Post-Quantum Threshold Ring Signature Applications from VOLE-in-the-Head

James Hsin-yu Chiang
Aarhus University
Denmark
jachiang@cs.au.dk

Ivan Damgård
Aarhus University
Denmark
ivan@cs.au.dk

William R. Duro
University of Trento
Italy
durowilliam@gmail.com

Sunniva Engan
Norwegian University of Science and Technology
Norway
sunnibem@gmail.com

Sebastian Kolby
Aarhus University
Denmark
sk@cs.au.dk

Peter Scholl
Aarhus University
Denmark
peter.scholl@cs.au.dk

ABSTRACT

We propose efficient, post-quantum threshold ring signatures constructed from one-wayness of AES encryption and the VOLE-in-the-Head zero-knowledge proof system. Our scheme scales efficiently to large rings and extends the linkable ring signatures paradigm. We define and construct key-binding deterministic tags for signature linkability, that also enable succinct aggregation with approximate lower bound arguments of knowledge; this allows us to achieve succinct aggregation of our signatures without SNARKs. Finally, we extend our threshold ring signatures to realize post-quantum anonymous ledger transactions in the spirit of Monero. Our constructions assume symmetric key primitives only.

Whilst it is common to build post-quantum signatures from the one-wayness property of AES and a post-quantum NIZK scheme, we extend this paradigm to define and construct novel security properties from AES that are useful for advanced signature applications. We introduce key-binding and pseudorandomness of AES to establish linkability and anonymity of our threshold ring signatures from deterministic tags, and similarly establish binding and hiding properties of block ciphers modeled as ideal permutations to build commitments from AES, a crucial building block for our proposed post-quantum anonymous ledger scheme.

CCS CONCEPTS

• **Security and privacy** → **Cryptography**.

KEYWORDS

Post-Quantum Cryptography, Ring Signatures, Zero-Knowledge

1 INTRODUCTION

A 1-of-n ring signature permits a single party in a ring of possible signers to authenticate a message without revealing which specific key it controls. Introduced by Rivest *et al.* [39], ring signatures enable applications such as anonymous whistle-blowing; the ring anonymity grants deniability. Threshold ring signatures permit t-of-n parties in the ring (of size n) to sign a message without the public verifier or signer learning which keys were involved in signing; each of the t signing parties anonymously broadcast their signature contributions for public aggregation, while a verifier gets the guarantee that at least t signers produced the signature. A key challenge in constructing such schemes is to ensure that a single key cannot contribute multiple partial signatures towards the threshold t. This is ensured by establishing *linkability* of signatures of the same signing instance that are contributed by the same key. A common approach to adding linkability is through a deterministic tag added to each signature.

We observe that this paradigm of constructing signatures with determinstic, binding tags can be exploited for succinct aggregation; here, multiple parties contribute partial signatures that are publicly aggregated to produce the final signature string. For succinctness in the number of required signers, a naive approach would be to apply a post-quantum SNARK proof-of-knowledge of all partial signatures. However, this comes at a cost of an expensive prover. Instead, the application of pseudorandom deterministic tags that bind the signing key and message instance is amenable for succinct aggregation with approximate lower bound arguments of knowledge, only requiring calls to a random oracle. For a t-of-n threshold (ring) signature, this approach comes at the cost of requiring the aggregator to possess more than t signatures to convince the public verifier.

The application horizon of linkable ring signatures is extended to anonymous transactions by Monero [37], arguably the most widely deployed application of anonymity-preserving signatures. Each coin is a public key controlled by the coin owner and a commitment hiding the value of the coin. A transfer of coins from a sender to a recipient implies nullifying spent coins, and generating new coins with fresh public keys controlled by the recipient. Transaction anonymity from hiding the identity of coins that are being spent is achieved with a nullifier that collides if a coin is spent twice. A well-formed, anonymous transfer must ensure that the hidden value of the new coins equals that of the spent ones; in Monero, this is achieved with homomorphic commitments and range proofs.

We provide detailed comparisons with prior work in implementation sections sections 4.2.1 and 5.4, and refer to section 2 for a wider coverage of related work.

Contributions. In this work, we present a toolkit for constructions of post-quantum signatures[1] with advanced functionalities based on the VOLE-in-the-head (VOLEitH) paradigm and AES encryption.

[1] https://github.com/jachiang/PQ-Threshold-Ring-Sigs-from-VOLEitH

VOLEitH Disjunctions and PQ Ring Signatures. We describe a new approach to building proofs of disjunctive statements in VOLE-based zero-knowledge proofs, which is simpler and/or more efficient than prior methods. Using our disjunctive proofs and VOLEitH, we implement and benchmark ring signatures and proofs of large disjunctions to obtain the state-of-the-art in signature sizes and practical runtimes. Compared with prior work based on identical symmetric encryption schemes [22] (AES/AES-EM) and MPC-in-the-Head techniques, our signature sizes are 35-40% smaller, for all reported ring sizes at the 128-bit security level.

PQ Threshold Ring Signatures. To obtain threshold ring signatures, we take the approach of adding deterministic tags to our ring signatures, to achieve linkability. Intuitively, if each signature comes with a proof that the tag was computed from a pseudorandom function depending on the signing key, then any two uses of the same signing key should be publicly detectable. However, applying this idea in the MPCitH/VOLEitH setting requires some care. Firstly, pseudorandom tags alone are not enough for linkability, since if a corrupt signer with secret key k can find another key k' that is valid for the same public key, then it can produce two signatures with the same tag. We therefore need a stronger *key binding* property (or, collision resistance) for the public key generation algorithm. Secondly, for efficiency inside VOLEitH, we wish to instantiate both the public key derivation function and tag functions using AES. Since AES has a fixed 128-bit block size, it fails to give key binding at the 128-bit security level. To remedy this, and also allow for higher security levels, we show how to use multiple blocks to instantiate both the public keys and tags with the required properties. We formalise these properties and show that they are satisfied by our construction in the ideal cipher model.

We implement and benchmark our threshold ring signatures. They achieve state-of-the-art efficiency compared to prior works with practical runtimes ($< 60s$), which are focused on lattice-based techniques and MPCitH; for small rings of size 2^3 our signer/verifier runtimes of 12 ms improve on [10] with 90 ms runtimes. Even with large rings of 2^{12} our signer/verifier runtimes remain below 50 ms.

PQ Succinct Signature Aggregation. We show that our tag functions introduced for linkability can also be used to obtain practical, succinct proofs of knowledge sets as introduced in Approximate Lower Bounds Arguments [14] (ALBA). ALBA originally requires fully unique signatures, for which efficient post-quantum constructions are not known. With tag functions, we instead get the guarantee that one component of the signature is unique. We therefore define Expanded ALBA (ELBA), which allows proof of partially unique items; this enables succinct aggregation of our linkable signatures with tags that are unique for each key and signing instance. This can be used, for instance, to build large-scale, approximate multi-signatures with post-quantum security.

PQ Anonymous Ledger Transactions. As another application of our ring signatures and tag functions, we propose a post-quantum anonymous ledger scheme in the template of Monero; ring anonymity hides the coins being spent, and the consistency of input and output coin amounts is maintained with range proofs of sums in VOLEitH. Our proof-of-concept introduces the first PQ Monero-like scheme constructed from symmetric primitives alone.

2 RELATED WORK

Ring Signatures. Introduced by Rivest *et al.* [39] ring signatures allow a member of a group, known as the ring, to sign a message without revealing which particular member produced the signature. In contrast to the previously introduced group signatures [15], ring signatures have much stronger anonymity guarantees as they do not require a trusted group manager or any previous interactive setup. Various post-quantum ring signatures scheme have been proposed, including constructions from lattices [10, 19, 24, 33, 34, 42], MPC-in-the-Head approaches [22, 26, 30], and other assumptions [1, 9, 10]. Most notably, in the context of this work, Feneuil and Rivain [22] achieve highly efficient (non-linkable) ring signatures; as discussed in section 4.2.1 and shown in table 2.

Threshold Ring Signatures. In a natural generalisation of ring signatures Bresson *et al.* [11] proposed *threshold* ring signatures, allowing a t members of a group to produce a signature, while remaining anonymous within the group. The security modelling of [11] has been strengthened in subsequent works. Haque and Scafuro [29] modify security to allow quantum adversaries, and strengthen modelling to encompass active adversaries which deviate arbitrarily from specified procedures. In an alternative strengthening Munch-Hansen *et al.* [36] require contributions to the threshold to be non-interactive, and demand anonymity even against other signers, something which [29] does not achieve. A similar notion called inter-signer anonymity is introduced in [28].

Approximate Arguments and Short Certificates. The challenge of proving knowledge of large sets with small comunication is grounded in seminal contributions by Sipser and Gács [40], as well as Goldwasser and Sipser [27]. Their work highlights universal hash functions as powerful tools in demonstrating that small sets are unlikely to yield certain collisions. In contrast, the pigeonhole principle asserts that larger sets must inevitably produce them. This problem is particularly relevant when full disclosure of the elements would be impractical and costly. Similar goal are obtained for multisignature schemes, including weighted multisignatures and compact certificates. These constructions, such as those by Micali *et al.* ([35]) and the weighted threshold multisignature in Mithril [13], introduce methods to validate that parties holding sufficient weight have signed a message. These schemes rely either on specific algebraic properties, such as bilinear pairings([25], [17]), or require auxiliary structures like Merkle trees ([35]).

Anonymous Ledger Transactions. Monero [38] and subsequently proposed improvements [32, 43] extend the application realm of ring signatures to anonymous ledger transactions. In this model, coins are publicly observable on the ledger state, but hold hidden amounts controlled by the anonymous public key owner. Each transaction adds fresh coins to the ledger state, and proves ownership of a subset of ledger coins (in the chosen ring) which were spent. Importantly, the value of spent and fresh coins intended for the recipient must be balanced, requiring balance proofs. In the discrete-log setting, these can be achieved with homomorphic Pedersen commitments and range proofs [12]. The latter help prevent balance overflow. MatRiCT [20] introduces the first post-quantum construction of Monero-like protocol with lattice-based instantiations. A key challenge is to scale the proof sizes with the number of coins being spent; in the unspent coin model of Monero, the total

			Ring size						
Ring Signature	Linkable	Assumption	2^3	2^6	2^8	2^{10}	2^{12}	2^{20}	Security
This work	✓	AES128	9.84	9.91	10.05	10.09	10.18	10.53	NIST I
This work		AES128	4.78	4.86	4.99	5.03	5.12	5.47	NIST I
This work		AES128-EM	4.34	4.42	4.55	4.59	4.68	5.03	NIST I
Falafl [10]	✓	MSIS/MLWE	30	32	-	-	35	39	NIST I
Raptor [33]	✓	MSIS/MLWE	10	81	333	1290	5161	-	100 bit
Calamari [10]	✓	CSIDH	5	8	-	-	14	23	128 bit
TCitH [22]		AES128	7.87	7.90	7.94	8.02	8.18	9.39	NIST I
		AES128-EM	6.81	6.84	6.88	6.96	7.12	8.27	NIST I
		MQ over $\mathbb{F}_{256}$	4.30	4.33	4.37	4.45	4.60	5.62	NIST I
		SD over $\mathbb{F}_{256}$	7.37	7.51	7.96	8.24	8.40	10.09	NIST I

Table 1: (Linkable) ring signature sizes in KB

balance of a user tends to fragment over many coins. A transaction may need to spend many coins to achieve sufficient funds for the recipient. In follow-up work MatRiCT+ [18], a balance proof based on CRT-packing is introduced which enables proofs which are sublinear in the amount of coins spent.

3 PRELIMINARIES

Notation. For a set $\mathcal{X}$ we will use $x \leftarrow \mathcal{X}$ to denote sampling x uniformly from $\mathcal{X}$. For PPT algorithms $\mathcal{A}$, we let $b \leftarrow \mathcal{A}(a)$ denote the process of running $\mathcal{A}$ with input a until it provides an output b. If $\mathcal{A}$ has oracle access to a procedure O we will represent this by superscript $\mathcal{A}^O$. We let the adversary be implicitly stateful in security games where it is invoked multiple times. For string concatenation $||$, we let $||_{i=1}^n s_i$ be shorthand for $s_1|| \dots ||s_n$. For $i \in \mathbb{N}$ we let $\mathsf{bits}(i)$ be the unsigned bit representation of i. For $s \in \{0,1\}^\ell$ and $\ell' \leq \ell$, let $\mathsf{trun}_{\ell'}(s)$ be the function returning the last ℓ' bits of s.

3.1 Zero-Knowledge from VOLE-in-the-Head

We provide a gentle introduction to the VOLE-in-the-Head proof system introduced in [4]. Towards this, we first describe interactive zero-knowledge derived from random Vector Oblivious Linear Evaluation (VOLE) correlations held by prover and verifier introduced in Quicksilver [41]; this interactive proof system is later lifted to the non-interactive setting with the VOLE-in-the-Head technique [5].

Random VOLE Correlations. Let a random VOLE correlation over $\mathbb{F}_{2^k}$ of length ℓ be characterized by a random global key $\Delta \in \mathbb{F}_{2^k}$, random bits $u \in \mathbb{F}_2^\ell$, random $v \in \mathbb{F}_{2^k}^\ell$, and keys $q \in \mathbb{F}_{2^k}^\ell$ such that:

$$q_i = u_i \cdot \Delta + v_i \quad \text{for } i = 0, \dots, \ell - 1.$$

For instance i, the prover knows the values u_i and v_i, while the verifier holds q_i and Δ. A VOLE correlation commits the prover to bit u_i. This commitment is hiding since random v_i masks u_i, and binding since the cheating prover would still need to preserve the correlation on unknown Δ to open to a different message u'_i, namely $q_i = u'_i\Delta + v'_i$. This occurs with probability 2^{-k}.

Quicksilver Polynomials. Given random VOLE instances held by prover and verifier, Quicksilver (QS) provides a constraint system to prove statements expressed as arithmetic constraint circuits; fresh random VOLE instances are interpreted as polynomials of degree 1. The prover holds the random polynomial coefficients $P(\gamma) = u\gamma + q$, such that $u \in \mathbb{F}_2$ and $q \in \mathbb{F}_{2^k}$, and the verifier holds an evaluation of the polynomial at coordinate Δ, namely $q = P(\Delta) = u\Delta + v$. Here, the highest-degree coefficient is interpreted as the message committed to the VOLE instance. Interpreting fresh VOLE instances as quicksilver polynomials (held by the prover) and their evaluation at a secret point (held by the verifier), we write $[\![u]\!]^{(d)}$ to denote a QS polynomial of degree d which commits message u at its highest-degree coefficient. The QS proof system is as follows;

- $\mathsf{Input}(x)$: Given a fresh instance $P(\gamma) = u\gamma + v$ and $P(\Delta)$ held by P and V, respectively, P sends $\delta = x - u$ to V and updates $P(\gamma) \leftarrow x\gamma + v$. V then updates $q \leftarrow q + \delta\Delta$.
- $\mathsf{Add}([\![x]\!]^{(d_x)}, [\![y]\!]^{(d_y)})$: Let $P_x(\gamma)$ and $P_y(\gamma)$ be polynomials of degrees d_x and d_y respectively, with $d_x \geq d_y$. P updates $P_{x+y}(\gamma) \leftarrow p_x(\gamma) + p_y(\gamma)\gamma^{d_x - d_y}$, and V updates $q_{x+y} \leftarrow q_x + q_y\Delta^{d_x - d_y}$.
- $\mathsf{CMult}(c, [\![x]\!]^{(d)})$: P updates $P_{cx}(\gamma) = c \cdot P_x(\gamma)$, and V updates $q_{cx} \leftarrow c \cdot q_x$.
- $\mathsf{Mult}([\![x]\!]^{(d_x)}, [\![y]\!]^{(d_y)})$: P updates $P_{xy}(\gamma) = P_x(\gamma) \cdot P_y(\gamma)$, and V updates $q_{xy} \leftarrow q_x \cdot q_y$.

We highlight that linear operations maintain the degree of the QS polynomials, whilst multiplication of $[\![x]\!]^{(d_x)}$ and $[\![y]\!]^{(d_y)}$ implies a degree increase. In general, each multiplicative depth of the evaluated arithmetic circuit will double the polynomial degree.

- $\mathsf{CheckZero}([\![x]\!]^{(d)})$: Let P possess $P_x(\gamma) = a_0 + \dots + a_d\gamma^d$ of degree d; for $x = 0$, it follows that $a_d = 0$. P generates random polynomial $P_r(\gamma)$ of degree $d-1$ with coefficients in $\mathbb{F}_{2^\lambda}$ from $\lambda(d-1)$ fresh VOLEs, enabling V to compute q_r locally. P sends $P'_x = P_x + P_r$ (of degree $d-1$) to V, which asserts $q_x + q_r = P'_x(\Delta)$.

