quantum computer. In this work our goal is to solely rely on unstructured assumptions, and thus resort to using Merkle-trees as accumulators. Merkle-trees were first used in the context of accumulators by Buldas, Laud, and Lipmaa in [BLL00], who called their primitive undeniable attesters. In the fashion of [DKNS04], we then extend the accumulator model to accumulators with one-way domain, i.e., accumulators where the accumulation domain coincides with the range of a one-way function so that one can accumulate images of the one-way function. For the associated zero-knowledge membership proof system, we build up on recent progress in proving statements over general circuits as discussed in Section 2.2.

The main technical hurdle we face in this context is designing the statement to be proven with the proof system so that we can actually obtain proofs which are sublinear (in particular logarithmic) in the number of accumulated elements. Obtaining sublinear proofs is complicated mainly due to the absence of any underlying algebraic structure on the accumulator.

3.1 Formal Model

We rely on the formalization of accumulators by [DHS15], which we slightly adapt to fit our requirement for a deterministic Eval algorithm. Based on this formalization we then restate the Merkle-tree accumulator (having a deterministic Eval algorithm) within this framework.

Definition 6 (Accumulator). *A static accumulator is a tuple of efficient algorithms* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{WitCreate}, \mathsf{Verify})$ *which are defined as follows:*

$\mathsf{Gen}(1^\kappa, t)$: *This algorithm takes a security parameter κ and a parameter t. If $t \neq \infty$, then t is an upper bound on the number of elements to be accumulated. It returns a key pair $(\mathsf{sk}_\Lambda, \mathsf{pk}_\Lambda)$, where $\mathsf{sk}_\Lambda = \emptyset$ if no trapdoor exists. We assume that the accumulator public key pk_Λ implicitly defines the accumulation domain D_Λ.*

$\mathsf{Eval}((\mathsf{sk}_\Lambda^\sim, \mathsf{pk}_\Lambda), \mathcal{X})$: *This deterministic algorithm takes a key pair $(\mathsf{sk}_\Lambda^\sim, \mathsf{pk}_\Lambda)$ and a set $\mathcal{X}$ to be accumulated and returns an accumulator $\Lambda_\mathcal{X}$ together with some auxiliary information* aux.

$\mathsf{WitCreate}((\mathsf{sk}_\Lambda^\sim, \mathsf{pk}_\Lambda), \Lambda_\mathcal{X}, \mathsf{aux}, x_i)$: *This algorithm takes a key pair $(\mathsf{sk}_\Lambda^\sim, \mathsf{pk}_\Lambda)$, an accumulator $\Lambda_\mathcal{X}$, auxiliary information* aux *and a value x_i. It returns $\perp$, if $x_i \notin \mathcal{X}$, and a witness wit_{x_i} for x_i otherwise.*

$\mathsf{Verify}(\mathsf{pk}_\Lambda, \Lambda_\mathcal{X}, \mathsf{wit}_{x_i}, x_i)$: *This algorithm takes a public key* pk_Λ*, an accumulator* $\Lambda_\mathcal{X}$*, a witness* wit_{x_i} *and a value* x_i*. It returns* 1 *if* wit_{x_i} *is a witness for* $x_i \in \mathcal{X}$ *and* 0 *otherwise.*

We require accumulators to be correct and collision free. While we omit the straight forward correctness notion, we recall the collision freeness notion below, which requires that finding a witness for a non-accumulated value is hard.

Definition 7 (Collision Freeness). *A cryptographic accumulator is* collision-free*, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that:*

$$\Pr\left[\begin{array}{l}(\mathsf{sk}_\Lambda, \mathsf{pk}_\Lambda) \leftarrow \mathsf{Gen}(1^\kappa, t),\\ (\mathsf{wit}^\star_{x_i}, x^\star_i, \mathcal{X}^\star) \leftarrow \mathcal{A}(\mathsf{pk}_\Lambda)\end{array} : \begin{array}{r}\mathsf{Verify}(\mathsf{pk}_\Lambda, \Lambda^\star, \mathsf{wit}^\star_{x_i}, x^\star_i) = 1 \wedge\\ x^\star_i \notin \mathcal{X}^\star\end{array}\right] \leq \varepsilon(\kappa),$$

where $\Lambda^\star \leftarrow \mathsf{Eval}_{r^\star}((\mathsf{sk}_\Lambda, \mathsf{pk}_\Lambda), \mathcal{X}^\star)$.

3.2 The Accumulator

In Scheme 1, we cast the Merkle-tree accumulator in the framework of [DHS15].

$\underline{\mathsf{Gen}(1^\kappa, t)}$: Fix a family of hash functions $\{H_k\}_{k\in\mathsf{K}^\kappa}$ with $H_k : \{0,1\}^* \to \{0,1\}^\kappa\ \forall\ k \in \mathsf{K}^\kappa$. Choose $k \xleftarrow{R} \mathsf{K}^\kappa$ and return $(\mathsf{sk}_\Lambda, \mathsf{pk}_\Lambda) \leftarrow (\emptyset, H_k)$.

$\underline{\mathsf{Eval}((\mathsf{sk}_\Lambda, \mathsf{pk}_\Lambda), \mathcal{X})}$: Parse pk_Λ as H_k and $\mathcal{X}$ as $(x_0, \ldots, x_{n-1})$.[a] If $\nexists\ k \in \mathbb{N}$ so that $n = 2^k$ return $\perp$. Otherwise, let $\ell_{u,v}$ refer to the u-th leaf (the leftmost leaf is indexed by 0) in the v-th layer (the root is indexed by 0) of a perfect binary tree. Return $\Lambda_\mathcal{X} \leftarrow \ell_{0,0}$ and $\mathsf{aux} \leftarrow ((\ell_{u,v})_{u\in[n/2^{k-v}]})_{v\in[k]}$, where

$$\ell_{u,v} \leftarrow \begin{cases} H_k(\ell_{2u,v+1}||\ell_{2u+1,v+1}) & \text{if } v < k, \text{ and}\\ H_k(x_i) & \text{if } v = k.\end{cases}$$