We note that CheckZero instances can be batched by applying a random linear function on all constraints $[\![z_i]\!]^{(d_i)}$ which must equal zero to verify. For the remainder of this work, we assume that only a single, batched instance of CheckZero is performed.

Proof Complexity. We note that the message complexity of the Quicksilver proof system is driven by the number of consumed random VOLEs (Figure 1). A VOLE correlation is consumed for each witness bit that is input by the prover, and for each additional degree of the final QS proof polynomial sent during CheckZero, k fresh VOLEs are consumed for masking.

To prevent the degree blow-up of QS polynomials, resulting in a larger prover messages during CheckZero, a degree reduction can be performed by allowing the prover to input a fresh witness $[\![xy]\!]^{(1)}$ for each multiplication of $[\![x]\!]^{(d)}$ and $[\![y]\!]^{(d)}$; prover and verifier then run $\mathsf{CheckZero}(\mathsf{Add}(\mathsf{Mult}([\![x]\!]^{(d)}, [\![y]\!]^{(d)}), -[\![xy]\!]^{(1)})$ to assert that $[\![xy]\!]^{(1)}$ indeed commits the multiplication of x and

y. The additional CheckZero does not impact the prover message complexity, as these can be batched.

VOLE-in-the-Head from GGM trees. We now show how VOLE correlations are generated with the all-but-one vector commitment technique from [2, 4]. Let the prover generate a GGM tree from a length-doubling PRG and then define $u = u_1 + \cdots + u_n$ and $v = -(1 \cdot u_1 + ... + n \cdot u_n)$ over pseudorandom leaves $u_{i \in [n]}$. Upon committing to the leaves of the tree, and then opening all leaves except one at position Δ chosen by the verifier, observe that the following correlation holds:

$$q = \sum_{i \in [n]} (\Delta - i) u_i = u\Delta + v.$$

The interactive protocol follows:

1. P sends vector commitment and input bits
2. V sends random challenge (for constraint batching)
3. P sends final QS proof polynomial
4. V sends random Δ
5. P sends vector comm. opening punctured at Δ

Note that the verifier can defer the sampling of Δ until its last message: at this point, all prover messages have been committed.

A NIZK is obtained from Fiat-Shamir security of the VOLEitH protocol. For security level λ, this naive construction requires the prover to open a vector commitment at $2^\lambda - 1$ indices; a practical construction is obtained by considering τ distinct tree instances with 2^k leaves each, such that $\lambda = \tau \cdot k$. To ensure that the same message is committed to each smaller instance, correction bits are sent by the prover; consistency is verified with a linear hash ([3]).

In this work, we implement our VOLEitH proofs with the all-but-one vector commitment optimization from [2]. Here, a single larger tree is interpreted as embedding τ smaller instances with an opening punctured at τ distinct indices; the authors of [2] show the active paths of these τ instances overlap with sufficient probability, resulting in smaller openings. A key parameter here is the internal node threshold T_{open} which cannot be exceeded during the opening phase. We parameterize this VOLEitH construction below and refer to [2] for further details.

VOLE-in-the-Head Proof Sizes. We provide an overview of VOLEitH proof sizes for all examples presented later sections in fig. 1. These implement the all-but-one-vector commitment construction from [2]; for security levels 128, 192 and 256, the sub-tree count τ and internal node threshold T_{Open} are set to $11, 16, 22$ and $102, 162, 245$ respectively. These parameters determine the linear relationship between random VOLEs required by the proof and the final proof size in fig. 1. We remind the reader that random VOLEs are required for the witness bits input by the prover, and for masking the d degree proof polynomial during the final, batched CheckZero instance.

3.2 Efficient proofs of AES.

In this work, we prove AES evaluations in VOLEitH with constraints introduced [2]. Recall that AES encryption proceeds in rounds, each consisting of SubBytes, ShiftRows, MixColumns and AddRoundKey operations, of which only SubBytes is non-linear. Recall from section 3.1 that the degree of commitment polynomials grows with 2^d, where d is the multiplicative depth of the evaluated constraint

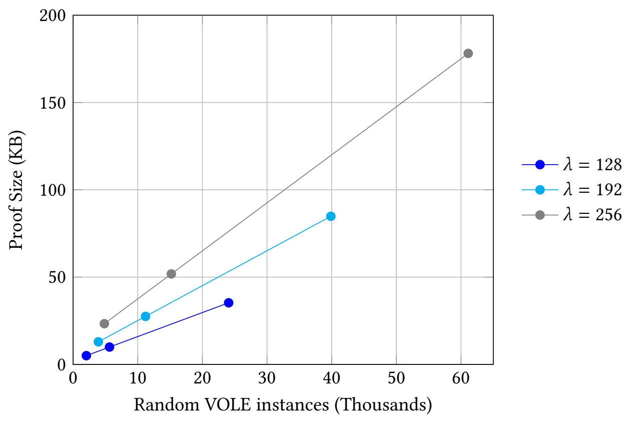

Figure 1: VOLEitH proof size is linearly dependent on the VOLE instances consumed in the proof. Data points for $\lambda = 128$ are detailed in example 4.1, example 5.10 and example 7.3; these are also shown for higher security levels above.

circuit. Evaluating all rounds of the encryption schedule naively therefore induces an exponential blow-up in the required VOLEs and resulting proof size. Instead, the 128 bit state following the ShiftRows operation for each round is committed as part of the witness. Since all operations other than SubBytes are linear, the AES state just before and after each SubBytes operation can be inferred from witness bits from rounds i and $i+1$. To show that SubBytes inversion are correctly evaluated, chunks of 8 bit elements over $\mathbb{F}_2$ in the committed AES state blocks are first lifted to elements in $\mathbb{F}_{2^8}$; to show inversion over bytes is well-formed, the following constraint is satisfied by each pair of bytes $[\![s_{\text{in}}]\!]^{(1)}, [\![s_{\text{out}}]\!]^{(1)}$ preceding and following a SubBytes operation in the AES schedule.

$$\begin{aligned} [\![z_2]\!]^{(2)} &= [\![s_{\text{in}}]\!]^{(1)} \cdot ([\![s_{\text{out}}]\!]^{(1)})^2 - [\![s_{\text{out}}]\!]^{(1)} \\ [\![z_3]\!]^{(2)} &= ([\![s_{\text{in}}]\!]^{(1)})^2 \cdot [\![s_{\text{out}}]\!]^{(1)} - [\![s_{\text{in}}]\!]^{(1)} \end{aligned} \tag{1}$$

In addition to enforcing well-formed inversion for non-zero elements, the constraints also ensure that both are 0 if either s_{in} or s_{out} are. While these constraints have degree 3 over $\mathbb{F}_{2^8}$, it is shown in [2] that squaring can be performed linearly over committed bits before lifting to $\mathbb{F}_{2^8}$, resulting in constraints of degree 2. All constraints over the AES schedule are batched by random linear combination resulting in a final CheckZero on a polynomial of degree 2 only. Constraints for the AES key schedule follow similarly. To prove an AES encryption $y = \mathsf{AES.ENC}(k, x)$ for public x and y, the VOLEitH prover inputs witness bits

$$[\![w]\!]^{(1)} = [\![w^{\text{key-sch}} | w^{\text{enc-sch}}]\!]^{(1)}$$

consisting of key $w^{\text{key-sch}} = k | w_1^{\text{key-sch}} | \dots | w_r^{\text{key-sch}}$ and encryption schedules $w^{\text{enc-sch}} = w_1^{\text{enc-sch}} | \dots | w_{r-1}^{\text{end-sch}}$ which encode intermediary state blockwise in each round (recall, x and y are public). Let $C^{\text{key-sch}}$ and $C_{x,y}^{\text{enc-sch}}$ denote (in shorthand) the batched inversion constraints shown in eq. (1) applied to both key schedule (induced by key k) and encryption schedule witness bits induced by $y = \mathsf{AES.ENC}(k, x)$.

$$\llbracket z^{\text{key-sch}} \rrbracket^{(2)} = C^{\text{key-sch}}(\llbracket w^{\text{key-sch}} \rrbracket^{(1)}) \tag{2}$$
$$\llbracket z^{\text{enc-sch}} \rrbracket^{(2)} = C^{\text{enc-sch}}_{x,y}(\llbracket w^{\text{enc-sch}} | w^{\text{key-sch}} \rrbracket^{(1)})$$

CheckZero on $\llbracket z^{\text{key-sch}} \rrbracket^{(2)}$ and $\llbracket z^{\text{enc-sch}} \rrbracket^{(2)}$ must hold.

Witness complexity. The AES block-cipher for security levels 128, 192, 256 are instantiated with 1, 2, 2 blocks of 128 bits each, 10, 12, 14 encryption rounds and key schedule periods of 16, 24, 16 respectively; the latter determines the byte spacing in sub-words operation in the key schedule. The witness bits required for proving the AES key schedule in Equation (2) are 448, 448, 672 for the respective security levels, implied by the aforementioned encryption round and key schedule period parameters.

The witness bits required for proving an AES encryption schedule are 1152, 2816, 3328 bits for the respective security levels. For example, at the 128 bit security level, this is determined by multiplying blocks (1), block size (128), rounds minus 1 (10-1); AES input and output blocks are public when used as a one-way-function for signatures.

3.3 Approximate Lower Bound Arguments

In recent work, Chaidos *et al.* [14] introduced Approximate Lower Bound Arguments (ALBA), which allow proving possession of a large set of elements while only sending a small, carefully chosen subset satisfying certain conditions with respect to the random oracle. This method significantly improves efficiency in terms of both communication and verification time relative to sending the entire set directly. However, this efficiency gain comes at the cost of introducing a gap between the size threshold for honestly generating a proof, denoted by n_p, and the threshold for an adversary, denoted by n_f. A larger ratio n_p/n_f leads to smaller proof sizes. Specifically, for 128-bit security and ratios $n_p/n_f = 60/40$, $66/33$ and $80/20$ the sets in the proof have 232, 136 and 68 elements respectively.

We wish to explore how ALBA may be applied to post-quantum signatures from a set of signers. The soundness of ALBA requires that the adversary holds a limited number of elements. However, in the context of signatures if the underlying scheme lacks uniqueness, an adversary could generate multiple signatures on the same message for any party it controls. For many post-quantum signature schemes, achieving uniqueness is not an option due to inherent randomization. To address this, Chaidos *et al.* [14] propose applying ALBA to verification keys rather than signatures, incorporating the corresponding signatures into the argument. This method has a notable drawback: the set of useful signatures becomes predictable, enabling adversaries to target parties with valuable signatures in denial-of-service attacks. In Section 6 we show that with very light modification ALBA schemes may still be used as long as the signatures in question have a computationally unique sub-string.

4 RING SIGNATURES FROM LARGE DISJUNCTIONS

Both ring signatures and anonymous ledger constructions can be realized with proofs of disjunctions over large rings (implying large anonymity sets). We show our techniques for proving large disjunctions in the VOLEitH setting, which allow the prover to obliviously select the active OR branch.

4.1 Efficient OR proofs

In the setting of VOLEitH, let $\llbracket z_1 \rrbracket^{(d)}, ..., \llbracket z_n \rrbracket^{(d)}$ denote the output wires of circuit evaluations $C_1(\llbracket w \rrbracket^{(1)}), ..., C_n(\llbracket w \rrbracket^{(1)})$ on witness w. As introduced in Sec. 3.1, the superscript (d) denotes the degree of the VOLEitH polynomials committing $z_1, ..., z_n$. Proving $\text{OR}(\llbracket z_1 \rrbracket^{(d)}, ..., \llbracket z_n \rrbracket^{(d)})$ then implies that $\llbracket z_i \rrbracket^{(d)}$ is 0 for at least one branch index $i \in [n]$. A naive approach is to simply multiply $\llbracket z_1 \rrbracket^{(d)}, ..., \llbracket z_n \rrbracket^{(d)}$ in a binary tree fashion. This results in proof complexity linear in n; such a circuit is of size $2n - 1$ and depth $\log(n)$. Evaluated on inputs $\llbracket z_1 \rrbracket^{(d)}, ..., \llbracket z_n \rrbracket^{(d)}$, this will result in an $O(n)$ increase of the proof size: recall from section 3.1 that the VOLEitH prover either commits to multiplication gate output wires (for degree reduction) or accepts a doubling of the commitment polynomial with each multiplicative circuit layer. The former results in the growth of the witness by $O(n)$, the latter grows the final proof polynomial by degree $O(n)$.

Our OR proof inspired from [23], applies a 1-hotvector selector to $\llbracket z_1 \rrbracket^{(d)}, ..., \llbracket z_n \rrbracket^{(d)}$, which obliviously outputs the input selected at the active index. Recall that a 1-hotvector is a bit array with a single active bit.

4.1.1 *OR proof with two 1-hotvectors.*

The prover commits to the active branch index $i \in [n]$ by first encoding the branch i in its base-$\sqrt{n}$ decomposed form, namely $(i_1, i_2) = \mathsf{decomp}_{\sqrt{n}}(i)$. This implies two 1-hotvectors $(\llbracket b_{1,1} \rrbracket^{(1)}, ..., \llbracket b_{1,\sqrt{n}} \rrbracket^{(1)})$ and $(\llbracket b_{2,1} \rrbracket^{(1)}, ..., \llbracket b_{2,\sqrt{n}} \rrbracket^{(1)})$, which have active bits at vector positions i_1, i_2 respectively. These are of length $\sqrt{n}$ and require prover witness bits sublinear in n. For example, consider $n = 2^4$ and active index 6; this implies a base-2^2 encoding of branch 6 in form of $(2, 2)$, requiring the prover activate the 2nd bit in both 1-hotvectors. Then, prover and verifier evaluate following constraints;

$$\forall i \in [n]: \quad \llbracket z_{\text{or},i} \rrbracket^{(d+2)} = \llbracket z_i \rrbracket^{(d)} \cdot \llbracket b_{1,i_1} \rrbracket^{(1)} \cdot \llbracket b_{2,i_2} \rrbracket^{(1)} \tag{3}$$
$$\text{where } (i_1, i_2) = \mathsf{decomp}_{\sqrt{n}}(i)$$

and execute $\mathsf{CheckZero}(\llbracket z_{\text{or},i} \rrbracket^{(d+2)})$ for each branch j; this must hold for the honest prover, since the product $b_{1,i_1} \cdot b_{2,i_2}$ activates a single branch and zeroes all others. The proof size is now sublinear in the number of OR branches n; each 1-hotvector consists of $\sqrt{n}$ witness bits and the polynomial underlying $\llbracket z_i \rrbracket^{(d)}$ for each branch i will be of degree $d + 2$; the additional VOLEs for the VOLEitH proof attributed to the OR over n branches is $2\sqrt{n} + 2\lambda$.

4.1.2 *OR proof with additional 1-hotvectors.*

We note that selection of the active OR branch selector with 1-hotvectors can be generalized to higher number of 1-hotvectors, enabling a tunable parameter to obtain optimal proof sizes for different number of OR branches. Let branch index $i \in [n]$ now be encoded by dim number of 1-hotvectors; the respective active bits are located at positions $(i_1, ..., i_{\text{dim}}) = \mathsf{decomp}_{\sqrt[\text{dim}]{n}}(i)$. Prover and verifier evaluate

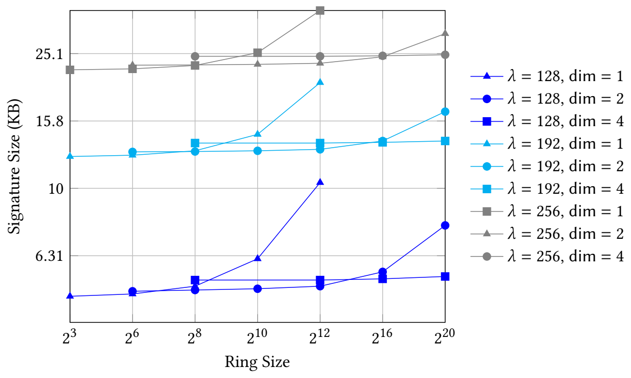

Figure 2: Ring Signature Size vs. Ring Size for various 1-hotvector dimensions (dim) and security levels (λ).

the following constraint

$$\forall i \in [n]: \quad [\![z_{\mathrm{or},i}]\!]^{(d+\mathrm{dim})} = [\![z_i]\!]^{(d)} \cdot \prod_{\mathrm{d}\in[\mathrm{dim}]} [\![b_{\mathrm{d},i_{\mathrm{d}}}]\!]^{(1)} \tag{4}$$
$$\text{where } (i_1, ..., i_{\mathrm{dim}}) = \mathsf{decomp}_{\sqrt[\mathrm{dim}]{n}}(i)$$

and assert $\mathsf{CheckZero}([\![z_{\mathrm{or},i}]\!]^{(d+\mathrm{dim})})$. This induces $\mathrm{dim} \cdot \sqrt[\mathrm{dim}]{n} + \mathrm{dim} \cdot \lambda$ consumption of VOLEs, where the first term is contributed by the 1-hotvector inputs to the witness and the latter by the degree increase of the final quicksilver proof polynomial. We illustrate this trade-off for various ring sizes in fig. 2.

4.1.3 Wellformedness of 1-hotvectors. Our protocol requires (1) the sum of all elements in the hotvector equal to 1 and that (2) at most one bit is active. The first constraint is trivially enforced by requiring the prover to input all but the last 1-hotvector bit, which is implied by the prior $\sqrt[\mathrm{dim}]{n} - 1$ elements. To prove (2), each element of each committed 1-hotvector $j \in \mathrm{dim}$ is first multiplied with its position $k \in [\sqrt[\mathrm{dim}]{n}]$;

$$\forall j \in \mathrm{dim}: \quad [\![\sigma_j]\!]^{(1)} = \sum_{k \in [\sqrt[\mathrm{dim}]{n}]} [\![b_{j,k}]\!]^{(1)} \cdot k$$

Then, the following constraint is applied over each σ_j and committed 1-hotvector b_j;

$$\forall j \in \mathrm{dim}: \quad \forall k \in \sqrt[\mathrm{dim}]{n}: \quad [\![z^{\text{1-htvc}}_{j,k}]\!]^{(2)} = ([\![\sigma_j]\!]^{(1)} - k) \cdot [\![b_{j,k}]\!]^{(1)} \tag{5}$$

Each $z^{\text{1-htvc}}_{j,k}$ is satisfied iff $\sigma_j = k$ or $b_{j,k} = 0$. For a malformed j'th 1-hotvector with multiple active bits, σ_j cannot equal the position of any of the individual active bits.

In contrast to [23], our constraints for well-formed 1-hotvectors do not require the prover to explicitly provide the active bit position for each 1-hotvector as part of the witness, saving $O(\mathrm{dim} \cdot \log(\sqrt[\mathrm{dim}]{n}))$ proof bits and simplifying the proof.

4.2 Ring signatures from disjunction proofs.

We now follow the paradigm of constructing signatures from a public key function $P(k, x) = \mathsf{AES.ENC}(k, x) = y$. Further, let ring $R = ((x_1, y_1), ..., (x_n, y_n))$ be defined over n public keys; the signer with knowledge of key k_i satisfying $\mathsf{AES.ENC}(k_i, x_i) = y_i$ then commits to extended witness $[\![w]\!]^{(1)} \in \mathbb{F}_2^l$ as detailed in section 3.2, such that $[\![z_i^{\text{key-sch}}]\!]^{(1)} = C^{\text{key-sch}}([\![w]\!]^{(1)})$ and $[\![z_i^{\text{enc-sch}}]\!]^{(1)} = C^{\text{key-enc}}_{x_i,y_i}([\![w]\!]^{(1)})$ are satisfied. A ring signature is then a FS signature derived from the VOLEitH proof of

$$[\![z_i]\!]^{(d)} = \mathsf{OR}(C_{x_1,y_1}([\![w]\!]^{(1)}), ..., C_{x_n,y_n}([\![w]\!]^{(1)})) \tag{6}$$

Here, we note that the witness w must also commit to 1-hotvectors encoding the active branch i; as shown in fig. 2, the dimension of 1-hotvectors that minimizes the proof size will differ for chosen security levels and ring sizes.

Example 4.1. We detail the parameterization and resulting signatures size for our implementation based on VOLEitH proofs of disjunctions for a ring size of 2^{10} and security level 128; the prover commits 2 separate 1-hotvectors of size $\sqrt{2^{10}}$ which encode the active branch; for chosen ring size of 2^{10}, the choice of 1-hotvector dimension of 2 is optimal (Figure 2). The prover further inputs following witness bits required to prove the public key function (eq. (2)) for the chosen active branch.

- 1x AES encryption schedule: 1152
- 1x AES key schedule: 448
- 1-hotvector bits: 2 x 31 (rounded to 8 bytes)

The 1664 witness bits consume the same number of random VOLE instances in input protocol (section 3.1). The final proof polynomial degree in eq. (6) is 4, since the batched key schedule and encrypytion schedule constraints are multiplied with an element from each 1-hotvectors; masking the QS proof polynomial in the final CheckZero protocol requires 3 $\mathbb{F}_{2^{128}}$ elements lifted from 3×128 random VOLE instances. The proof consumes $2048 = 1664 + 3 \times 128$ random VOLEs resulting in a proof size of 5034 bits (Figure 1).

4.2.1 Implementation & Benchmarking. As a stepping stone towards threshold ring signatures, we implement the ring signature scheme with public key functions instantiated with AES and AES-EM; the latter defines a public key function as $y = \mathsf{AES.ENC}(x, k) + k$, where the key k is a private input to the block cipher and x from public key x, y is a block cipher public input. This is a less conservative building block as AES, but permits fewer witness bits, as the key schedule is now public; further, it permits better comparison with prior art [22] with AES-EM as an underlying building block.

Our ring signature sizes represent an 35-40 % improvement over state-of-art, as shown in Table 1. We highlight [22] as a post-quantum ring signature scheme based on mpc-in-the-head; for a comparison based on the same conservative block-cipher assumptions (AES/AES-EM), our signatures sizes represent an 35-40 % improvement. Ring signature sizes in [22] based on non-standard MQ block ciphers are matched by our ring signatures based on AES-EM and slightly surpassed for larger rings. In principle, our scheme can also be based on other blockciphers; based on the improvements in signature sizes over [22] for AES/AES-EM, we conjecture similar improvements for MQ/SD constructions as well.

Whilst entirely practical, our implementation run-times shown in Figure 9 do not seem to match those reported in [22] which are based on less conservative MQ and SD block cipher assumptions; the authors report sub 20 ms signing time for ring size 2^{12} (Fig. 4 in [22]), whereas our AES128 implementation is bench-marked at 46 ms for signing and 31 ms verification; the authors do not report runtimes for AES/AES-EM constructions. The ring signature [22]) does not implement a general disjunction techinque over arbitrary circuits; rather, it implements an oblivious selection of public keys, over which the encryption schedule is proven. We conjecture that this trade-off of generality for application-specific efficiency is a key source of the runtime discrepancy.

Benchmarks in this work are performed on a third generation Intel Xeon Scalable machine featuring 16 threads and 32 GB of memory; our implementations do not exploit multi-threading support of the underlying system. We implement our threshold ring signatures by extending the VOLEitH implementation of [2] written in C; this requires adding the support of our public key and tag functionalities (section 3.1), support for large disjunctions (section 4.1) and quicksilver polynomials of higher degree (section 3.1). The source code is made available as part of this submission and will be open sourced.

5 THRESHOLD RING SIGNATURES

In the threshold ring signature setting, it does not suffice to aggregate multiple ring signatures, as ring anonymity permits a single key to generate multiple signatures without detection. Thus, we require ring signature *linkability* in the threshold setting. Our approach is to involve a tag function $T : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$, which binds a signing key and signing instance to a deterministic, pseudorandom tag. Colliding tags prevent a single signing key from contributing multiple partial signatures towards threshold t.

For each partial signature on m by key k in ring R, the signer publishes $\tau = T(k, h(m, R))$ and extends the VOLEtiH proof to satisfy constraint C^T, which outputs 0 iff the tag is well-formed and consistent with the active key and signing instance. We instantiate a post-quantum tag function with block ciphers modeled as ideal permutations, supporting the heuristic instantiation with AES; towards this goal, we highlight that both public key function P and tag function T need to be carefully constructed from multiple blocks to maintain security. In the presence of tags, the public key function must now satisfy *key-binding* (Eq. 8) in addition to one-wayness; the tag function must satisfy *pseudorandomness* (Eq. 9) to preserve signer anonymity.

5.1 Deterministic tags

For the tag τ to be unique we must ensure that a public key is (computationally) binding with respect to the secret key. Otherwise, it may be possible to produce signatures with distinct tags τ, τ' by finding secret keys $\mathsf{sk} \neq \mathsf{sk}'$ where $y = P_{\mathsf{sk}}(x) = P_{\mathsf{sk}'}(x)$. In other words, we require collision resistance for P as a function of sk and a fixed x.

Throughout this section we will explore various instantiations of P and T in the ideal cipher model. This supports the heuristic instantiation of permutations with AES.

5.1.1 Security properties. Let π be a family of independent random permutations $\pi_k : \{0,1\}^\ell \to \{0,1\}^\ell$ for every $k \in \mathcal{K}$, where the oracles allow querying the permutation $\pi_k(\cdot)$ and its inverse $\pi_k^{-1}(\cdot)$.

Public key function: A function $P^\pi : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, that is:

One-way, for all x, given $x, y = P^\pi(k, x)$ it should be hard to find k,

$$\Pr\left[\begin{matrix} k \leftarrow \mathcal{K}; \\ k^* \leftarrow \mathcal{A}^\pi(x, y) \end{matrix} \ : \ P^\pi(k,x) = P^\pi(k^*, x)\right] = \mathsf{negl}(\lambda). \quad (7)$$

Key binding, it should be hard to find two keys preserving the mapping of an input,

$$\Pr\left[(x, k, k') \leftarrow \mathcal{A}^\pi \ : \ \begin{matrix} (k \neq k') \wedge \\ (P^\pi(k,x) = P^\pi(k', x)) \end{matrix}\right] = \mathsf{negl}(\lambda). \quad (8)$$

Tag function: A function $T : \mathcal{K} \times \mathcal{X} \times \{0,1\}^* \to \mathcal{T}$, that is pseudorandom: for $x \leftarrow \mathcal{X}, k \leftarrow \mathcal{K}$ a PPT adversary has negligible advantage in distinguishing $T(k, x, \cdot)$ from a random function R given oracle access and $x, y = P^\pi(k, x)$,

$$\left|\Pr[\mathcal{A}^{T^\pi(k,x,\cdot),\pi}(x,y) = 1] - \Pr[\mathcal{A}^{R(\cdot),\pi}(x,y) = 1]\right| \quad (9)$$
$$= \mathsf{negl}(\lambda).$$

When constructing T we will employ a compressing hash function $H : \{0,1\}^* \to \{0,1\}^{\ell_\mathsf{H}}$ which is collision resistant,

$$\Pr\left[H \leftarrow \mathcal{H}; (v, v') \leftarrow \mathcal{A} \ : \ H(v) = H(v')\right] = \mathsf{negl}(\lambda).$$

5.1.2 Security for single blocks. Consider functions P, T, given access to a family of random permutation oracles π, using H with $\ell_\mathsf{H} = \ell$.

$$P^\pi : \mathcal{K} \times \{0,1\}^\ell \to \{0,1\}^\ell, \quad T^\pi : \mathcal{K} \times \{0,1\}^\ell \times \{0,1\}^* \to \{0,1\}^\ell$$

$$P^\pi(k, x) = \pi_k(\mathsf{H}(0||x)), \text{ and } T^\pi(k, x, m) = \pi_k(\mathsf{H}(1||x||m)). \quad (10)$$

The function P is one-way when π is instantiated by a secure PRP following [6, Lemma 7]. We may argue that P is key binding.

We consider an adversary $\mathcal{A}$ making a bounded number of permutation queries q. In general the adversary may output (x, k, k') where it has not queried $\pi_x(k)$ or $\pi_x(k')$. Throughout the remainder of this section it will be convenient to assume that the outputs produced by the adversary have always been queried to the oracles, as this avoids having to treat this as a special case. More formally, one could simply construct an adversary $\mathcal{A}'$ which runs $\mathcal{A}$ and then queries $\pi_x(k)$ and $\pi_x(k')$, for a total of $q' = q + 2$ queries. We stress that this has no impact on our asymptotic results. It will also be convenient to assume that the adversary never repeats any query to the oracles, this is without loss of generality for

Lemma 5.1. *In the ideal cipher model the probability that an adversary making at most q queries breaks key binding (8) for the public key function P^π (10) is bounded by $O(q^2/2^\ell)$ given that $q \leq c \cdot 2^\ell$ for some constant fraction $c \in (0, 1)$.*

Proof. Consider an adversary having made a sequence of i queries $(k_1, x_1, y_1) \ldots (k_i, x_i, y_i)$ s.t. $y_j = \pi_{k_j}(x_j)$. We wish to bound the probability that the next query results in a collision. For a query $y = \pi_k(x)$ or $x = \pi_k^{-1}(y)$ this occurs if there was a previous query (k_j, x, y) where $k_j \neq k$. For any fixed previous query (k_j, x_j, y_j) where $k \neq k_j$, the probability of a collision is at most $1/(2^\ell - i + 1)$; this is the case where all other previous queries (k_r, x_r, y_r) have $k_r = k_j$ and $x_r \neq x$ (or $y_r \neq y$ in the case of π^{-1}). Union bounding across all queries $j \in [q]$, the probability the adversary breaks key binding is at most,

$$\sum_{i=1}^{q} \frac{i}{2^\ell - i + 1} \le \frac{1}{2^\ell - q + 1} \sum_{i=1}^{q} i = \frac{q(q+1)}{2(1-c)2^\ell} = O(q^2/2^\ell),$$

when q is at most a constant fraction of 2^ℓ. □

The above bound explicitly requires the number of queries be sublinear in the block size and gives no guarantees when $q \approx 2^\ell$. In fact, we cannot hope for any security against an adversary making more than $2 \cdot 2^\ell$ queries. Let π, π' be two random permutations over n values. There are $n!$ choices of a permutation over n values. For a fixed π, there are only $(n-1)!$ choices for π' where $\pi'(x) \neq \pi(x)$ for all $x \in [n]$. This may be seen as $\pi'(1)$ permits $(n-1)$ choices subject to $\pi'(1) \neq \pi(1)$, and fixing the first value $y_1' = \pi'(1)$ there are only $(n-2)$ choices for $y_2' = \pi'(2) \neq \pi(2)$, and so on. The probability that π' has no collisions with π is therefore $(n-1)!/n! = 1/n$. This would mean an adversary querying two complete permutations over $\{0,1\}^\ell$ would in fact succeed in finding a collision with high probability: $(1 - 2^{-\ell})$.

Lemma 5.2. *In the ideal cipher model the advantage of an adversary making at most q π queries and t T queries in the PRF game (9) for T, P (10) is bounded by $\epsilon_{\mathrm{COL}}^{\mathsf{H}} + q/|\mathcal{K}| + t(t-1)/2^{\ell+1}$, where $\epsilon_{\mathrm{COL}}^{\mathsf{H}}$ is a bound on the probability of finding a collision in* H.

Proof. We follow a sequence of hybrids. Recall we assume w.l.o.g. that the adversary never makes the same query twice.

- $\mathcal{H}_0$: Run the PRF game for P, T and pass on the output of $\mathcal{A}$.
- $\mathcal{H}_1$: As previous, except if the adversary queries $m \neq m'$ to $T(k, \cdot)$ where $\mathsf{H}(1||x||m) = \mathsf{H}(1||x||m')$ output 0 directly. If the adversary queries m where $\mathsf{H}(0||x) = \mathsf{H}(1||x||m)$ to $T(k, \cdot)$ also output 0 directly.
- $\mathcal{H}_2$: As previous, but rather than sampling $T(k,m) = \pi_k(\mathsf{H}(1||x||m))$, set $T(k,m) = r$ for random r in $\{0,1\}^\ell$, sampling independently of π_k but without replacement.
- $\mathcal{H}_3$: As previous, but sample $T(k,m) = r \leftarrow \{0,1\}^\ell$ with replacement. This is identical to the true random case.

The behaviour of $\mathcal{H}_0, \mathcal{H}_1$ is different exactly when $\mathcal{A}$ provides a collision for H.

$$\epsilon_{\mathrm{COL}}^{\mathsf{H}} \ge |\Pr[\mathcal{H}_0 = 1] - \Pr[\mathcal{H}_1 = 1]|\,.$$

The adversaries view in $\mathcal{H}_1$ and $\mathcal{H}_2$ is identical if the adversary does not query π_k. Let $\mathcal{K}'$ be the set of keys k' where $\pi_{k'}$ has been queried. If the adversary has not yet queried the correct key, then all $k^* \in \mathcal{K} \setminus \mathcal{K}'$ are equally likely to be the correct key.

$$\frac{q}{|\mathcal{K}|} \ge |\Pr[\mathcal{H}_1 = 1] - \Pr[\mathcal{H}_2 = 1]|\,.$$

By $\mathcal{H}_1$ we are guaranteed that π_k is always queried on distinct inputs for distinct m. The distribution of $T(k, \cdot)$ is independent of the oracles π. We may apply the standard PRP/PRF switching lemma with t queries, see [8, Lemma 1], to get

$$\frac{t(t-1)}{2^{\ell+1}} \ge |\Pr[\mathcal{H}_2 = 1] - \Pr[\mathcal{H}_3 = 1]|\,.$$

This gives, $\mathsf{Adv}_{P,T,\mathcal{A}}^{\mathsf{PRF}} \le \epsilon_{\mathrm{COL}}^{\mathsf{H}} + \frac{q}{|\mathcal{K}|} + \frac{t(t-1)}{2^{\ell+1}}$. □

For $\epsilon_{\mathrm{COL}}^{\mathsf{H}} = \mathsf{negl}(\lambda)$ the output space of H must be at least $\ell_{\mathsf{H}} = \ell \ge 2\lambda$.