$\underline{\mathsf{WitCreate}((\mathsf{sk}_{\tilde{\Lambda}}, \mathsf{pk}_\Lambda), \Lambda_\mathcal{X}, \mathsf{aux}, x_i)}$: Parse aux as $((\ell_{u,v})_{u\in[n/2^{k-v}]})_{v\in[k]}$ and return wit_{x_i} where

$$\mathsf{wit}_{x_i} \leftarrow (\ell_{\lfloor i/2^v\rfloor+\eta, k-v})_{0\leq v\leq k}, \text{ where } \eta = \begin{cases} 1 & \text{if } \lfloor i/2^v \rfloor \pmod 2 = 0\\ -1 & \text{otherwise.}\end{cases}$$

$\underline{\mathsf{Verify}(\mathsf{pk}_\Lambda, \Lambda_\mathcal{X}, \mathsf{wit}_{x_i}, x_i)}$: Parse pk_Λ as H_k, $\Lambda_\mathcal{X}$ as $\ell_{0,0}$, set $\ell_{i,k} \leftarrow H_k(x_i)$. Recursively check for all $0 < v < k$ whether the following holds and return 1 if so. Otherwise return 0.

$$\ell_{\lfloor i/2^{v+1}\rfloor, k-(v+1)} = \begin{cases} H_k(\ell_{\lfloor i/2^v\rfloor, k-v}||\ell_{\lfloor i/2^v\rfloor+1, k-v}) & \text{if } \lfloor i/2^v \rfloor \pmod 2 = 0\\ H_k(\ell_{\lfloor i/2^v\rfloor-1, k-v}||\ell_{\lfloor i/2^v\rfloor, k-v}) & \text{otherwise.}\end{cases}$$

[a] We assume without loss of generality that $\mathcal{X}$ is an ordered sequence instead of a set.

Scheme 1: Merkle-tree accumulator.

Then, we restate some well-known lemmas and sketch the respective proofs.

Lemma 1. *Scheme 1 is correct.*

The lemma above is easily verified by inspection. The proof is omitted.

Lemma 2. *If $\{H_k\}_{k \in \mathsf{K}^\kappa}$ is a family of collision resistant hash functions, the accumulator in Scheme 1 is collision free.*

Proof (Sketch). Upon setup, the reduction engages with a collision resistance challenger for the family of hash functions, obtains H_k, and completes the setup as in the original protocol. Now, one may observe that every collision in the accumulator output by the adversary implies that the reduction knows at least two colliding inputs for H_k, which upper bounds the probability of a collision in the accumulator by the collision probability of the hash function.

3.3 Accumulators with One-Way Domain

We now extend the definition of accumulators to ones with one-way domain following the definition of [DKNS04], but we adapt it to our notation.

Definition 8 (Accumulator with One-Way Domain). *A collision-free accumulator with accumulation domain D_Λ and associated function family $\{f_\Lambda : \mathsf{I}_\Lambda \to \mathsf{D}_\Lambda\}$ where $\mathsf{Gen}(1^\kappa, t)$ also selects f_Λ is called an accumulator with one-way domain if*

Efficient verification. *There exists an efficient algorithm D that on input $(x, z) \in \mathsf{D}_\Lambda \times \mathsf{I}_\Lambda$ returns 1 if and only if $f_\Lambda(z) = x$.*

Efficient sampling. *There exists a (probabilistic) algorithm W that on input 1^κ returns a pair $(x, z) \in \mathsf{D}_\Lambda \times \mathsf{I}_\Lambda$ with $D(x, z) = 1$.*

One-wayness. *For all PPT adversaries $\mathcal{A}$ there is a negligible function $\varepsilon(\cdot)$ such that:*

$$\Pr\left[\,(x, z) \leftarrow W(1^\kappa), z^\star \leftarrow \mathcal{A}(1^\kappa, x) \,:\, D(x, z) = 1\,\right] \leq \varepsilon(\kappa).$$

Note that when we set f_Λ to be the identity function, then we have a conventional accumulator.

3.4 Membership Proofs of Logarithmic Size

The main technical tool used by [DKNS04] to obtain zero-knowledge membership proofs of constant size is to exploit a property of the accumulator which is called quasi-commutativity. Clearly, such a property requires some underlying algebraic structure which we explicitly want to sacrifice in favor of being able to solely rely on assumptions related to symmetric-key primitives with relatively well understood post-quantum security. To this end we have to use a different technique. First observe that when naïvely proving that a non-revealed value is a member of our accumulator would amount to a disjunctive proof of knowledge over all members, which is at least of linear size. Therefore, this is not an option and we have to develop an alternative technique.

The Relation. Essentially our idea is to "emulate" some kind of commutativity within the order of the inputs to the hash function in each level by a disjunctive proof statement, i.e., we exploit the disjunction to hide where the path through the tree continues. The single statements in every level of the tree are then included in one big conjunction. The length of this statement is $\mathcal{O}(k) = \mathcal{O}(\log n)$. More formally we define a relation R on $\{0,1\}^\kappa \times \{f_\Lambda\} \times \{H_k\} \times \mathsf{I}_\Lambda \times (\{0,1\}^\kappa)^{2k}$ which—for a given non-revealed pre-image z—attests membership of the corresponding image $f_\Lambda(z)$ in the accumulator $\Lambda_\mathcal{X}$:

$$((\Lambda_\mathcal{X}, f_\Lambda, H_k), (z, (a_i)_{i\in[k]}, (b_i)_{i\in[k]})) \in R \iff (a_k = f_\Lambda(z) \ \vee \ b_k = f_\Lambda(z))$$
$$\wedge \ \bigwedge_{i=0}^{k-1} (a_i = H_k(a_{i+1}||b_{i+1}) \vee a_i = H_k(b_{i+1}||a_{i+1}),$$

where $\Lambda_\mathcal{X} = a_0$. In Figure 1 we illustrate that the relation indeed works for arbitrary members of the accumulator without influencing the form of the statement or the witness. This illustrates that proving the statement in this way does not reveal any information on which path in the tree was taken. To see this, observe that at each level of the tree the relation covers both cases where a_i is either a left or right child. Given that, it is easy to verify that having a witness for relation R implies having a witness for the accumulator together with some (non-revealed) member.

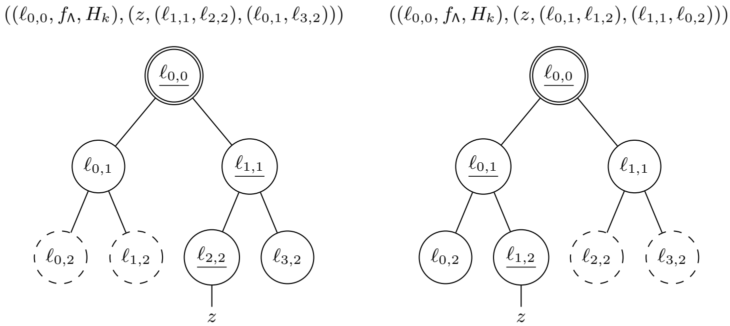

Fig. 1: Visualization of different paths in the Merkle-tree and the corresponding witness. The nodes on the path corresponding to a_0, a_1 and a_2 are underlined.

Remark 1. In order to use relation R with the conventional accumulator in Scheme 1, we just have to set f_Λ to be the identity function (which yields $x = z$) and then set $a_k = H_k(z)$ and $b_k = H_k(z)$.

3.5 Converting Accumulator Witnesses

Now, the remaining piece to finally be able to plug in a witness $\mathsf{wit}_{f_\Lambda(z)}$ for some accumulated value $f_{\Lambda(z)}$ with pre-image z into the relation R above is some efficient helper algorithm which rearranges the values z and $\mathsf{wit}_{f_\Lambda(z)}$ so that they are compatible with the format required by R. Such an algorithm is easily implemented, which is why we only define the interface below.

$\mathsf{Trans}(z, \mathsf{wit}_{f_\Lambda(z)})$: Takes as input a value z as well as a witness $\mathsf{wit}_{f_{\Lambda(z)}}$ and returns a witness of the form $(z, (a_i)_{i\in[k]}, (b_i)_{i\in[k]})$ for R.

Since Trans can be viewed as a permutation on the indexes it is easy to see that the function implemented by Trans is bijective and its inverse is easy to compute. We denote the computation of the inverse of the function implemented by Trans as $(z, \mathsf{wit}_{f_\Lambda(z)}) \leftarrow \mathsf{Trans}^{-1}(z, (a_i)_{i\in[n]}, (b_i)_{i\in[n]})$.

4 Logarithmic Size Ring Signatures

The two main lines of more recent work in the design of ring signatures target reducing the signature size or removing the requirement for random oracles (e.g., [DKNS04, CGS07, GK15, BCC+15, DS16, Gon17, MS17]). We, however, note that all these approaches require assumptions that do not withstand a quantum computer. To the best of our knowledge, the first non-trivial post-quantum scheme (i.e., one that does not have linear size signatures) in the random oracle model is the lattice-based scheme recently proposed by Libert et al. [LLNW16]. We provide an alternative construction in the random oracle model with logarithmic sized signatures, but avoid lattice assumptions and only rely on symmetric-key primitives.

4.1 Formal Model

Below, we formally define ring signature schemes (adopting [BKM09]).

Definition 9 (Ring Signature). *A ring signature scheme* RS *is a tuple* $\mathsf{RS} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *of PPT algorithms, which are defined as follows.*

$\mathsf{Setup}(1^\kappa)$: *This algorithm takes as input a security parameter* κ *and outputs public parameters* PP.

$\mathsf{Gen}(\mathsf{PP})$: *This algorithm takes as input parameters* PP *and outputs a keypair* $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})$: *This algorithm takes as input a secret key* sk_i*, a message* $m \in \mathcal{M}$ *and a ring* $\mathcal{R} = (\mathsf{pk}_j)_{j\in[n]}$ *of* n *public keys such that* $\mathsf{pk}_i \in \mathcal{R}$*. It outputs a signature* σ.

$\mathsf{Verify}(m, \sigma, \mathcal{R})$: *This algorithm takes as input a message* $m \in \mathcal{M}$*, a signature* σ *and a ring* $\mathcal{R}$*. It outputs a bit* $b \in \{0, 1\}$.

A secure ring signature scheme needs to be correct, unforgeable, and anonymous. While we omit the obvious correctness definition, we subsequently provide formal definitions for the remaining properties following [BKM09]. We note that Bender et al. in [BKM09] have formalized multiple variants of these properties, where we always use the *strongest* one.

Unforgeability requires that without any secret key sk_i that corresponds to a public key $\mathsf{pk}_i \in \mathcal{R}$, it is infeasible to produce valid signatures with respect to arbitrary such rings $\mathcal{R}$. Our unforgeability notion is the strongest notion defined in [BKM09] and is there called *unforgeability w.r.t. insider corruption.*

Definition 10 (Unforgeability). *A ring signature scheme provides unforgeability, if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\varepsilon(\cdot)$ *such that it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{PP} \leftarrow \mathsf{Setup}(1^\kappa),\\ \{(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{Gen}(\mathsf{PP})\}_{i\in[\mathsf{poly}(\kappa)]},\\ \mathcal{O} \leftarrow \{\mathsf{Sig}(\cdot,\cdot,\cdot), \mathsf{Key}(\cdot)\},\\ (m^\star, \sigma^\star, \mathcal{R}^\star) \leftarrow \mathcal{A}^{\mathcal{O}}(\{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]})\end{array} : \begin{array}{r}\mathsf{Verify}(m^\star, \sigma^\star, \mathcal{R}^\star) = 1 \ \wedge\\ (\cdot, m^\star, \mathcal{R}^\star) \notin \mathcal{Q}^{\mathsf{Sign}} \ \wedge\\ \mathcal{R}^\star \subseteq \{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]\setminus\mathcal{Q}^{\mathsf{Key}}}\end{array}\right] \le \varepsilon(\kappa),$$

where $\mathsf{Sig}(i, m, \mathcal{R}) := \mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})$*,* Sig *returns* $\perp$ *if* $\mathsf{pk}_i \notin \mathcal{R} \ \vee \ i \notin [\mathsf{poly}(\kappa)]$*, and* $\mathcal{Q}^{\mathsf{Sig}}$ *records the queries to* Sig*. Furthermore,* $\mathsf{Key}(i)$ *returns* sk_i *and* $\mathcal{Q}^{\mathsf{Key}}$ *records the queries to* Key.