5.1.3 *Stronger binding with multiple blocks.*

For single block public keys we have encountered a quadratic security loss in the number of permutation queries. The resulting bit-security is therefore essentially half the bit-length of the block size. Put differently, the block size for the public key must be twice the desired security parameter for security to hold. To increase security we may pursue one of two options, increasing the blocksize by using the Rijndael cipher rather than standardised AES,[2] or extending the public key to include multiple blocks such that collisions are harder to find. In this section we explore the latter option. The goal is to strengthen security by moving from a single block to n blocks. We must restrict the choice of input to prevent the adversary from attacking the construction block-by-block. We define P^π and T^π for multiple blocks as,

$$P^\pi : \mathcal{K} \to \{0,1\}^{n \cdot \ell}, \quad T^\pi : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^\ell,^{3}$$

$$P^\pi(k) = ||_{i=1}^{n} \pi_k(0 \ldots 0||\mathsf{bits}(i)), \text{ and } T^\pi(k,m) = \mathsf{CBC\text{-}MAC}_{f_k}^{n}(k, \mathsf{H}(m)). \tag{11}$$

where CBC-MAC (Figure 3) is instantiated with the PRF

$$f_k(x) = \mathsf{trunc}_{\ell-1}(\pi_k(1||x)).$$

Observe, by defining f_k as above we ensure separation of permutation inputs between P^π and T^π. For the sake of stronger key binding P no longer takes an argument x, this comes at the cost of weakening multi-user security, as an adversary given many public keys $P^\pi(k)$ for many keys k may recover one of the keys more efficiently than by attacking each public key individually.

The function P^π is one-way when π is instantiated by a secure PRP following [6, Lemma 7]. Perhaps surprisingly, this result is relatively non-trivial in the context of multiple blocks, as an adversary may find a key k' which is different to the actual key k but consistent with $y = P(k', x)$. Such a key would allow winning the OWF game, without breaking PRP security.

```
CBC-MAC$_f^n(x)$
1 :  / Given $f : \{0,1\}^w \to \{0,1\}^w$
2 :  parse $x = x_1||\ldots||x_n$
3 :  $c_0 \leftarrow 0^w$
4 :  for $i \in [m]$ :
5 :      $c_i \leftarrow f(c_{i-1} \oplus x_i)$
6 :  return $c_n$
```

Figure 3: The CBC-MAC algorithm for a fixed length input.

Lemma 5.3. *For an integer constant $n > 0$, the probability that an adversary making q queries breaks key binding (8) for the counter mode construction (11) with n blocks in the ideal cipher model is bounded by $O(q^2|\mathcal{X}|^{-n})$.*

See Appendix B.1 for the proof of Lemma 5.3.

[2]See the FAEST specification [6] for a deeper discussion of the use of Rijndael with alternative block sizes.
[3]Note, the functions P, T now no longer take an input x.

Lemma 5.4. *In the ideal cipher model the advantage of an adversary making at most q π queries and t T queries in the PRF game for T, P (10) is bounded as*

$$\mathbf{Adv}^{PRF}_{P,T,\mathcal{A}}(\lambda, q, t) \leq \epsilon^{\mathsf{H}}_{\mathrm{COL}} + \frac{q}{|\mathcal{K}|} + 6.5\frac{t^2n^2}{2^\ell}.$$

Proof. First we will argue that $f_k(x) = \mathsf{trunc}_{\ell-1}(\pi_k(1||x))$ is a PRF, for all PPT adversaries making at most q_f queries to f_k and random k, x,

$$\left|\Pr\left[\mathcal{A}^{\pi,f_k}(x, P^\pi(k,x)) = 1\right] - \Pr\left[\mathcal{A}^{\pi,R}(x, P^\pi(k,x)) = 1\right]\right|$$
$$\leq \frac{q}{|\mathcal{K}|} + \frac{q_f(q_f-1)}{2^{\ell+1}}.$$

This follows by two hybrids. First, when computing f_k replace π_k with a freshly sampled permutation π^* to get $f_k(x) = \mathsf{trunc}_{\ell-1}(\pi^*(1||x))$, also compute $P(k,x)$ using π^*. This is indistinguishable unless the adversary queries π_k. Second, replace π^* by a random function $r : \{0,1\}^\ell \rightarrow \{0,1\}^\ell$. At this point f_k is a random function, as all inputs $1||x$ are mapped to a uniform string over $\{0,1\}^\ell$ which is then truncated resulting giving a uniform string over $\{0,1\}^{\ell-1}$. Note, f_k is now independent of $P(k,x)$ as their inputs are disjoint. The change is indistinguishable by the PRP/PRF switching lemma [8, Lemma 1] with advantage $q_f(q_f-1)/2^{\ell+1}$ where q_f is the number of queries to f_k.

We may now use the PRF security of f_k to show PRF security of T. For random functions $R : \{0,1\}^{\ell'} \rightarrow \{0,1\}^{\ell'}$ and $R' : \{0,1\}^{m\cdot\ell'} \rightarrow \{0,1\}^{\ell'}$ Bellare *et al.* [7, Theorem 3.1] show,

$$\left|\Pr\left[\mathcal{A}^{\mathsf{CBC\text{-}MAC}^m_R}(1^\lambda) = 1\right] - \Pr\left[\mathcal{A}^{\pi,R'}(1^\lambda) = 1\right]\right| \leq \frac{3t^2n^2}{2^{\ell'}}.$$

By assumption H is collision resistant, thus we may assume that the adversary does not produce $m \neq m'$ such that $\mathsf{H}(m) = \mathsf{H}(m')$ with more than negligible advantage $\epsilon^{\mathsf{H}}_{\mathrm{COL}}$. If the adversary does not cause any collisions then all distinct queries to T result in distinct queries to $\mathsf{CBC\text{-}MAC}^m_{f_k}$. Combining the above bounds, using $q_f = tn$ and $\ell' = \ell - 1$ we may conclude,

$$\left|\Pr[\mathcal{A}^{T^\pi(k,\cdot),\pi}(x, P^\pi(k,x)) = 1] - \Pr[\mathcal{A}^{R(\cdot),\pi}(x, P^\pi(k,x)) = 1]\right|$$
$$\leq \epsilon^{\mathsf{H}}_{\mathrm{COL}} + \frac{q}{|\mathcal{K}|} + \frac{tn(tn-1)}{2^{\ell+1}} + \frac{3t^2n^2}{2^{\ell-1}} \leq \epsilon^{\mathsf{H}}_{\mathrm{COL}} + \frac{q}{|\mathcal{K}|} + 6.5\frac{t^2n^2}{2^\ell}$$

□

5.1.4 Summary of security. In Figure 4 we summarize the hardness of breaking key binding or pseudorandomness for the instantiations P and T. Due to the quadratic loss in the number of queries, single block approaches require a block-length of twice the desired security level.
In Figure 5 we propose appropriate parameters for the multi-block approach at various security levels.

Instantiation	Functions	Key binding	PRF
Single-block	$P(k,x) = \pi_k(\mathsf{H}(0\|\|x))$ $T(k,x,m) = \pi_k(\mathsf{H}(1\|\|m))$	$O(q^2/2^\ell)$	$\epsilon^{\mathsf{H}}_{\mathrm{COL}} + q/\|\mathcal{K}\| +$ $t(t-1)/2^{\ell+1}$
Multi-block	$P(k) = \|\|^n_{i=1}\pi_k(0\ldots0\|\|\mathsf{bits}(i))$ $T(k,m) = \mathsf{CBC\text{-}MAC}^n_{f_k}(k, \mathsf{H}(m))$	$O(q^2/2^{n\ell})$	$\epsilon^{\mathsf{H}}_{\mathrm{COL}} + q/\|\mathcal{K}\| +$ $6.5t^2n^2/2^\ell$

Figure 4: A summary of advantage bounds for P and T, for an adversary making q queries to π and t queries to $T(k,\cdot)$. We assume $\pi_k : \{0,1\}^\ell \rightarrow \{0,1\}^\ell$ is a random permutation drawn from the family π, and $\mathsf{H} : \{0,1\}^* \rightarrow \{0,1\}^{\ell_{\mathsf{H}}}$ is collision resistant. In the single block case $\ell_{\mathsf{H}} = \ell$, while in the multiblock case $\ell_{\mathsf{H}} = n(\ell-1)$.

Instantiation	λ ($\approx \log_2(q)$)	n	π	t
Multi-block	128	2	AES128	2^{29}
	192	3	AES192	2^{29}
	256	4	AES256	2^{28}

Figure 5: Required block-number and permutation for varying security levels, along with the maximum allowed number of tag evaluations t. We take $n = 2\log_2(q)/\ell$, where ℓ is the block length in bits. For AES $\ell = 128$. We compute t to ensure the quantity $6.5t^2n^2/2^\ell \leq 2^{-\lambda_s}$ for a statisitical security parameter $\lambda_s = 64$.

VOLEitH proofs of multi-block public key and tag functions. The prover provides witness bits for key, encryption (public key function) and the CBC-MAC (tag function). Note that the cbc encryption schedule will require an additional block state for each cipher block following the first, resulting from the chaining of encryption schedules.

$$[\![w]\!]^{(1)} = [\![w^{\mathsf{key\text{-}sch}}]\!]^{(1)}|[\![w^{\mathsf{enc\text{-}sch}}_1|\ldots|w^{\mathsf{enc\text{-}sch}}_n]\!]^{(1)}$$
$$|[\![w^{\mathsf{enc\text{-}sch\text{-}cbc}}_1|\ldots|w^{\mathsf{enc\text{-}sch\text{-}cbc}}_n]\!]^{(1)}$$

These must satisfy key schedule and encryption constraints with fixed input block $\mathsf{in}(i)$; here, let $C^{\mathsf{enc\text{-}sch\text{-}cbc}}$ denote the sbox inversion constraints (eq. (1)) on the encryption schedule witness bits of the cbc encryption applied to public input $x_1, \ldots, x_n$ and final block output y.

$$[\![z^{\mathsf{key\text{-}sch}}]\!]^{(2)} = C^{\mathsf{key\text{-}sch}}([\![w^{\mathsf{key\text{-}sch}}]\!]^{(1)}) \quad (12)$$
$$\forall i \in [n] : [\![z^{\mathsf{enc\text{-}sch}}_i]\!]^{(2)} = C^{\mathsf{enc\text{-}sch}}_{\mathsf{in}(i),y}([\![w^{\mathsf{enc\text{-}sch}}_i|w^{\mathsf{key\text{-}sch}}]\!]^{(1)})$$
$$[\![z^{\mathsf{enc\text{-}sch\text{-}cbc}}]\!]^{(2)} = C^{\mathsf{enc\text{-}sch\text{-}cbc}}_{x_1,\ldots,x_n,y}([\![w^{\mathsf{enc\text{-}sch\text{-}cbc}}_{1\ldots n}|w^{\mathsf{key\text{-}sch}}]\!]^{(1)})$$

5.2 Threshold ring signature definition

We recall the syntax and defintions of threshold ring signatures as set out in [36].

Definition 5.5 (Threshold ring signature, [36]). A tuple of PPT algorithms $\mathsf{TRS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Combine}, \mathsf{Verify})$ with syntax

$\mathsf{Setup}(1^\lambda) \rightarrow \mathsf{pp}$**:** On input security parameter 1^λ outputs public parameters pp.

$\mathsf{KeyGen}(\mathsf{pp}) \rightarrow (\mathsf{pk}, \mathsf{sk})$**:** On input public parameters pp outputs a public key, secret key pair $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Sign}(\mathsf{pp}, m, \{(j, \mathsf{pk}_j)\}_{j\in\mathcal{R}}, \mathsf{sk}_i) \rightarrow \sigma$ **:** On input public parameters pp; a message m; a ring of public keys $\{\mathsf{pk}_j\}_{j\in\mathcal{R}}$; and a secret key sk_i corresponding to a public key in the ring, outputs a signature σ.

$\mathsf{Combine}(\mathsf{pp}, \{\sigma_i\}_{i\in[t]}, t) \to \sigma$**:** Given public parameters pp; a set of t signatures $\{\sigma_i\}_{i\in[t]}$, outputs combined signature σ.

$\mathsf{Verify}(\mathsf{pp}, m, \sigma, \{(j, \mathsf{pk}_j)\}_{j\in\mathcal{R}}, t) \to 0/1$**:** On input public parameters pp; a message m; a ring of public keys $\{\mathsf{pk}_j\}_{j\in\mathcal{R}}$; and a threshold t, outputs accept (1) or reject (0).

is said to be a threshold ring signature (TRS) scheme, if it satisfies the definitions of correctness (Definition 5.6), unforgeability (Definition 5.7) and anonymity (Definition 5.8).

In our use of the TRS syntax we will suppress public parameters pp leaving this as an implicit input.

We require correctness of signatures. Note, we do not require perfect correctness as in [36], this weakening is necessary as for our scheme there exist signatures from distinct signers which cannot be combined if their well-formed tags collide.

Definition 5.6 (Correctness [36, Modified]). A threshold ring signature TRS scheme is said to be correct if for any message m, signer set and ring $\mathcal{S} \subseteq \mathcal{R}$, with $|\mathcal{S}| = t$, following the procedure: $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$; $\{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}()\}_{i\in\mathcal{R}}$; $\{\sigma_i \leftarrow \mathsf{Sign}(m, \{(j, \mathsf{pk}_j)\}_{j\in\mathcal{R}}, \mathsf{sk}_i)\}_{i\in\mathcal{S}}$; $\sigma \leftarrow \mathsf{Combine}(\{\sigma_i\}_{i\in\mathcal{S}}, t)$, it satisfies

$$\mathsf{Adv}^{corr}_{\mathsf{TRS},\mathcal{A}} = 1 - \Pr\left[\mathsf{Verify}(m, \sigma, \{(i, \mathsf{pk}_i)\}_{i\in\mathcal{R}}, t) = 1\right] = \mathsf{negl}(\lambda).$$

Intuitively a signature is a forgery for message m, ring $\mathcal{R}$, and threshold t, if the signature verifies and t is larger than the number of honest parties for which the adversary has seen a signature on m with ring $\mathcal{R}$, plus the number of corrupt parties in the ring.

Definition 5.7 (Unforgeability [36]). A threshold ring signature TRS scheme is said to be unforgeable if for any PPT adversary $\mathcal{A}$

$$\Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}^{\mathsf{Unforge}}_{\mathsf{TRS},\mathcal{A}}(1^\lambda)\right] = \mathsf{negl}(\lambda).$$

where $\mathsf{Game}^{\mathsf{Unforge}}_{\mathsf{TRS},\mathcal{A}}$ is defined in Figure 6.

```
Game^{Unforge}_{TRS,A,n}(1^λ)
1 : $\mathcal{U} \leftarrow [n], Q_{\mathsf{corr}} \leftarrow \emptyset, Q_{\mathsf{sig}} \leftarrow \emptyset$
2 : $\{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TRS.KeyGen}(1^\lambda)\}_{i\in[n]}$
3 : $(\sigma', m', \mathcal{R}', t) \leftarrow \mathcal{A}^{\mathsf{Corrupt,Register,Sign}}(\{\mathsf{pk}_i\}_{i\in[n]})$
4 : $\mathcal{A}$ wins iff
        $\mathsf{TRS.Verify}(m, \{(i, \mathsf{pk}_i)\}_{i\in\mathcal{R}'}, \sigma, t) = 1$,
        $\mathcal{R}' \subset \mathcal{U}$ and $|\mathcal{R}' \cap (Q_{\mathsf{corr}} \cup Q^{m',\mathcal{R}'}_{\mathsf{sig}})| < t$

Corrupt(i)
$Q_{\mathsf{corr}} \leftarrow Q_{\mathsf{corr}} \cup \{i\}$;
return $\mathsf{sk}_i$

Register(pk)
$Q_{\mathsf{corr}} \leftarrow Q_{\mathsf{corr}} \cup \{i\}, \mathcal{U} \leftarrow \mathcal{U} \cup \{i\}$
$\mathsf{pk}_i \leftarrow \mathsf{pk}$ for $i = |\mathcal{U}|$.

Sign(m, R, i)
if $i \notin \mathcal{U} \setminus Q_{\mathsf{corr}} \lor i \notin \mathcal{R}$ : return $\bot$
$Q^{m,\mathcal{R}}_{\mathsf{sig}} \leftarrow Q^{m,\mathcal{R}}_{\mathsf{sig}} \cup \{i\}$
$\sigma_i \leftarrow \mathsf{TRS.Sign}(m, \{(j, \mathsf{pk}_j)\}_{j\in R}, \mathsf{sk}_i)$
```

Figure 6: Threshold ring signature unforgeability [36].