Anonymity requires that it is infeasible to tell which ring member produced a certain signature as long as there are at least two honest members in the ring. Our anonymity notion is the strongest notion defined in [BKM09] and is there called *anonymity against full key exposure.*

Definition 11 (Anonymity). *A ring signature scheme provides anonymity, if for all PPT adversaries $\mathcal{A}$ and for all polynomials* $\mathsf{poly}(\cdot)$*, there exists a negligible function $\varepsilon(\cdot)$ such that it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{PP} \leftarrow \mathsf{Setup}(1^\kappa),\\ \{(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{Gen}(\mathsf{PP})\}_{i\in[\mathsf{poly}(\kappa)]},\\ b \xleftarrow{R} \{0,1\},\ \mathcal{O} \leftarrow \{\mathsf{Sig}(\cdot,\cdot,\cdot)\},\\ (m, j_0, j_1, \mathcal{R}, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}}(\{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]}),\\ \sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{j_b}, m, \mathcal{R}),\\ b^\star \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{st}, \sigma, \{\mathsf{sk}_i\}_{i\in[\mathsf{poly}(\kappa)]})\end{array} \ : \ \begin{array}{r} b = b^\star \ \wedge \\ \{\mathsf{pk}_{j_i}\}_{i\in\{0,1\}} \subseteq \mathcal{R}\end{array}\right] \leq 1/2 + \varepsilon(\kappa),$$

where $\mathsf{Sig}(i, m, \mathcal{R}) \coloneqq \mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})$*.*

4.2 Generic Approaches to Design Ring Signatures

A folklore approach to design ring signatures in the random oracle model is to use the **NP** relation R_{RS} together with a one-way function μ, which defines the relation between secret and public keys:

$$(\mathcal{R}, \mathsf{sk}) \in R_{\mathsf{RS}} \iff \exists\ \mathsf{pk}_i \in \mathcal{R}_{\mathsf{RS}}\ :\ \mathsf{pk}_i = \mu(\mathsf{sk}),$$

and allows to demonstrate knowledge of a witness (a secret key) of one of the public keys in the ring $\mathcal{R}$. Usually, one then designs a Σ-protocol for relation R_{RS} and converts it into a signature scheme using the Fiat-Shamir heuristic.

Linear-size signatures. A frequently used instantiation of the above approach is instantiating the relation above by means of a disjunctive proof of knowledge [CDS94]. Using this approach, one obtains ring signatures of linear size. It might be tempting to think that there is a lot of optimization potential for signature sizes in ring signatures. However, without additional assumptions about how the keys are provided to the verifier, signatures of linear size are already the best one can hope for: the verifier needs to get every public key in the ring to verify the signature.

Reducing signature size. However, to further reduce the signature size there is a nice trick which is based on the observation that in many practical scenarios the prospective ring members are already clear prior to the signature generation. Consequently, one can compactly encode all public keys in this ring within some suitable structure and compute the signatures with respect to this compact structure. This trick was first used by Dodis et al. [DKNS04]. Loosely their approach can be described as follows. They use a cryptographic accumulator with a one-way domain to accumulate the ring $\mathcal{R}$, a set of public keys being the output of applying the one-way function μ to the respective secret key. This way they obtain a succinct representation of $\mathcal{R}$. Then, they use a proof system that allows to prove knowledge of a witness of one accumulated value (i.e., the public key) and knowledge of the pre-image thereof (i.e., the corresponding secret key). This proof can be turned into a signature using the Fiat-Shamir heuristic.

Depending on the size of the zero-knowledge membership proof this can yield sublinear (logarithmic or even constant size) signatures. Dodis et al. presented an instantiation of an accumulator together with the respective zero-knowledge proofs that yield constant size ring signatures based on the strong RSA assumption. Logarithmic size ring signatures under lattice assumptions are presented in [LLNW16].

4.3 Our Construction of Logarithmic Size Ring Signatures

Our construction basically follows the approach discussed above to reduce signature size. However, in contrast to Dodis et al., besides targeting the post-quantum setting, we (1) *do not* require a trusted setup[7], and (2) cannot rely on accumulators with one-way domain which provide *quasi-commutativity.* Latter is too restricting and not compatible with the setting in which we work. In particular, it excludes Merkle-tree accumulators, which is why we chose to rely on a more generic formalization of accumulators (cf. Section 3). Like Dodis et al., we assume that in practical situations rings often stay the same for a long period of time (e.g., some popular rings are used very often by various members of the ring), or have an implicit short description. Consequently, we measure the signature size as that of the actual signature, i.e., the information one requires *in addition* to the group description. We want to stress once again that when counting the description of the ring as part of the signature, every secure ring signature schemes needs to have signature sizes which are at least linear in the size of the ring.

For the ease of presentation let us fix one such popular ring $\mathcal{R}$ identified by the corresponding accumulator $\Lambda_{\mathcal{R}}$ and we assume that $|\mathcal{R}| = 2^t$ for some $t \in \mathbb{N}$.[8] We present our construction as Scheme 2.

Remark 2. Note that in Scheme 2 crs is not a common reference string (CRS) that needs to be honestly computed by a trusted third party. We simply stick with the notion including a CRS for formal reasons, i.e,. to allow the abstract notion of NIZKs, but as we exclusively use NIZK from Σ-protocols, we do not require a trusted setup and crs is just a description of the hash function which can be globally fixed, e.g., to SHA-256 or SHA-3. Recall, within Fiat-Shamir $\Pi.\mathsf{Setup}(1^\kappa)$ fixes a RO $H : \mathsf{A} \times \mathsf{X} \to \mathsf{E}$, sets $\mathsf{crs} \leftarrow (1^\kappa, H)$ and returns crs.

Remark 3. A trusted setup in context of ring signatures is actually problematic, as it assumes that some mutually trusted party honestly executes the setup. For instance, in case of the strong RSA accumulator [BP97, CL02] as used within [DKNS04], the party running the Gen algorithm of the accumulator can arbitrarily cheat. This can easily be done by keeping the accumulator secret (a trapdoor) instead of discarding it. Using this information, a dishonest setup allows to insert and delete arbitrary elements into and from the accumulator

[7] A trusted setup somehow undermines the idea behind ring signatures.

[8] If this is not the case, one can always add dummy keys to the ring to satisfy this condition.

$\underline{\mathsf{Setup}(1^\kappa)}$: Let Λ be the accumulator with one-way domain based on Scheme 1, run $(\mathsf{sk}_\Lambda, \mathsf{pk}_\Lambda) \leftarrow \Lambda.\mathsf{Gen}(1^\kappa, t)$ (note that $\mathsf{sk}_\Lambda = \emptyset$). Run $\mathsf{crs} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$ and return $\mathsf{PP} \leftarrow (\mathsf{pk}_\Lambda, \mathsf{crs}) = ((H_k, f_\Lambda), (1^\kappa, H))$.

$\underline{\mathsf{KeyGen}(\mathsf{PP})}$: Parse PP as $((H_k, f_\Lambda), \mathsf{crs})$, run $(x, z) \leftarrow f_\Lambda.W(1^\kappa)$, and set $\mathsf{pk} \leftarrow (\mathsf{PP}, x)$, $\mathsf{sk} \leftarrow (\mathsf{pk}, z)$. Return $(\mathsf{sk}, \mathsf{pk})$.

$\underline{\mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})}$: Parse sk_i as $((((H_k, f_\Lambda), \mathsf{crs}), x_i), z_i)$ and $\mathcal{R}$ as $(\mathsf{pk}_1, \ldots, \mathsf{pk}_t) = ((\cdot, x_1), \ldots, (\cdot, x_t))$. Let $\mathcal{X} = (x_1, \ldots, x_t)$, run $(\Lambda_\mathcal{X}, \mathsf{aux}) \leftarrow \Lambda.\mathsf{Eval}((\cdot, \mathsf{pk}_\Lambda), \mathcal{X})$ and $\mathsf{wit}_{f_\Lambda(z_i)} \leftarrow \Lambda.\mathsf{WitCreate}((\cdot, \mathsf{pk}_\Lambda), \Lambda_\mathcal{X}, \mathsf{aux}, f_\Lambda(z_i))$. Obtain $(z_i, (a_j)_{j\in[t]}, (b_j)_{j\in[t]}) \leftarrow \mathsf{Trans}(z_i, \mathsf{wit}_{f_\Lambda(z_i)})$, and return the signature $\sigma \leftarrow (\pi, \Lambda_\mathcal{X})$, where

$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\Lambda_\mathcal{X}, f_\Lambda, H_k), (z_i, (a_j)_{j\in[t]}, (b_j)_{j\in[t]})).$$

$\underline{\mathsf{Verify}(m, \sigma, \mathcal{R})}$: Parse σ as $(\pi, \Lambda_\mathcal{X})$ and $\mathcal{R}$ as $(\mathsf{pk}_1, \ldots, \mathsf{pk}_t) = ((((H_k, f_\Lambda), \mathsf{crs}), x_1), \ldots, (\cdot, x_t))$. Let $\mathcal{X} = (x_1, \ldots, x_t)$, and compute

$$(\Lambda'_\mathcal{X}, \mathsf{aux}') \leftarrow \Lambda.\mathsf{Eval}((\cdot, \mathsf{pk}_\Lambda), \mathcal{X}).$$

If $\Lambda'_X \neq \Lambda_X$ return 0. Otherwise return $\Pi.\mathsf{Verify}(\mathsf{crs}, (\Lambda_\mathcal{X}, f_\Lambda, H_k), \pi)$.

Scheme 2: Construction of logarithmic size RS.

without changing the accumulator value. In context of ring signatures one thus can arbitrarily modify existing rings used within signatures, which could lead to modification of rings to just include public keys into the ring so that for every member of the ring the sole fact to know that one of these persons produced a signature already leads to severe consequences. *We stress that in our case there is no trusted setup. In particular, there is no accumulator secret and the public parameters are just descriptions of hash functions and a OWF.*

Now, we argue that our ring signature presented in Scheme 2 represents a secure ring signature scheme, where we omit correctness which is straightforward to verify.

Theorem 1. *If Λ is a collision free accumulator with one-way domain with respect to f_Λ and Π is a simulation-sound extractable non-interactive proof system, then the ring signature scheme in Scheme 2 is unforgeable.*

Proof. We prove unforgeability using a sequence of games.

Game 0: The original unforgeability game.

Game 1: As Game 0, but we modify Gen to setup (crs, τ) using $\mathcal{S}_1$ and henceforth simulate all proofs in Sign without a witness using τ.

Transition - Game 0 → Game 1: A distinguisher between Game 0 and Game 1 is a zero-knowledge distinguisher for Π, i.e., $|\Pr[S_0] - \Pr[S_1]| \leq \varepsilon_{\mathsf{zk}}(\kappa)$.

Game 2: As Game 1, but we further modify Gen to setup $(\mathsf{crs}, \tau, \xi)$ using $\mathcal{E}_1$ and store ξ.

Transition - Game 1 → Game 2: By simulation-sound extractability, this change is only conceptual, i.e., $\Pr[S_1] = \Pr[S_2]$.

Game 3: As Game 2, but for the forgery $(m^\star, \sigma^\star, \mathcal{R}^\star)$ output by the adversary we parse $\sigma^\star$ as $(\pi, \Lambda_\mathcal{X})$ and obtain $(z_i, (a_i)_{i\in[k]}, (b_i)_{i\in[k]}) \leftarrow \mathcal{E}_2(\mathsf{crs}, \xi, (\Lambda_\mathcal{X}, f_\Lambda, H_k), \pi)$. If the extractor fails, we abort.

Transition - Game 2 → Game 3: Game 2 and Game 3 proceed identically, unless we abort. The probability for the abort event to happen is upper bounded by $\varepsilon_{\mathsf{ext}}(\kappa)$ which is why we can conclude that $|\Pr[S_3] - \Pr[S_2]| \leq \varepsilon_{\mathsf{ext}}(\kappa)$.