Anonymity requires that an adversary should only have negligible advantage beyond random guessing in distinguishing the signatures produced by two honest parties in a ring, even if all other members of the ring are corrupt.

Definition 5.8 (Anonymity [36]). A threshold ring signature TRS scheme is said to have anonymity if for any PPT adversary $\mathcal{A}$

$$\left|\Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}^{\mathsf{Anon},0}_{\mathsf{TRS},\mathcal{A}}(1^\lambda)\right] - \Pr\left[\mathcal{A} \text{ wins } \mathsf{Game}^{\mathsf{Anon},1}_{\mathsf{TRS},\mathcal{A}}(1^\lambda)\right]\right| \leq \mathsf{negl}(\lambda).$$

where $\mathsf{Game}^{\mathsf{Anon},b}_{\mathsf{TRS},\mathcal{A}}$ is defined in Figure 7.

```
Game^{Anon,b}_{TRS,A}(1^λ)
1 : $\mathcal{U} \leftarrow [n], Q_{\mathsf{corr}} \leftarrow \emptyset, Q_{\mathsf{sig}} \leftarrow \emptyset$
2 : $\{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TRS.KeyGen}(1^\lambda)\}_{i\in[n]}$
3 : $(m, \mathcal{R}, \mathcal{U}, i_0 \in \mathcal{R}, i_1 \in \mathcal{R})$
        $\leftarrow \mathcal{A}^{\mathsf{Corrupt,Register,Sign}}(\{\mathsf{pk}_i\}_{i\in[n]})$
4 : $\sigma_{i_b} \leftarrow \mathsf{TRS.Sign}(m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, \mathsf{sk}_{i_b})$
5 : $b' \leftarrow \mathcal{A}^{\mathsf{Corrupt,Register,Sign}}(\sigma_{i_b})$
6 : $\mathcal{A}$ wins iff
7 :     $b = b' \land (\forall \beta \in \{0,1\} : i_\beta \notin Q^{m,\mathcal{R}}_{\mathsf{sig}} \land i_\beta \notin Q_{\mathsf{corr}})$
```

Figure 7: Threshold ring signature anonymity [36]. The oracles given to the adversary are defined as in Figure 6.

5.3 Threshold ring signature construction

For threshold ring signatures, we wish to combine a set of t signatures for the same message and ring. This procedure should be public to avoid any additional interaction which could compromise the anonymity of the signers. A simple approach one might take would be to just concatenate the set of ring signatures. However, following this approach there would be no guarantee that a sequence of t signatures originated from distinct members of the ring, and not just the same signer. To resolve this we may further augment the proof to include a substring or "tag" which is deterministically and provably derived from the secret key for a fixed message and ring. If two signatures have the same tag, then they are *linkable* and must be from the same signer and vice versa. With the addition of tags threshold ring signatures may be constructed by concatenation.

To construct our threshold ring signature scheme we will need a public key function and a tag function as described in Section 5.1. Furthermore we require a non-interactive zero-knowledge argument-of-knowledge for the relation,

$$\mathscr{R} = \left\{ \left(\phi = \begin{pmatrix} \mathsf{PK} = \{(i, x_i, y_i)\}_{i\in\mathcal{R}}, \\ m, \tau \\ w = k \end{pmatrix} \right) \middle| \begin{array}{l} \exists i \in [n] : \\ P(k, x_i) = y_i, \\ T(k, (\mathsf{PK}, m)) = \tau \end{array} \right\}$$

with completeness, zero-knowledge and weak simulation-extractability, see Section A.1.

We construct our threshold ring signature scheme in Figure 8. Removing Combine and Verify would give a linkable ring signature scheme, where the tags are computationally unique.

```
$\mathsf{Setup}(1^\lambda)$
    $\mathsf{pp_{NIZK}} \leftarrow \mathsf{NIZKAoK.Setup}(1^\lambda)$
    return $\mathsf{pp_{NIZK}}$

$\mathsf{KeyGen}(\mathsf{pp})$
    $x \leftarrow \mathcal{X};\ k \leftarrow \mathcal{K}$
    $y \leftarrow P(k, x)$
    return $(\mathsf{pk} = (x, y), \mathsf{sk} = k)$

$\mathsf{Sign}(m, \mathsf{PK}, \mathsf{sk}_i)$
    $\tau \leftarrow T(\mathsf{sk}_i, (\mathsf{PK}, m))$
    $\phi \leftarrow (\mathsf{PK}, m, \tau)$
    $\pi \leftarrow \mathsf{NIZKAoK.Prove}(\phi, \mathsf{sk}_i)$
    return $(\pi, \tau)$

$\mathsf{Combine}(\mathsf{PK}, \{\sigma_i\}_{i \in [t]}, t)$
    return $\sigma = \sigma_1 || \ldots || \sigma_t$

$\mathsf{Verify}(m, \sigma, \mathsf{PK}, t)$
    / Immediately reject if $\sigma$ does not contain $t$ items.
    parse $\sigma = ||_{i=1}^{t} (\pi_i, \tau_i)$
    if $\exists i \in [t], \mathsf{PVerify}(m, \sigma_i, \mathsf{PK}) = 0$ :
        return 0
    if $\exists i, j \in [t],\ i \neq j,\ \tau_i = \tau_j$ : return 0
    else : return 1

$\mathsf{PVerify}(m, \sigma, \mathsf{PK})$
    parse $\sigma = (\pi, \tau)$
    return $\mathsf{NIZKAoK.Verify}((\mathsf{PK}, m, \tau), \pi)$
```

Figure 8: Our threshold ring signature construction. Note, $\mathsf{PVerify}$ **gives the verification algorithm for a linkable ring signature scheme, where signatures with identical tags are linked. If we wish for signatures to be linked across messages signing may be modified such that $h \leftarrow T(\mathsf{sk}_i, \mathsf{PK})$.**

THEOREM 5.9. *The threshold ring signature scheme defined in Figure 8 is secure (Definition 5.6, Definition 5.8, Definition 5.7) if* $\mathsf{NIZKAoK}$ *has completeness, zero-knowledge and weak simulation-extractability, and P, T are pseudorandom, one-way and key binding.*

We prove security through lemmas B.1, B.2, B.3 in Appendix B.2.

5.4 Performance evaluation

5.4.1 Implementation & Benchmarking. We report threshold ring signature sizes in table 1, where the sizes of individual, linkable partial signatures are shown, and prover/verifier runtimes in fig. 9. These are performed on the machine specification reported in section 4.2.1. Our proof of disjunctions technique require prover and verifier to evaluate each OR branch, resulting in runtimes that must (at minimum) scale linearly with the ring size.

We argue that our prover and verifier run-times for threshold ring signatures are practical; even at 256 bit security, runtimes for ring sizes up to $\approx 2^{10}$ for security levels up to 256 bits remain practical. We note that the dimension of 1-hotvectors chosen to minimize signature sizes (fig. 2) has an impact on performance. This is evident for ring size 2^{16} in fig. 9, where the 4 dimension 1-hotvector encoding at 128 and 192 bits exceeds the 2 dimension 1-hotvector encoding at 256 bits. This results in faster runtimes at the 256 bit security level than at 192 bits.

5.4.2 Comparison to other post-quantum threshold ring signatures. We first discuss comparisons with prior art in table 1 with practical runtimes. Lattice-based Falafl [10] features partial threshold signatures which are sublinear in the ring size. The signature sizes of Falafl are about twice of ours at the 128 bit security level, and the authors only report signing times of 90 ms for small ring size of 2^3, which is orders of magnitude slower than our implementation (< 1 ms). Raptor [33] features smaller signature sizes for smaller rings $< 2^6$ and does not scale to larger rings.

Linkable	Assumption	Ring size 2^3	2^6	2^8	2^{10}	2^{12}	2^{16}
✓	AES128	12	12	13	17	47	830
✓	AES196	33	33	40	69	224	4710
✓	AES256	56	57	66	103	294	4074
	AES128	11	11	12	16	46	817
	AES192	28	29	34	64	224	4714
	AES256	46	47	51	91	279	4098
	AES128-EM	11	12	17	38	133	2031
	AES192-EM	27	32	52	149	550	8702
	AES256-EM	46	56	100	315	1044	16277

Figure 9: Max of signer and verifier runtimes (ms) for linkable ring signatures in this work.

Whilst Isogeny-based Calamari [10] features smaller partial signature sizes (with exception of very large rings of size 2^{20}), the authors only report signing times of 79 s for small rings (2^3); arguably, this is beyond the realm of practicality for current hardware platforms and applications.

We highlight the non-linkable ring signatures contributed by [22], based on the MPC-in-the-head with practical similarities to VOLEitH. In principle, our public key and tag functionalities could also be realized in the framework of [22]. Still, as discussed in section 4.2.1, our (non-linkable) ring signature implementations with AES/AES-EM provide heuristic evidence of VOLEitH as advantageous to [22] for practical applications of concern in this work.

Example 5.10. We analyze the size of our threshold ring signature at $\lambda = 128$ and ring size 2^{10}; as in the ring signature case of Example 4.1, both key schedule and encryption schedule bits are input by the prover. However, public key and tag functions now require two AES blocks at $\lambda = 128$ (Figure 5). The public key function is instantiated by two AES encryptions in parallel, whereas the tag function requires two AES blocks in CBC mode, requiring one additional 128 bit block of state to be input as witness;

- 1 AES key schedule: 448
- 2 AES enc schedules: $2 \times 1152 = 2304$
- 2 AES enc schedules in CBC-mode: $1152 + 1280 = 2432$
- 2 1-hotvectors: $2 \times 31 \approx 64$ (rounded to bytes)

As in our parameterization of ring signatures in example 4.1, the degree of the final QS polynomial remains 4 (for two 1-hotvectors), requiring a total of $5248 = 448 + 2304 + 2432 + 64$ witness bits and 3×128 random VOLE instances to mask the degree 4 proof polynomial; The VOLEitH proof consumes $5632 = 5248 + 3 \times 128$ random VOLEs resulting in a proof size of 9962 (Figure 1).

6 SUCCINCT MULTI-SIGNATURES

We begin by recalling an Approximate Lower Bound Argument scheme proposed by Chaidos *et al.* [14], leaving the formal definition to Appendix 6. Consider a binary hash function that for any input outputs 1 and with a some probability ε independent of other outputs. It follows that the probability that any input from a fixed set results in an output 1 grows with the size of the set. For an appropriate choice of ε one can ensure that a 1 will likely be found given n_p elements, and will only be found with smaller probability if fewer than n_f elements are known. Chaidos *et al.* [14]

observe that this disparity in success probability may be boosted by requiring particular hash outputs for longer sequences of elements.

ALBA Scheme [14]. Let S_p be the set of elements known to a prover, with $|S_p| \geq n_p$. The prover aims to convince a verifier that $|S_p| > n_f$, where $n_f < n_p$. Let H_1 and H_2 be hash functions that output 1 with probabilities $1/n_p$ and q, respectively.

$\mathsf{Verify}^{H_1,H_2}(t, s_1, \ldots, s_u)$ checks that a sequence of u elements to constitute a valid proof:

- $t \in [d]$ and for all $i \in [u]$: $H_1(t, s_1, \ldots, s_i) = 1$,
- and $H_2(t, s_1, \ldots, s_u) = 1$.

To construct a proof, the prover employs a depth-first search approach. For a chosen parameter d, the prover begins by evaluating $H_1(i, s)$ for $i \in [d]$ and $s \in S_p$, selecting the first pair for which H_1 outputs 1. For (i, s) with $H_1(i, s) = 1$ the prover then attempts to extend the sequence until $s' \in S_p$ is found s.t. $H_1(i, s, s') = 1$, backtracking if this fails. The prover proceeds like this until a sequence of length u is found. Loosely, increasing d allows improving completeness, while u and q tune the soundness error. For appropriate parameters, the proof size u is logarithmic, while completeness and soundness error are negligible, all with prover runtime[4] $O(n_p \cdot \lambda^2)$ in expectation, and $O(n_p \cdot \lambda^3)$ worst case. A more complex approach *et al.* [14] allows improving runtimes to $n_p + O(\lambda^2)$ in expectation and $n_p + O(\lambda^3)$ worst case.

For a given security parameter, the parameterization and resulting proof size of ALBA only depends on n_p/n_f;

$$u \geq \frac{\lambda_{sound} + \log \lambda_{comp} + 1 - \log\log e}{\log \frac{n_p}{n_f}};\ d \geq \frac{2u\lambda_{comp}}{\log e};\ q = \frac{2\lambda_{comp}}{d \log e}. \tag{13}$$

6.1 Expanded Approximate Lower Bound Arguments

We define Expanded Approximate Lower Bound Arguments (ELBA), extending ALBA to allow proofs of partially unique items. To this end we interpret every element s as consisting of two parts (σ, ρ), where σ will be the unique substring. The algorithms of ELBA are parameterized by a weight oracle expressing whether elements are valid. Throughout our exposition we restrict ourselves to the unweighted case, where all valid elements are of equal weight, i.e. the weight oracle must be binary $W : \{0,1\}^* \to \{0,1\}$. See [14] for a discussion of how weights may be introduced. We no longer care if the adversary given $s = (\sigma, \rho)$ with $W(s) = 1$ is able to derive $s' = (\sigma, \rho')$ with $W(s') = 1$ as the substring σ is common to both elements, these will be considered equivalent in the game. For notational convenience let $W(S) = |\{\sigma \mid \exists \rho, ((\sigma, \rho) \in S \wedge W(\sigma, \rho) = 1)\}|$ express the number of unique substrings of valid elements in a set S.

Definition 6.1. The tuple of PPT oracle algorithms (Prove, Verify, Read) taking a random oracle H and weight oracle W constitute an $(\lambda_{sound}, \lambda_{comp}, n_p, n_f)$-ELBA scheme if and only if they satisfy:

- Completeness: for all weight oracles W and all S_p such that $W(S_p) \geq n_p$,

$$\Pr[\pi \leftarrow \mathsf{Prove}^{H,W}(S_p); \mathsf{Verify}^{H,W}(\pi) = 1] \geq 1 - 2^{-\lambda_{comp}}.$$

  Furthermore, for all π where $\mathsf{Verify}^{H,W}(\pi) = 1$, then $W(s) = 1$ must hold for all $s \in \mathsf{ELBA.Read}^{H,W}(\pi)$ and $\mathsf{ELBA.Read}^{H,W}(\pi) \subseteq S_p$.

- Knowledge-soundness: there exists an algorithm Extract which for any PPT adversary $\mathcal{A}$ running in time T, with advantage

$$\begin{aligned}\mathsf{Adv}^{\mathsf{sound}}_{\mathsf{ELBA},\mathcal{A}} = \Pr[&H \xleftarrow{\$} \mathcal{H}, \pi \leftarrow \mathcal{A}^{H,W}(); \\ &\mathsf{Verify}^{H,W}(\pi) = 1],\end{aligned} \tag{14}$$

  letting $\varepsilon = \mathsf{Adv}^{\mathsf{sound}}_{\mathsf{ELBA},\mathcal{A}} - 2^{-\lambda_{sound}}$, then $S \leftarrow \mathsf{Extract}^{H,W}(\mathcal{A})$ runs in expected time $\mathrm{poly}(T, n_f, 1/\varepsilon)$ and $\Pr[W(S) > n_f] = 1$.

Note the above definition has significant similarities with Definition 3 of [14] for ALBA in the CRS model.