Game 4: As Game 3, but we abort if we have extracted $(z_i, (a_i)_{i\in[n]}, (b_i)_{i\in[n]})$ so that $(\cdot, \mathsf{wit}_{f_\Lambda(z_i)}) \leftarrow \mathsf{Trans}^{-1}(z_i, (a_i)_{i\in[n]}, (b_i)_{i\in[n]})$ is a valid witness for some $f_\Lambda(z_i)$ which was never accumulated.

Transition - Game 3 → Game 4: If we abort in Game 4, we have a collision for the accumulator. That is $|\Pr[S_3] - \Pr[S_4]| \leq \varepsilon_{\mathsf{cf}}(\kappa)$.

Game 5: As Game 4, but we guess the index $i^\star$ the adversary will attack beforehand, and abort if our guess is wrong.

Transition - Game 4 → Game 5: The success probability in Game 4 is the same as in Game 5, unless our guess is wrong, i.e., $\Pr[S_5] = {}^1/_{\mathsf{poly}(\kappa)} \cdot \Pr[S_4]$.

Game 6: As Game 5, but instead of honestly generating the keypair for user $i^\star$, we engage with a challenger of a OWF to obtain $x_{i^\star}$ and include it in $\mathsf{pk}_{i^\star}$ accordingly. We set $\mathsf{sk}_{i^\star} \leftarrow \emptyset$.

Transition - Game 5 → Game 6: This change is conceptual, i.e., $\Pr[S_5] = \Pr[S_6]$.

In the last game, we have an adversary against the OWF, i.e., $\Pr[S_6] \leq \varepsilon_{\mathsf{owf}}(\kappa)$. All in all, we have that $\Pr[S_0] \leq \mathsf{poly}(\kappa) \cdot \varepsilon_{\mathsf{owf}}(\kappa) + \varepsilon_{\mathsf{zk}}(\kappa) + \varepsilon_{\mathsf{ext}}(\kappa) + \varepsilon_{\mathsf{cf}}(\kappa)$

Theorem 2. *If* Π *is a zero-knowledge non-interactive proof system, then the ring signature scheme in Scheme 2 is anonymous.*

Proof. We prove anonymity using a sequence of games.

Game 0: The original anonymity game.

Game 1: As Game 0, but we modify Gen to setup (crs, τ) using $\mathcal{S}_1$ and henceforth simulate all proofs in Sign without a witness using τ.

Transition - Game 0 → Game 1: A distinguisher between Game 0 and Game 1 is a zero-knowledge distinguisher for Π, i.e., $|\Pr[S_0] - \Pr[S_1]| \leq \varepsilon_{\mathsf{zk}}(\kappa)$.

In Game 1 the simulation is independent of b, meaning that $\Pr[S_1] = {}^1/_2$. Thus, we have $\Pr[S_0] \leq {}^1/_2 + \varepsilon_{\mathsf{zk}}(\kappa)$, which concludes the proof. □

5 Implementation Aspects and Evaluation

In this section we discuss some implementation aspects regarding instantiating our ring signature scheme. Moreover, we evaluate the efficiency of a concrete instantiation. Since we require simulation-sound extractable NIZK proof systems, we confirm that the Fiat-Shamir (resp. Unruh) transformed version of ZKB++ represents a suitable proof system in the ROM (resp. QROM). We again want to note that we were not able to include the ZKB++ construction due to space limitations, but refer the reader to [CDG+17] for the details.

5.1 Simulation-Sound Extractability of ZKB++

To instantiate our ring signature scheme using ZKB++, we first need to confirm that the NIZK proof system obtained by applying the Fiat-Shamir/Unruh transform to ZKB++ is in fact simulation-sound extractable. For the Unruh-transformed proof system, this was already shown in [CDG+17, Theorem 2] in the QROM, which is why we only focus on the Fiat-Shamir version. We base our argumentation upon the argumentation in [FKMV12]. What we have to do is to show that the FS transformed ZKB++ is zero-knowledge and provides quasi-unique responses in the ROM. We do so by proving two lemmas. Combining those lemmas with [FKMV12, Theorem 2 and Theorem 3] then yields simulation-sound extractability as a corollary.

Lemma 3. *Let Q_H be the number of queries to the random oracle H, Q_S be the overall queries to the simulator, and let the commitments be instantiated via a RO H' with output space $\{0,1\}^\rho$ and the committed values having min entropy ν. Then the probability $\epsilon(\kappa)$ for all PPT adversaries $\mathcal{A}$ to break zero-knowledge of κ parallel executions of the FS transformed* ZKB++ *is bounded by $\epsilon(\kappa) \leq {}^{s}/{2^\nu} + {}^{(Q_S \cdot Q_H)}/{2^{3\cdot\rho}}$.*

The lemma above was already proven for ZKBOO in [DOR+16]. For ZKB++ the argumentation is the same. We restate the proof below for completeness.

Proof. We bound the probability of any PPT adversary $\mathcal{A}$ to win the zero-knowledge game by showing that the simulation of the proof oracle is statistically close to the real proof oracle. For our proof let the environment maintain a list $\mathtt{H}$ where all entries are initially set to $\perp$.

Game 0: The zero-knowledge game where the proofs are honestly computed, and the ROs are simulated honestly.

Game 1: As Game 0, but whenever the adversary requests a proof for some tuple (x, w) we choose $\mathsf{e} \xleftarrow{R} \{0,1,2\}^\kappa$ before computing a and z. If $\mathtt{H}[(\mathsf{a}, x)] \neq \perp$ we abort and call that event E. Otherwise, we set $\mathtt{H}[(\mathsf{a}, x)] \leftarrow \mathsf{e}$.

Transition - Game 0 → Game 1: Both games proceed identically unless E happens. The message a includes 3 RO commitments with respect to H', i.e., the min-entropy is lower bounded by $3\cdot\rho$. We have $|\Pr[S_0] - \Pr[S_1]| \leq {}^{(Q_S \cdot Q_H)}/{2^{3\cdot\rho}}$.

Game 2: As Game 1, but we compute the commitments in a so that the ones which will never be opened according to e contain random values.

Transition - Game 1 → Game 2: The statistical difference between Game 1 and Game 2 can be upper bounded by $|\Pr[S_1] - \Pr[S_2]| \leq \kappa \cdot {}^{1}/{2^\nu}$ (for compactness we collapsed the s game changes into a single game).

Game 3: As Game 2, but we use the HVZK simulator to obtain $(\mathsf{a}, \mathsf{e}, \mathsf{z})$.

Transition - Game 2 → Game 3: This change is conceptual, i.e., $\Pr[S_2] = \Pr[S_3]$.

In Game 0, we sample from the first distribution of the zero-knowledge game, whereas we sample from the second one in Game 3; the distinguishing bounds shown above conclude the proof. □

Lemma 4. *Let the commitments be instantiated via a RO H' with output space $\{0,1\}^\rho$ and let $Q_{H'}$ be the number of queries to H', then the probability to break quasi-unique responses is bounded by ${}^{Q_{H'}^2}/{2^\rho}$.*

Proof. To break quasi-unique responses, the adversary would need to come up with two valid proofs $(\mathsf{a}, \mathsf{e}, \mathsf{z})$ and $(\mathsf{a}, \mathsf{e}, \mathsf{z}')$. The last message z (resp z') only contains openings to commitments, meaning that breaking quasi unique responses implies finding a collision for at least one of the commitments. The probability for this to happen is upper bounded by $Q_{H'}^2/2^\rho$ which concludes the proof. ☐

Combining Lemma 3 and Lemma 4 with [FKMV12, Theorem 2 and Theorem 3] yields the following corollary.

Corollary 1. *The FS transformed* ZKB++ *is simulation-sound extractable.*

5.2 Selection of Symmetric-Key Primitives

When instantiating our ring signature scheme using ZKB++, the selection of the underlying primitives is of importance for the actual signature sizes as well as the overall performance. As ZKB++'s proof size depends on the number of multiplication gates and the size of the operands, we require a OWF and a collision-resistant hash function with a representation as circuit, where the product of the multiplicative complexity and the number of bits required to store field elements is minimal. Note that for the OWF we can observe that, when instantiating it with a block cipher, only one plaintext-ciphertext pair per key is visible to an adversary. Hence, we have the same requirements as in [CDG+17], which is why we also choose LowMC [ARS+15, ARS+16] with a reduced data complexity to build the OWF. For the selection of the collision-resistant hash function we are presented with different options:

Standardized Hash Functions. SHA-256 or SHA-3 are the obvious choices for collision resistant hash functions. SHA-256's compression function requires around 25000 multiplication gates [BCG+14] and SHA-3's permutation even more with around 38400 gates [NIS15].

Sponge Construction with Low Multiplicative Complexity Ciphers. Using a block cipher with small multiplicative complexity as permutation in a sponge construction, e.g., using LowMC or MiMC [AGR+16], enables the construction of hash functions with similar security guarantees as SHA-256 and SHA-3, but with a significantly reduced multiplicative complexity. Using the numbers from [AGR+16], MiMCHash-256 requires 1293 multiplications with a field size of 1025 bits. LowMCHash-256 only requires a 1 bit binary field and 3540 AND gates[9]. Thus, a hash based on LowMC is a better candidate for our use case.

Finally we present signature sizes when instantiating our ring signature scheme with LowMC for both OWF and the hash function. Table 1 presents the maximal signature sizes for some choices of ring sizes and aiming at a 128 bit post-quantum security level and we compute them using the formulas from [CDG+17]. For the Fiat-Shamir-transformed proof system the involved proofs have a maximal size of $t \cdot (c + 2s + \log_2(3) + \ell \cdot m + 2i))$ bits and $t \cdot (c + 3s + \log_2(3) + 2\ell \cdot m + 2i)$ bits for the Unruh-transformed proofs, where t is the number of repetitions, c the size of the commitments, i the size of the input to the circuit, ℓ the size of the

[9] Numbers updated according to a personal discussion with Christian Rechberger.

underlying field, m the number of AND gates, and s the size of the seeds used to generate the random tapes. We use ZKB++ as instantiated in [CDG+17] and give the numbers for both the Fiat-Shamir and Unruh transformed proof system.

Ring size	$\|\sigma\|$ (FS/ROM)	$\|\sigma\|$ (Unruh/QROM)
2^k	$1335900 + 3213168 \cdot k$ bits	$2059476 + 4763688 \cdot k$ bits
2^5	2125 KB	3159 KB
2^{10}	4086 KB	6067 KB
2^{20}	8008 KB	11882 KB

Table 1: Signature sizes at the 128 bit post-quantum security level.

We note that Ligero [AHIV17], a recent NIZK proof system for general circuits, offers proofs logarithmic in the number of multiplication gates in the prime field case respectively in the number of AND and XOR gates in the case of binary fields, which would allow us to reduce the signature size significantly. However, to the best of our knowledge, it is unclear whether Ligero provides simulation-sound extractability.

6 Conclusions

In this this work we made some important steps towards establishing privacy-enhancing primitives which are solely built from symmetric-key primitives and therefore do not require any structured hardness assumptions. In our work, we followed a modular concept and first introduced a post-quantum accumulator with efficient zero-knowledge membership proofs of sublinear size. Besides the applications to logarithmic size ring signatures as we presented in this paper, we believe that our post-quantum accumulator construction with zero-knowledge proofs may well have broader impact in the construction of other (privacy-enhancing) protocols in the post-quantum setting.

Open Questions. In addition, we believe that our work also opens up quite some possibilities for further research.

First, in the context of privacy-enhancing protocols, it would be interesting to investigate how to extend our methods to obtain group signatures [CvH91], i.e., anonymous signatures that provide the possibility to re-identify anonymous signers by a dedicated party. We note that Dodis et al. [DKNS04] informally discuss that when adding ID escrow functionality to their ring signature scheme yields group signatures. Basically, the lattice-based construction of Libert et al. [LLNW16] can be considered as an instantiation of the former paradigm. The problem is that this paradigm requires IND-CCA2 secure public-key encryption, which does not exist given our constraints. In addition, it is well known [AW04, CG04] that group signatures in the static model by Bellare et al. in [BMW03] imply public-key encryption. This means that the best one could hope for would be a construction being secure in a weakened version of the Bellare et al. model. Work in this direction was earlier pursued by Camenisch and Groth [CG04], who showed how to construct group signature schemes in a weaker model from one-way functions and non-interactive zero-knowledge arguments. The question which remains open in our context is whether one can find instantiations without the requirement for structured hardness assumptions and providing the practical efficiency one would hope for, i.e., ideally instantiations which just require to prove statements with respect to a few evaluations of a block cipher.