6.2 Constructing ELBA from ALBA

Given an ALBA scheme constructing an ELBA is rather straight forward, we simply input the unique substrings to the ALBA prover and appropriately redefine the weight oracle. For a formal construction see Figure 10.

```
Prove^{H,W}(S = {s_i = (σ_i, ρ_i)}_{i∈[n_p]})
  1 :  $\pi \leftarrow \mathsf{ALBA.Prove}^{H,W_S}(\{\rho_i\}_{i\in[n_p]})$
       / collect complete elements contained in the proof
  2 :  $R \leftarrow \{\, (\sigma^*, \rho^*) \in S \mid \sigma^* \in \mathsf{ALBA.Read}(\pi)\}$; return $(\pi, R)$

Verify^{H,W}(π, R)
  1 :  $\Sigma \leftarrow \mathsf{ALBA.Read}(\pi)$; return $(\Sigma \subset W_R \wedge \mathsf{ALBA.Verify}^{H,W_R}(\pi))$
```

Figure 10: The ELBA construction. We define $W_X(\sigma) = (\exists s = (\sigma, \cdot) \in X, W(s) = 1)$**. Note** ALBA.Read **simply returns the elements** σ **contained in the proof. Similarly,** $\mathsf{ELBA.Read}^{H,W}((\pi, R))$ **outputs an element** $s = (\sigma, \rho) \in R$ **for each** $\sigma \in \mathsf{ALBA.Read}^{H,W^*}(\pi)$ **if** $W(s) = 1$**.**

We will state the security of ELBA, the proof follows from the same reasoning as [14, Theorem 21], nonetheless a full proof can be found in Appendix B.3.

THEOREM 6.2. *The construction in Figure 10 is a* $(\lambda_{sound}, \lambda_{comp}, n_p, n_f)$*-ELBA scheme if* ALBA *is a* $(\lambda_{sound}, \lambda_{comp}, n_p, n_f)$*-ALBA scheme.*

Example 6.3. Assume we want to apply Telescope ALBA [Section 3][14] to our ELBA construction. As we do not require any additional structure on the signature the size of the proof can be directly evaluated by multiplying the size of one signature by u[5]. With $\lambda_{sound} = \lambda_{comp} = 128$ and $n_p = 2 \cdot n_f$ a sequence of least 136 elements is required (eq. (13)).

[4]Runtime is measured in the number of oracle queries. λ is the parameter for soundness and completeness.

[5]It could be smaller, as the sequence could potentially contain the same element multiple times

7 ANONYMOUS LEDGER TRANSACTIONS

One of the most wide-spread usage of ring-signatures is Monero [38], an anonymous ledger which leverages linkable ring signatures and Pedersen commitments in the discrete log setting to hide transaction amounts. Each coin is a tuple comm(v, r), pk; a commitment to the coin value v and a unique public key pk controlled by the coin owner. A transaction spends *input* coins and generates fresh *output* coins with (1) commitments holding the same total value as the input coins and (2) fresh public keys controlled by the recipient.

The ledger state consists of coins and pseudorandom nullifiers. The latter are bound to each spent coin, preventing a coin to be spent twice; nullifiers cannot be linked to specific coins. Let ring R define the anonymity set of coins in the ledger. A Monero-style transaction consists of;

- Input coins nullifiers
- Output coins commitments and public keys
- NIZK proof-of-knowledge of

$\mathcal{R}_1$ Secret keys that correspond to spent coins in R

$\mathcal{R}_2$ Well-formedness of input coin nullifiers

$\mathcal{R}_3$ Input coin amounts equal output coin amounts.

We contribute a post-quantum version of Monero from deterministic tags and commitments built from proving (multi-block) AES evaluations in zero-knowledge. We do not provide a full security treatment of our construction as this is beyond the scope of this work, and refer to [16] for a formal security analysis.

Compared to Monero, which employs discrete-log based ring signatures and bullet-proofs [12] to bound coin amounts and prevent inflation of the coin supply, our construction only assumes symmetric primitives. We construct commitments from block ciphers, which hide the value of coins in the ledger. Then, we introduce proof of addition without overflow, which help enforce the coin supply invariant for each transaction. The multi-block tag construction from section 5.1.3 serves as the coin nullifier.

7.1 Commitments from block ciphers

We show that key-binding eq. (8) and pseudorandomness eq. (9) properties are closely related to hiding and binding required to realize commitments from multi-block evaluations of a block cipher.

Consider a message space $m = m_1||...||m_n$ to be small relative to the block-cipher width ℓ, such that $|m| = 2^d \ll 2^{n\cdot\ell}$. Then, the commitment function is defined as follows, where π is instantiated from an AES block.

$$\begin{aligned} C^\pi &: \{0,1\}^d \times \mathcal{K} \to \{0,1\}^{n\cdot\ell}, \\ C^\pi(m, r) &= ||_{i=1}^n \pi_r(0\ldots0||\mathsf{bits}(i)||m_i) \end{aligned} \tag{15}$$

Similar to the multi-block public key function in Section 5.1.3, inputs to each block are fixed for all but the trailing message bits; for each block, only d/n input bits are freely chosen. The cipher-block key represents randomness sampled for the commitment. We state message hiding (eq. 16) and binding (eq. 17) and prove these for our multi-block commitment construction in Appendix B.4.

Commitment function: A function $C^\pi : \mathcal{K} \times \mathcal{M} \to \mathcal{Y}$, that is:

Message hiding, it should be hard to distinguish commitments of known messages.

$$\Pr\left[b \leftarrow \mathcal{A}^\pi(\mathsf{st}, c) \;\middle|\; \begin{array}{l} (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}^\pi() \\ b \leftarrow \{0,1\}, r \leftarrow \mathcal{K} \\ c = C^\pi(m_b, r) \end{array} \right] = \mathsf{negl}(\lambda). \tag{16}$$

Message binding, it should be hard to find commitments with different openings.

$$\Pr\left[(m, r, m', r') \leftarrow \mathcal{A}^\pi \;:\; \begin{array}{c} (m \neq m')\wedge \\ (C^\pi(m,r) = C^\pi(m',r')) \end{array} \right] = \mathsf{negl}(\lambda). \tag{17}$$

Lemma 7.1. *In the ideal cipher model the advantage of an adversary making at most q queries to π in the hiding game (16) for C^π (15) is bounded by*

$$\mathbf{Adv}^{Hiding}_{C,\mathcal{A}}(\lambda, q, t) \leq \frac{q}{|\mathcal{K}|} + \frac{q(q-1)}{2^{\ell+1}}$$

Lemma 7.2. *For an integer constant $n > 0$, the probability that an adversary making q queries breaks message binding (17) for multi-block commitment construction (15) with n blocks in the ideal cipher model is bounded by $O(q^2(2^{\ell-d/n})^{-n}) = O(q^2(2^\ell)^{-n})$.*

Proving knowledge of commitment opening. The VOLEitH prover must provide a witness

$$[\![w]\!]^{(1)} = [\![m_1|...|m_n|w^{\mathsf{key\text{-}sch}}|w_1^{\mathsf{enc\text{-}sch}}|\ldots|w_n^{\mathsf{enc\text{-}sch}}]\!]^{(1)}$$

consisting of a key schedule, n encryption schedules and committed message m, which can be interpreted as chunks of d/n bits. These must satisfy following key schedule and encryption constraints;

$$\begin{aligned} [\![z^{\mathsf{key\text{-}sch}}]\!]^{(2)} &= C^{\mathsf{key\text{-}sch}}([\![w^{\mathsf{key\text{-}sch}}]\!]^{(1)}) \\ \forall i \in [n] : [\![z_i^{\mathsf{enc\text{-}sch}}]\!]^{(2)} &= \\ &C_c^{\mathsf{enc\text{-}sch}}(\mathsf{prefix}(i)|[\![m_i]\!]^{(1)}|[\![w_i^{\mathsf{enc\text{-}sch}}|w^{\mathsf{key\text{-}sch}}]\!]^{(1)}) \end{aligned} \tag{18}$$

7.2 Proofs of addition without overflow

Let $[\![v]\!]^{(1)} = [\![v_1]\!]^{(1)}, \ldots, [\![v_d]\!]^{(1)}$ be the VOLEitH commitment of the bitwise encoding of coin value. For a well-formed anonymous transaction, we require the sum of input coin values to equal the sum of all output coin values. We must prove that each addition in these summations does not exceed $2^d - 1$ to prevent overflow. We restate bitwise addition on $v_a, v_b \in \{0,1\}^d$ below; operations are bitwise on elements of v_a, v_b.

(1) Input: $v_a^1, v_b^1 \in \{0,1\}^d$

(2) For $i \in [d]$:

  (a) $v_a^{i+1} \leftarrow v_a^i + v_b^i$

  (b) $v_b^{i+1} \leftarrow \mathsf{ShiftLeft}(v_a^i \cdot v_b^i)$

(3) Output: v_a^{i+1}

To prove a bitsum over witness bit vectors $[\![v_a]\!]^{(1)}$ and $[\![v_b]\!]^{(1)}$, the VOLEitH prover also inputs $[\![v_b^i]\!]^{(1)}$ for $i \in [2 : d+1]$ as witness (to mitigate the degree blow-up of the final proof polynomial); the

Scheme	PQ	Input Coins 2	20	50
This work	✓	35	257	627
MatRiCT+ [18]	✓	29	39	40
MatRiCT [20]	✓	110	310	610
Monero[38]		1.72	9.03	21.22

Figure 11: Transaction proof sizes (KB) with 2 output coins and ring size 2^4.

constraints satisfied by the prover are as follows; they are shown over committed bit vectors for brevity.

$$[\![z^{i+1}_{\text{add}}]\!]^{(1)} = [\![v^{i+1}_a]\!]^{(1)} - ([\![v^i_a]\!]^{(1)} + [\![v^i_b]\!]^{(1)}) \tag{19}$$
$$[\![z^{i+1}_{\text{carry}}]\!]^{(2)} = [\![v^{i+1}_b]\!]^{(1)} - \mathsf{ShiftLeft}([\![v^i_a]\!]^{(1)} \cdot [\![v^i_b]\!]^{(1)})$$

To prove that no overflow occurs during each addition, the prover also proves satisfaction of

$$[\![z^{i+1}_{\text{overflow}}]\!]^{(2)} = \mathsf{msb}([\![v^i_a]\!]^{(1)} \cdot [\![v^i_b]\!]^{(i)}) \tag{20}$$

7.3 Anonymous transactions

A transaction consists of a VOLEitH fiat-shamir signature, nullifiers (ℓ bits) for each spent input coin, and a fresh commitment ($n \cdot \ell$ bits) and public key ($n \cdot \ell$ bits) for each output coin. The VOLEitH proof signs nullifiers and output coins, and proves the following: (1) Commitment opening constraints from eq. (18) for each input/output coin (2) Public key ownership & tag constraints from eq. (12) for each spent input coin (3) Well-formed summation of input and output coin amounts with eq. (19) and eq. (20) for the balance proof.

Example 7.3. We illustrate the anatomy of a VOLEitH proof for an anonymous transaction with 2 input and 2 output coins with coin anonymity set of size 2^{10}; here, let coin amounts be encoded by 32 bits. For each of the 4 input and output coin commitments, the following bits are required to prove commitment opening; 448 bits for 1 AES key schedule, $2 \times 1152 = 2304$ bits for 2 AES encryption schedules, 32 bits for committing the coin value. The total witness bits input for all 4 commitments is $11136 = 4 \times (448 + 2304 + 32)$. For each spent input coins in ring R, proof of ownership and well-formed tags requires a total of $10496 = 2 \times (448 + 2304 + 2432 + 64)$ witness bits;

- 1 AES key schedule: 448
- 2 AES enc schedules: $2 \times 1152 = 2304$
- 2 AES enc schedules in CBC-mode: $1152 + 1280 = 2432$
- 2 1-hotvectors: $2 \times 31 \approx 64$ (rounded to bytes)

Finally, 2 summations are required to prove equality of input and output amounts; for coin values encoded as 32 bits, this implies $2048 = 2 \times 32 \times 32$ witness bits for the bitwise summation of coin amounts (eq. (19)). The total number of witness bits is $23680 = 10496 + 11136 + 2048$, and the degree 4 of the final QS polynomial is determined by the AES constraints of degree 2 that is increased by multiplicative selection via two 1-hotvector elements (fig. 2). The proof consumes a total of VOLEs $24064 = 23680 + 3 \times 128$ resulting in a proof size of $35'306$ bits (Figure 1).

As shown in fig. 11, our proof sizes scale quasi linearly with the number of spent input coins, and are larger than prior work with post-quantum security based on lattice assumptions. In MatRiCT+[18], proof sizes sublinear in the number of inputs are enabled with balance proofs based on CRT-packing. Our constructions for anonymous transactions are the first to rely on symmetric key primitives only.

ACKNOWLEDGEMENTS

This project was supported by an Academic Research Award from the SUI foundation and a collaboration grant from the DIGIT Aarhus University Centre for Digitalisation, Big Data and Data Analytics.

REFERENCES

[1] Alessandro Barenghi, Jean-Francois Biasse, Tran Ngo, Edoardo Persichetti, and Paolo Santini. 2022. Advanced Signature Functionalities from the Code Equivalence Problem. Cryptology ePrint Archive, Report 2022/710. https://eprint.iacr.org/2022/710

[2] Carsten Baum, Ward Beullens, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, and Peter Scholl. 2025. One tree to rule them all: Optimizing ggm trees and owfs for post-quantum signatures. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 463–493.

[3] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Christian Majenz, Shibam Mukherjee, Sebastian Ramacher, Christian Rechberger, Emmanuela Orsini, Lawrence Roy, et al. 2023. *FAEST: algorithm specifications*. Technical Report. Technical report, National Institute of Standards and Technology.

[4] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. 2023. Publicly verifiable zero-knowledge and post-quantum signatures from vole-in-the-head. In *Annual International Cryptology Conference*. Springer, 581–615.

[5] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. 2023. Publicly Verifiable Zero-Knowledge and Post-Quantum Signatures from VOLE-in-the-Head. In *Advances in Cryptology – CRYPTO 2023*, Helena Handschuh and Anna Lysyanskaya (Eds.). Springer Nature Switzerland, Cham, 581–615.

[6] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Christian Majenz, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, and Peter Scholl. 2023. *FAEST*. Technical Report. National Institute of Standards and Technology. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.

[7] Mihir Bellare, Joe Kilian, and Phillip Rogaway. 2000. The Security of the Cipher Block Chaining Message Authentication Code. *J. Comput. System Sci.* 61, 3 (2000), 362–399.

[8] Mihir Bellare and Phillip Rogaway. 2004. Code-Based Game-Playing Proofs and the Security of Triple Encryption. Cryptology ePrint Archive, Report 2004/331. https://eprint.iacr.org/2004/331

[9] Emanuele Bellini, Andre Esser, Carlo Sanna, and Javier A. Verbel. 2022. MR-DSS - Smaller MinRank-Based (Ring-)Signatures. In *Post-Quantum Cryptography - 13th International Workshop, PQCrypto 2022*, Jung Hee Cheon and Thomas Johansson (Eds.). Springer, Cham, Switzerland, Virtual Event, 144–169. https://doi.org/10.1007/978-3-031-17234-2_8

[10] Ward Beullens, Shuichi Katsumata, and Federico Pintore. 2020. Calamari and Falafl: Logarithmic (Linkable) Ring Signatures from Isogenies and Lattices. In *Advances in Cryptology – ASIACRYPT 2020, Part II (Lecture Notes in Computer Science, Vol. 12492)*, Shiho Moriai and Huaxiong Wang (Eds.). Springer, Cham, Switzerland, Daejeon, South Korea, 464–492. https://doi.org/10.1007/978-3-030-64834-3_16

[11] Emmanuel Bresson, Jacques Stern, and Michael Szydlo. 2002. Threshold Ring Signatures and Applications to Ad-hoc Groups. In *Advances in Cryptology – CRYPTO 2002 (Lecture Notes in Computer Science, Vol. 2442)*, Moti Yung (Ed.). Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA, 465–480. https://doi.org/10.1007/3-540-45708-9_30

[12] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. 2018. Bulletproofs: Short Proofs for Confidential Transactions and More. In *2018 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Francisco, CA, USA, 315–334. https://doi.org/10.1109/SP.2018.00020

[13] Pyrros Chaidos and Aggelos Kiayias. 2021. Mithril: Stake-based Threshold Multisignatures. Cryptology ePrint Archive, Report 2021/916. https://eprint.iacr.org/2021/916

[14] Pyrros Chaidos, Aggelos Kiayias, Leonid Reyzin, and Anatoliy Zinovyev. 2024. Approximate Lower Bound Arguments. In *Advances in Cryptology – EUROCRYPT 2024, Part IV (Lecture Notes in Computer Science, Vol. 14654)*, Marc Joye

and Gregor Leander (Eds.). Springer, Cham, Switzerland, Zurich, Switzerland, 55–84. https://doi.org/10.1007/978-3-031-58737-5_3

[15] David Chaum and Eugène van Heyst. 1991. Group Signatures. In *Advances in Cryptology – EUROCRYPT'91 (Lecture Notes in Computer Science, Vol. 547)*, Donald W. Davies (Ed.). Springer Berlin Heidelberg, Germany, Brighton, UK, 257–265. https://doi.org/10.1007/3-540-46416-6_22

[16] Cas Cremers, Julian Loss, and Benedikt Wagner. 2024. A Holistic Security Analysis of Monero Transactions. In *Advances in Cryptology – EUROCRYPT 2024*, Marc Joye and Gregor Leander (Eds.). Springer Nature Switzerland, Cham, 129–159.