Second, in the context of symmetric-key primitives, one may observe that—despite the recent trend to construct symmetric-key primitives with particularly low AND count—there is no practical application so far which would require collision resistant hash functions with particularly low AND count. Since our accumulator construction relies on collision resistant hash functions, our work may well also open up new fields of research in the symmetric-key community.

Acknowledgments. The authors have been supported by EU H2020 Project PRISMACLOUD, grant agreement n°644962. We thank Christian Rechberger for discussions on the choice of symmetric-key primitives, especially regarding the instantiation of hash functions using LowMC, as well as for providing us with updated LowMC instances.

References

[AGR+16] M. R. Albrecht, L. Grassi, C. Rechberger, A. Roy, and T. Tiessen. Mimc: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In *ASIACRYPT*, 2016.

[AHIV17] S. Ames, C. Hazay, Y. Ishai, and M. Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *CCS*, 2017.

[ARS+15] M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. In *EUROCRYPT*, 2015.

[ARS+16] M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. *IACR Cryptology ePrint Archive*, 2016:687, 2016.

[AW04] M. Abdalla and B. Warinschi. On the minimal assumptions of group signature schemes. In *ICICS*, 2004.

[BCC04] E. F. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In *CCS*, 2004.

[BCC+15] J. Bootle, A. Cerulli, P. Chaidos, E. Ghadafi, J. Groth, and C. Petit. Short accountable ring signatures based on DDH. In *ESORICS*, 2015.

[BCD+17] F. Baldimtsi, J. Camenisch, M. Dubovitskaya, A. Lysyanskaya, L. Reyzin, K. Samelin, and S. Yakoubov. Accumulators with applications to anonymity-preserving revocation. In *IEEE EuroS&P 2017*, 2017.

[BCG+14] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *IEEE SP*, 2014.

[Bd93] J. Benaloh and M. de Mare. One-way Accumulators: A Decentralized Alternative to Digital Signatures. In *EUROCRYPT*, 1993.

[BKM09] A. Bender, J. Katz, and R. Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology*, 22(1):114–138, 2009.

[BL07] E. Brickell and J. Li. Enhanced privacy id: a direct anonymous attestation scheme with enhanced revocation capabilities. In *WPES*, 2007.

[BLL00] A. Buldas, P. Laud, and H. Lipmaa. Accountable certificate management using undeniable attestations. In *CCS*, 2000.

[BMW03] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT*, 2003.

[BP97] N. Baric and B. Pfitzmann. Collision-free Accumulators and Fail-stop Signature Schemes Without Trees. In *EUROCRYPT*, pages 480–494, 1997.

[BPW12] D. Bernhard, O. Pereira, and B. Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In *ASIACRYPT*, 2012.

[CDG+17] M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, and G. Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In *CCS*, 2017.

[CDS94] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, 1994.

[CG04] J. Camenisch and J. Groth. Group signatures: Better efficiency and new theoretical aspects. In *SCN*, 2004.

[CGS07] N. Chandran, J. Groth, and A. Sahai. Ring signatures of sub-linear size without random oracles. In *ICALP*, 2007.

[CL02] J. Camenisch and A. Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In *CRYPTO*, 2002.

[CvH91] D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT*, 1991.

[Dam10] I. Damgård. On Σ-protocols. 2010. http://www.cs.au.dk/~ivan/Sigma.pdf.

[DHS15] D. Derler, C. Hanser, and D. Slamanig. Revisiting cryptographic accumulators, additional properties and relations to other primitives. In *CT-RSA*, 2015.

[DKNS04] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous identification in ad hoc groups. In *EUROCRYPT*, 2004.

[DOR+16] D. Derler, C. Orlandi, S. Ramacher, C. Rechberger, and D. Slamanig. Digital signatures from symmetric-key primitives. *IACR Cryptology ePrint Archive*, 2016:1085, 2016.

[DS16] D. Derler and D. Slamanig. Key-homomorphic signatures and applications to multiparty signatures and non-interactive zero-knowledge. *IACR Cryptology ePrint Archive*, 2016:792, 2016.

[FKMV12] S. Faust, M. Kohlweiss, G. A. Marson, and D. Venturi. On the non-malleability of the fiat-shamir transform. In *INDOCRYPT*, 2012.

[FS86] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, 1986.

[GK15] J. Groth and M. Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In *EUROCRYPT*, 2015.

[GMO16] I. Giacomelli, J. Madsen, and C. Orlandi. Zkboo: Faster zero-knowledge for boolean circuits. In *USENIX Security*, 2016.

[Gon17] A. González. A ring signature of size $\theta(\sqrt[3]{n})$ without random oracles. Cryptology ePrint Archive, Report 2017/905, 2017.

[IKOS09] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM J. Comput.*, 39(3):1121–1152, 2009.

[LLNW16] B. Libert, S. Ling, K. Nguyen, and H. Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT*, 2016.

[MCG08] C. A. Melchor, P. Cayrel, and P. Gaborit. A new efficient threshold ring signature scheme based on coding theory. In *PQCrypto*, 2008.

[MGGR13] I. Miers, C. Garman, M. Green, and A. D. Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *IEEE S&P*, 2013.

[MP17] M. S. E. Mohamed and A. Petzoldt. Ringrainbow - an efficient multivariate ring signature scheme. In *AFRICACRYPT*, 2017.

[MS17] G. Malavolta and D. Schröder. Efficient ring signatures in the standard model. In *ASIACRYPT*, 2017.

[NIS15] NIST. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. National Institute of Standards and Technology (NIST), FIPS PUB 202, U.S. Department of Commerce, 2015.

[RST01] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT*, 2001.

[Unr12] D. Unruh. Quantum proofs of knowledge. In *EUROCRYPT*, 2012.

[Unr15] D. Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *EUROCRYPT*, 2015.

[Unr16] D. Unruh. Computationally binding quantum commitments. In *EUROCRYPT*, 2016.