[17] Sourav Das, Philippe Camacho, Zhuolun Xiang, Javier Nieto, Benedikt Bünz, and Ling Ren. 2023. Threshold Signatures from Inner Product Argument: Succinct, Weighted, and Multi-threshold. In *ACM CCS 2023: 30th Conference on Computer and Communications Security*, Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda (Eds.). ACM Press, Copenhagen, Denmark, 356–370. https://doi.org/10.1145/3576915.3623096

[18] Muhammed F Esgin, Ron Steinfeld, and Raymond K Zhao. 2022. MatRiCT+: More efficient post-quantum private blockchain payments. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1281–1298.

[19] Muhammed F. Esgin, Raymond K. Zhao, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. 2019. MatRiCT: Efficient, Scalable and Post-Quantum Blockchain Confidential Transactions Protocol. In *ACM CCS 2019: 26th Conference on Computer and Communications Security*, Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz (Eds.). ACM Press, London, UK, 567–584. https://doi.org/10.1145/3319535.3354200

[20] Muhammed F Esgin, Raymond K Zhao, Ron Steinfeld, Joseph K Liu, and Dongxi Liu. 2019. MatRiCT: efficient, scalable and post-quantum blockchain confidential transactions protocol. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 567–584.

[21] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. 2012. On the Non-malleability of the Fiat-Shamir Transform. In *Progress in Cryptology - INDOCRYPT 2012: 13th International Conference in Cryptology in India (Lecture Notes in Computer Science, Vol. 7668)*, Steven D. Galbraith and Mridul Nandi (Eds.). Springer Berlin Heidelberg, Germany, Kolkata, India, 60–79. https://doi.org/10.1007/978-3-642-34931-7_5

[22] Thibauld Feneuil and Matthieu Rivain. 2023. Threshold Computation in the Head: Improved Framework for Post-Quantum Signatures and Zero-Knowledge Arguments. Cryptology ePrint Archive, Report 2023/1573. https://eprint.iacr.org/2023/1573

[23] Thibauld Feneuil and Matthieu Rivain. 2023. Threshold Computation in the Head: Improved Framework for Post-Quantum Signatures and Zero-Knowledge Arguments. Cryptology ePrint Archive, Paper 2023/1573. https://eprint.iacr.org/2023/1573

[24] Phillip Gajland, Jonas Janneck, and Eike Kiltz. 2024. Ring Signatures for Deniable AKEM: Gandalf's Fellowship. In *Advances in Cryptology – CRYPTO 2024, Part I (Lecture Notes in Computer Science, Vol. 14920)*, Leonid Reyzin and Douglas Stebila (Eds.). Springer, Cham, Switzerland, Santa Barbara, CA, USA, 305–338. https://doi.org/10.1007/978-3-031-68376-3_10

[25] Sanjam Garg, Abhishek Jain, Pratyay Mukherjee, Rohit Sinha, Mingyuan Wang, and Yinuo Zhang. 2023. hinTS: Threshold Signatures with Silent Setup. Cryptology ePrint Archive, Report 2023/567. https://eprint.iacr.org/2023/567

[26] Aarushi Goel, Matthew Green, Mathias Hall-Andersen, and Gabriel Kaptchuk. 2022. Efficient Set Membership Proofs using MPC-in-the-Head. *Proceedings on Privacy Enhancing Technologies* 2022, 2 (April 2022), 304–324. https://doi.org/10.2478/popets-2022-0047

[27] Shafi Goldwasser and Michael Sipser. 1986. Private Coins versus Public Coins in Interactive Proof Systems. In *18th Annual ACM Symposium on Theory of Computing*. ACM Press, Berkeley, CA, USA, 59–68. https://doi.org/10.1145/12130.12137

[28] Abida Haque, Stephan Krenn, Daniel Slamanig, and Christoph Striecks. 2022. Logarithmic-Size (Linkable) Threshold Ring Signatures in the Plain Model. In *PKC 2022: 25th International Conference on Theory and Practice of Public Key Cryptography, Part II (Lecture Notes in Computer Science, Vol. 13178)*, Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe (Eds.). Springer, Cham, Switzerland, Virtual Event, 437–467. https://doi.org/10.1007/978-3-030-97131-1_15

[29] Abida Haque and Alessandra Scafuro. 2020. Threshold Ring Signatures: New Definitions and Post-quantum Security. In *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part II (Lecture Notes in Computer Science, Vol. 12111)*, Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas (Eds.). Springer, Cham, Switzerland, Edinburgh, UK, 423–452. https://doi.org/10.1007/978-3-030-45388-6_15

[30] Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. 2018. Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures. In *ACM CCS 2018: 25th Conference on Computer and Communications Security*, David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang (Eds.). ACM Press, Toronto, ON, Canada, 525–537. https://doi.org/10.1145/3243734.3243805

[31] Ahmed Kosba, Zhichao Zhao, Andrew Miller, Yi Qian, Hubert Chan, Charalampos Papamanthou, Rafael Pass, abhi shelat, and Elaine Shi. 2015. C∅C∅: A Framework for Building Composable Zero-Knowledge Proofs. Cryptology ePrint Archive, Report 2015/1093. https://eprint.iacr.org/2015/1093

[32] Russell W. F. Lai, Viktoria Ronge, Tim Ruffing, Dominique Schröder, Sri Aravinda Krishnan Thyagarajan, and Jiafan Wang. 2019. Omniring: Scaling Private Payments Without Trusted Setup. In *ACM CCS 2019: 26th Conference on Computer and Communications Security*, Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz (Eds.). ACM Press, London, UK, 31–48. https://doi.org/10.1145/3319535.3345655

[33] Xingye Lu, Man Ho Au, and Zhenfei Zhang. 2019. Raptor: A Practical Lattice-Based (Linkable) Ring Signature. In *ACNS 19: 17th International Conference on Applied Cryptography and Network Security (Lecture Notes in Computer Science, Vol. 11464)*, Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung (Eds.). Springer, Cham, Switzerland, Bogota, Colombia, 110–130. https://doi.org/10.1007/978-3-030-21568-2_6

[34] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. 2021. SMILE: Set Membership from Ideal Lattices with Applications to Ring Signatures and Confidential Transactions. In *Advances in Cryptology – CRYPTO 2021, Part II (Lecture Notes in Computer Science, Vol. 12826)*, Tal Malkin and Chris Peikert (Eds.). Springer, Cham, Switzerland, Virtual Event, 611–640. https://doi.org/10.1007/978-3-030-84245-1_21

[35] Silvio Micali, Leonid Reyzin, Georgios Vlachos, Riad S. Wahby, and Nickolai Zeldovich. 2021. Compact Certificates of Collective Knowledge. In *2021 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, San Francisco, CA, USA, 626–641. https://doi.org/10.1109/SP40001.2021.00096

[36] Alexander Munch-Hansen, Claudio Orlandi, and Sophia Yakoubov. 2021. Stronger Notions and a More Efficient Construction of Threshold Ring Signatures. In *Progress in Cryptology - LATINCRYPT 2021: 7th International Conference on Cryptology and Information Security in Latin America (Lecture Notes in Computer Science, Vol. 12912)*, Patrick Longa and Carla Ràfols (Eds.). Springer, Cham, Switzerland, Bogotá, Colombia, 363–381. https://doi.org/10.1007/978-3-030-88238-9_18

[37] Shen Noether. 2015. Ring Signature Confidential Transactions for Monero. Cryptology ePrint Archive, Report 2015/1098. https://eprint.iacr.org/2015/1098

[38] Shen Noether. 2015. Ring Signature Confidential Transactions for Monero. Cryptology ePrint Archive, Paper 2015/1098. https://eprint.iacr.org/2015/1098

[39] Ronald L. Rivest, Adi Shamir, and Yael Tauman. 2001. How to Leak a Secret. In *Advances in Cryptology – ASIACRYPT 2001 (Lecture Notes in Computer Science, Vol. 2248)*, Colin Boyd (Ed.). Springer Berlin Heidelberg, Germany, Gold Coast, Australia, 552–565. https://doi.org/10.1007/3-540-45682-1_32

[40] Michael Sipser. 1983. Borel Sets and Circuit Complexity. In *15th Annual ACM Symposium on Theory of Computing*. ACM Press, Boston, MA, USA, 61–69. https://doi.org/10.1145/800061.808733

[41] Kang Yang, Pratik Sarkar, Chenkai Weng, and Xiao Wang. 2021. QuickSilver: Efficient and Affordable Zero-Knowledge Proofs for Circuits and Polynomials over Any Field. In *ACM CCS 2021: 28th Conference on Computer and Communications Security*, Giovanni Vigna and Elaine Shi (Eds.). ACM Press, Virtual Event, Republic of Korea, 2986–3001. https://doi.org/10.1145/3460120.3484556

[42] Tsz Hon Yuen, Muhammed F. Esgin, Joseph K. Liu, Man Ho Au, and Zhimin Ding. 2021. DualRing: Generic Construction of Ring Signatures with Efficient Instantiations. In *Advances in Cryptology – CRYPTO 2021, Part I (Lecture Notes in Computer Science, Vol. 12825)*, Tal Malkin and Chris Peikert (Eds.). Springer, Cham, Switzerland, Virtual Event, 251–281. https://doi.org/10.1007/978-3-030-84242-0_10

[43] Tsz Hon Yuen, Shifeng Sun, Joseph K. Liu, Man Ho Au, Muhammed F. Esgin, Qingzhao Zhang, and Dawu Gu. 2020. RingCT 3.0 for Blockchain Confidential Transaction: Shorter Size and Stronger Security. In *FC 2020: 24th International Conference on Financial Cryptography and Data Security (Lecture Notes in Computer Science, Vol. 12059)*, Joseph Bonneau and Nadia Heninger (Eds.). Springer, Cham, Switzerland, Kota Kinabalu, Malaysia, 464–483. https://doi.org/10.1007/978-3-030-51280-4_25

A OMITTED DEFINITIONS

A.1 Non-interactive Zero-knowledge Arguments-of-Knowledge

Let us recall the syntax and definitions of non-interactive zero-knowledge proofs, with a focus on arguments-of-knowledge. We employ the definitions of [31] although transferred to the random oracle model, in the spirit of [21].

A NIZKAoK includes the following PPT algorithms,

$\mathsf{Setup}^{\mathsf{H}}(\mathscr{R}_{\mathcal{L}}, 1^\lambda) \rightarrow (\mathsf{crs})$**:** Given an NP relation $\mathscr{R}_{\mathcal{L}}$ for language $\mathcal{L}$ and the security parameter 1^λ produces a common reference string crs.

			Ring size						
Ring Signature	Linkable	Assumption	2^3	2^6	2^8	2^{10}	2^{12}	2^{20}	Security
This work	✓	AES128	9.84	9.91	10.05	10.09	10.18	10.53	NIST I
This work	✓	AES196	27.15	27.26	27.58	27.64	27.77	28.54	NIST III
This work	✓	AES256	51.12	51.27	51.80	51.98	52.15	53.56	NIST V
This work		AES128	4.78	4.86	4.99	5.03	5.12	5.47	NIST I
This work		AES192	12.43	12.54	12.86	12.92	13.05	13.82	NIST III
This work		AES256	22.48	22.63	23.16	23.34	23.51	24.92	NIST V
This work		AES128-EM	4.34	4.42	4.55	4.59	4.68	5.03	NIST I
This work		AES192-EM	10.51	10.62	10.94	11.00	11.13	11.90	NIST III
This work		AES256-EM	21.33	21.49	22.02	22.19	22.37	23.79	NIST V
TCitH [22]		AES128	7.87	7.90	7.94	8.02	8.18	9.39	NIST I
		AES128-EM	6.81	6.84	6.88	6.96	7.12	8.27	NIST I
		MQ over $\mathbb{F}_{256}$	4.30	4.33	4.37	4.45	4.60	5.62	NIST I
		SD over $\mathbb{F}_{256}$	7.37	7.51	7.96	8.24	8.40	10.09	NIST I
KKW [30]		LowMC	-	250	-	-	456	-	NIST V
GGHK [26]		LowMC	-	-	-	56	-	-	NIST V
Falafl [10]	✓	MSIS/MLWE	30	32	-	-	35	39	NIST I
Raptor [33]	✓	MSIS/MLWE	10	81	333	1290	5161	-	100 bit
MatRiCT [19]		MSIS/MLWE	19	31	-	-	148	-	NIST II
SMILE [34]		MSIS/MLWE	16	-	-	18	19	22	128 bit
DualRing [42]		MSIS/MLWE	4.63	6.02	10.78	30.13	106.57	-	128 bit
Gandalf [24]		MSIS/MLWE	4.87	38.81	155.16	621	-	-	128 bit
Calamari [10]	✓	CSIDH	5	8	-	-	14	23	128 bit
LESS [1]		Code Equivalence	11	14	-	-	20	-	128 bit
MR-DSS [9]		MinRank	27	36	64	145	422	-	NIST I

Table 2: Expanded overview: (Linkable) ring signature sizes in kilobytes.

$\mathsf{Prove}^{\mathsf{H}}(\mathsf{crs}, \phi, \mathsf{w}) \to \pi$ Given the common reference string and statement, witness pair in the relation $(\phi, \mathsf{w}) \in \mathscr{R}$, produces a non-interactive proof π.

$\mathsf{Verify}^{\mathsf{H}}(\mathsf{crs}, \phi, \pi) \to \{0, 1\}$ Given the common reference string and statement, and a proof for the statement π, outputs either 1 (accept) or 0 (reject).

We require that verification always accepts an honestly produced proof for a statement in the language.

Definition A.1 (Perfect Completeness [31, Modified]). For all $(\phi, \mathsf{w}) \in \mathscr{R}_{\mathcal{L}}$ it holds that

$$\Pr\left[\ \mathsf{crs} \leftarrow \mathsf{Setup}^{\mathsf{H}}(\mathscr{R}_{\mathcal{L}}, 1^{\lambda}) \quad : \ \mathsf{Verify}^{\mathsf{H}}(\mathsf{crs}, \phi, \pi) = 1\right] = 1.$$

We further require zero-knowledge which is the property that proofs do not reveal anything about the witness used to produce them. This is captured following the simulation paradigm, where we require it be possible to produce simulate proofs without access to the witness which are indistinguishable from real proofs. At the same time it should not be possible for the adversary to produce proofs without knowing the witness. We give the simulator additional power to achieve this by allowing an alternative setup:

$\mathsf{TdSetup}^{\mathsf{H}}(\mathscr{R}_{\mathcal{L}}, 1^{\lambda}) \to (\mathsf{crs}, \tau, \tau_{ext})$: Given an NP relation $\mathscr{R}$ and the security parameter 1^{λ} produces a common reference string crs along with a simulation trapdoor τ and extraction trapdoor τ_{ext}.

As we are in the random oracle model we also give the simulator the power to simulate the random oracle. The simulator for a scheme $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ consists of a random oracle simulator $\mathcal{S}_1$ and a proof oracle simulator $\mathcal{S}_2$, we let the simulators hold a shared mutable state st and give them access to the random oracle H, but leave this implicit in the notation. We quantify the simulator as part of the scheme as it is common to the following two definitions.

Definition A.2 (Computational Zero-knowledge [31, Modified]). For all PPT adversaries $\mathcal{A}$ we require,

$$\Pr\left[\ \mathsf{crs} \leftarrow \mathsf{Setup}^{\mathsf{H}}(\mathscr{R}_{\mathcal{L}}, 1^{\lambda}) \quad : \ \mathcal{A}^{\mathsf{H}, \mathcal{P}(\mathsf{crs}, \cdot, \cdot)}(\mathsf{crs}) = 1\right]$$
$$\approx \Pr\left[\ (\mathsf{crs}, \tau, \tau_{ext}) \leftarrow \mathsf{TdSetup}^{\mathsf{H}}(\mathscr{R}, 1^{\lambda}) \quad : \ \mathcal{A}^{\mathcal{S}_1(\mathsf{crs}, \tau, \cdot), \mathcal{S}_2'(\mathsf{crs}, \tau, \cdot, \cdot)}(\mathsf{crs}) = 1\right],$$

where $\mathcal{P}(\mathsf{crs}, \phi, \mathsf{w})$ (resp. $\mathcal{S}_2'(\mathsf{crs}, \tau, \phi, \mathsf{w})$) first checks $(\phi, \mathsf{w}) \in \mathscr{R}_{\mathcal{L}}$, aborting the experiment if this does not hold, and outputs $\pi \leftarrow \mathsf{Prove}^{\mathsf{H}}(\mathsf{crs}, \phi, \mathsf{w})$ (resp. $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, \phi)$).

As we are constructing signature schemes it suffices for it to be hard for the adversary to produce a proof for a statement ϕ without a witness where it has not seen any simulated proofs for ϕ.

Definition A.3 ((Black-box) Weak Simulation-Extractability [31, Modified]). There exists an extractor $\mathcal{E}$ such that for all PPT adversaries $\mathcal{A}$,

$$\Pr\left[\begin{array}{lcl} (\mathsf{crs}, \tau, \tau_{ext}) \leftarrow \mathsf{TdSetup}^{\mathsf{H}}(\mathscr{R}, 1^{\lambda}) & & (x^*, \_) \notin \mathcal{T} \wedge \\ (x^*, \pi^*) \leftarrow \mathcal{A}^{\mathcal{S}_1(\mathsf{crs}, \tau, \cdot), \mathcal{S}_2'(\mathsf{crs}, \tau, \cdot, \cdot)}(\mathsf{crs}) & : & (x^*, w^*) \notin \mathcal{R}_{\mathcal{L}} \wedge \\ w^* \leftarrow \mathcal{E}^{\mathsf{H}}(\tau_{ext}, x^*, \pi^*) & & \mathsf{Verify}^{\mathcal{S}_1}(x^*, \pi^*) = 1 \end{array}\right]$$
$$\leq \mathsf{negl}(\lambda),$$

where $\mathcal{T}$ is the set of statement witness pairs (ϕ, w) where $\mathcal{S}_2'(\mathsf{crs}, \tau, \phi, \mathsf{w})$ has been queried. As for zero-knowledge, $\mathcal{S}_2'$ simply checks $(\phi, \mathsf{w}) \in \mathscr{R}_{\mathcal{L}}$ returning $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, \phi)$ if this is the case, and aborting otherwise.

A.2 Omitted definitons from Section 6

We recall an definition of an ALBA scheme from [14, Definition 1].

Definition A.4. The triple (Prove, Verify, Read) is a $(\lambda_{sound}, \lambda_{comp}, n_p, n_f)-$ALBA scheme if and only if:

- Prove is a probabilistic polynomial-time (PPT) program with access to the random oracle H and the binary oracle W.
- Verify is a program with access to the random oracle H and the binary oracle W.
- Read is a program that reads an ALBA proof and returns the signatures.
- Extract is a program with access to the random oracle H and the binary oracle W. The program is also given access to the adversary $\mathcal{A}$, but is restricted to only run it once observing the queries $\mathcal{A}$ makes to the real oracles H, W.

Moreover, it must satisfy the following properties:

- **Completeness:** For all oracles W and all S_p such that $W(S_p) \geq n_p$,

  $\Pr[\mathsf{Verify}^{H,W}(\mathsf{Prove}^{H,W}(S_p)) = 1] \geq 1 - 2^{-\lambda_{comp}}$.
- **Proof of knowledge:** Let E be the random variable over the random coins of $\mathcal{A}$ and Extract, which is 1 exactly when $W(S_f) > n_f$ for $S_f \leftarrow \mathsf{Extract}^{H,W,\mathcal{A}}()$. Knowledge soundness requires for all oracles W,

  $\Pr[E = 1] \geq \Pr[\mathsf{Verify}^{H,W}(\mathcal{A}^{H,W}())] - 2^{-\lambda_{sound}}$.

  Note extraction is straight-line to the constrains on Extract.

B OMITTED PROOFS

B.1 Omitted proofs from Section 5.1

Proof of Lemma 5.3. Consider an adversary having made i previous queries $\{(k_j, y_{j,0}, y_{j,1}, \ldots, y_{j,n})\}_{j\in[i]}$ such that $y_{j,\ell} = \pi_{k_j}(0\ldots0||\mathsf{bits}(\ell))$ for $\ell \in [n]$. We consider k_j to have been queried if the adversary has queries π_{k_j} or $\pi^{-1}_{k_j}$ on any value. The adversary finds a collision if it queries a fresh key $k \neq k_j$ for $j \in [i]$ where π_k matches π_{k_j} for inputs 0 to n. For each of the previous queries we may bound the probability,

$$\begin{aligned}
&\Pr\left[y_{j,1} = \pi_k(0\ldots0||\mathsf{bits}(1)), \ldots, y_{j,n} = \pi_k(0\ldots0||\mathsf{bits}(n))\right]\\
&\quad= \Pr\left[y_{j,1} = \pi_k(0\ldots0||\mathsf{bits}(1))\right]\\
&\qquad\cdot \Pr\left[y_{j,2} = \pi_k(0\ldots0||\mathsf{bits}(2)) \mid y_{j,1} = \pi_k(0\ldots0||\mathsf{bits}(1))\right]\cdot\ldots\\
&\qquad\cdot \Pr\left[y_{j,n} = \pi_k(0\ldots0||\mathsf{bits}(n)) \mid \ldots, y_{j,n-1} = \pi_k(0\ldots0||\mathsf{bits}(n-1))\right]\\
&\quad\leq \left(\frac{1}{|\mathcal{X}|}\right)\left(\frac{1}{|\mathcal{X}|-1}\right)\cdots\left(\frac{1}{|\mathcal{X}|-n+2}\right) = 1\Bigg/\left(\prod_{r=0}^{n-2}|\mathcal{X}|-r\right) = O\left(|\mathcal{X}|^{-n}\right).
\end{aligned}$$

where the last equality holds for constant n. For each new query we apply a union bound over the collision probability with each previous query. Bounding across all queries, the combined collision probability is then at most $\sum_{i=2}^{q} i\cdot O\left(|\mathcal{X}|^{-n}\right) = O(q^2|\mathcal{X}|^{-n})$. □

B.2 Omitted proofs from Section 5.3

Lemma B.1. *The threshold ring signature scheme defined in Figure 8 is correct (Definition 5.6) if* NIZKAoK *has completeness and* P, T *are pseudorandom.*

Proof of Lemma B.1. In a sequence of hybrids we will show that the probability distribution of successful verification is indistinguishable from verification always succeeding. Fix arbitrary polynomial size message m, signer set and ring $\mathcal{S} \subseteq \mathcal{R}$, with $|\mathcal{S}| = t$.

$\mathcal{H}_0$: Consider the sequence,

$$\begin{aligned}
&\mathsf{pp} \leftarrow \mathsf{TRS.Setup}(1^\lambda);\\
&\{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TRS.KeyGen}()\}_{i\in\mathcal{R}};\\
&\{\sigma_i \leftarrow \mathsf{TRS.Sign}(m, \{\mathsf{pk}_j\}_{j\in\mathcal{R}}, \mathsf{sk}_i)\}_{i\in\mathcal{S}};\\
&\sigma \leftarrow \mathsf{TRS.Combine}(\{\sigma_i\}_{i\in\mathcal{S}}, t)
\end{aligned}$$

let V be the random variable taking the value of

$$\mathsf{TRS.Verify}(m, \sigma, \{\mathsf{pk}_i\}_{i\in\mathcal{R}}, t)$$

with the probability being taken over the random coins of the algorithms.

$\mathcal{H}_1$: As the previous game, but let $V = 1$ if verification rejects due to $\mathsf{NIZKAoK.Verify}((\mathsf{pk}, m, \tau), \pi)$ rejecting.

$\mathcal{H}_2$: (Sequence of t hybrids) In a sequence of hybrids, with one for each $i \in \mathcal{S}$, each is as the previous, but during signing for $i \in \mathcal{S}$ compute the tag as a random element of the output space $\tau \leftarrow \mathcal{Y}$ rather than $F(\mathsf{sk}_i, (\mathsf{PK}, m))$.

$\mathcal{H}_3$: Let V be a random variable which is always 1.

If $\mathcal{H}_0 \approx \mathcal{H}_3$ we may conclude that the scheme is correct except with negligible error. We prove indistinguishability of the sequence.

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** By correctness of NIZKAoK verification of an honest proof can at most fail with some negligible probability $\epsilon^{\mathsf{NIZKAoK}}_{\mathsf{COR}}$.

$\mathcal{H}_1 \approx \mathcal{H}_2$**:** For each $i \in \mathcal{S}$ the tag an adversary distinguishing the case where τ is random in $\mathcal{Y}$ and $T_{\mathsf{sk}_i}(h)$ may be reduced to an adversary winning the PRF game for T with the same advantage.

$\mathcal{H}_2 \approx \mathcal{H}_3$**:** The only case where $\mathcal{H}_2$ and $\mathcal{H}_3$ differ is if there exist two tags τ_i, τ_j for $i \neq j \in \mathcal{S}$ which are identical. As all tags are uniformly random any single pair collides with probability $|\mathcal{Y}|^{-1}$. A union bound across all pairs then bounds the probability that there is any collision.

The bound follows,

$$\mathbf{Adv}^{\mathsf{Correct}}_{\mathsf{TRS},\mathcal{A}} \leq \epsilon^{\mathsf{NIZKAoK}}_{\mathsf{COR}} + t\cdot\epsilon^{P,T}_{\mathsf{PRF}} + \frac{t(t-1)}{2}|\mathcal{Y}|^{-1}.$$

□

We may now proceed to prove anonymity.

Lemma B.2. *The threshold ring signature scheme defined in Figure 8 is anonymous (Definition 5.8) if* NIZKAoK *is zero-knowledge and* P, T *are pseudorandom.*

Proof of Lemma B.2. Let $u = \mathsf{poly}(\lambda)$ be a bound on $|\mathcal{U}|$. Throughout this proof we will consider the challenge query $(m, \mathcal{R} \subset \mathcal{U}, i_0, i_1)$ to be one of the signing queries. Consider the sequence of hybrids,

$\mathcal{H}_0$: Run $\mathsf{Game}^{\mathsf{Anon},b}_{\mathsf{TRS},\mathcal{A}}(1^\lambda)$ as in Figure 7, and sample two distinct indices uniformly at random $i^* \neq j^* \leftarrow [u]$.

$\mathcal{H}_1$: As previous but rather than providing real proofs, replace all invocations of NIZKAoK.Prove with simulated proofs.

This requires producing the CRS through TdSetup and replacing the random oracle with the simulated random oracle.

$\mathcal{H}_2$: As previous, but if the challenge query $(m, \mathcal{R} \subset \mathcal{U}, i_0, i_1)$ doesn't satisfy $\{i^*, j^*\} = \{i_0, i_1\}$ sample a random coin $b \leftarrow \{0,1\}$ and output this as the result of the game.

$\mathcal{H}_3$: As previous, but replace y in pk_{i^*} with a uniform random element of $\mathcal{Y}$. Additionally, for every signing query $(\mathsf{Sign}, m, \mathcal{R}, i^*)$ where i^* has not yet signed $m, \mathcal{R}$ (i.e. $i^* \notin Q^{m,\mathcal{R}}_{\mathsf{sig}}$) rather than computing $F_{\mathsf{sk}_{i^*}}(\mathsf{H}(m, \{(i, \mathsf{pk}_i)\}_{i \in \mathcal{R}}))$ sample $\tau \leftarrow \mathcal{Y}$, uniform independently. If $i^* \in Q^{m,\mathcal{R}}_{\mathsf{sig}}$, retrieve the τ which was previously sampled.

$\mathcal{H}_4$: As previous, making the same modifications, but for j^*.

Starting from $\mathcal{H}_0$ for $b = 0$ or $b = 1$ we may make the same hybrid jumps arriving at $\mathcal{H}_4$ where the two cases are identical.

$\mathcal{H}_0 \approx \mathcal{H}_1$: Indistinguishability follows directly from zero-knowledge of NIZKAoK. Any adversary distinguishing the two hybrids may be reduced to an adversary breaking zero-knowledge of NIZKAoK with the same advantage.

$$\epsilon^{\mathsf{NIZKAoK}}_{\mathsf{ZK}} \geq |\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_1] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_0]|\,.$$

$\mathcal{H}_1 \approx \mathcal{H}_2$: When the adversary wins there must be at least two honest parties remaining in the system, the probability that these have indices i^*, j^* is $2/u(u-1)$,

$$\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_2] = \left(1 - \frac{2}{u(u-1)}\right) \cdot \frac{1}{2} + \frac{2}{u(u-1)} \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_1].$$

$\mathcal{H}_2 \approx \mathcal{H}_3$: From $\mathcal{H}_2$ we know i^* is never corrupted, so outputs of $T(\mathsf{sk}_{i^*}, \cdot)$ may be replaced by values produced by a PRF challenger. Indistinguishability follows by pseudorandomness of P, T,

$$\epsilon^{F}_{\mathsf{PRF}} \geq |\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_3] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_2]|\,.$$

$\mathcal{H}_3 \approx \mathcal{H}_4$: By the same argument as for $\mathcal{H}_3$,

$$\epsilon^{F}_{\mathsf{PRF}} \geq |\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_4] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_3]|\,.$$

The signatures produced for a challenge query $(m, \mathcal{R} \subset \mathcal{U}, i_0, i_1)$, $i^* \in \{i_0, i_1\}$ are now entirely independent of b, as the tag is sampled uniformly (and freshly) due to $i^*, j^* \notin Q^{m,\mathcal{R}}_{\mathsf{sig}}$ and NIZKAoK proofs are simulated.

Let $\mathcal{H}_0{}^{(b)}$ be the hybrid starting at $\mathsf{Game}^{\mathsf{Anon},b}_{\mathsf{TRS}\mathcal{A}}(1^\lambda)$, and $\mathcal{H}_i{}^{(b)}$ be the hybrid after i jumps in the sequence above. We know

$$|\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_4{}^{(0)}] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_4{}^{(1)}]| = 0,$$

therefore,

$$|\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_2{}^{(0)}] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_2{}^{(1)}]| \leq 4 \cdot \epsilon^{F}_{\mathsf{PRF}}.$$

Also,

$$|\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_0{}^{(b)}] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_1{}^{(b)}]| \leq \epsilon^{\mathsf{NIZKAoK}}_{\mathsf{ZK}}$$

Following,

$$|\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_1{}^{(0)}] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_1{}^{(1)}]|$$
$$= \frac{u(u-1)}{2} |\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_2{}^{(0)}] - \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_2{}^{(1)}]|$$

we derive the following advantage bound,

$$\mathsf{Adv}^{\mathsf{Anon}}_{\mathsf{TRS},\mathcal{A}} \leq \epsilon^{\mathsf{NIZKAoK}}_{\mathsf{ZK}} + 2u(u-1) \cdot \epsilon^{F}_{\mathsf{PRF}}.$$

□

Finally, we show unforgeability.

Lemma B.3. *The threshold ring signature scheme defined in Figure 8 is unforgeability (Definition 5.7) if* NIZKAoK *is zero-knowledge and weak-simulation-extractable and* P, T *are collision resistant, pseudorandom and one-way.*

Proof of Lemma B.3. Let $u = \mathsf{poly}(\lambda)$ be a bound on $|\mathcal{U}|$, and $t = \mathsf{poly}(\lambda)$ be a bound on $|\mathcal{R}|$.

$\mathcal{H}_0$: Run $\mathsf{Game}^{\mathsf{Unforge}}_{\mathsf{TRS}\mathcal{A}}(1^\lambda)$ as in Figure 6.

$\mathcal{H}_1$: As previous, but select a random index $i^* \leftarrow [n]$. At the end of the game, let $\mathcal{A}$ *lose* if $i^* \notin \mathcal{U}$ or $i^* \in Q_{\mathsf{corr}}$.

$\mathcal{H}_2$: As previous but rather than providing real proofs, replace all invocations of NIZKAoK.Prove with simulated proofs, replace the random oracle with the simulated oracle associated with proof simulation.

$\mathcal{H}_3$: As previous, but for a winning forgery $(\sigma, m, \mathcal{R}, t)$ with $\sigma = (\pi_1, \tau_1)|| \ldots ||(\pi_{t^*}, \tau_{t^*})$, run the extractor to obtain a witness from each statement (PK, m, τ_i) and proof π_i for $i \in [t^*]$. Let $\mathcal{A}$ *lose* if extraction fails for any statement $\phi = (\mathsf{PK}, m, \tau_i)$ where no proof has been simulated for ϕ during the signing queries.

$\mathcal{H}_4$: Let $\mathcal{E} \subset [t^*]$ be the indices where extraction succeeds (resulting in k_i for $i \in \mathcal{E}$), let $\mathcal{A}$ *lose* if there exist any $i \neq j \in \mathcal{E}$, such that k_i, k_j correspond to the same public key, i.e. for $\mathsf{pk} = (x, y) \in \mathcal{R}$ it holds $y = P_{k_i}(x) = P_{k_j}(x)$.

$\mathcal{H}_5$: Let $\mathcal{A}$ lose if the secret key for i^* is not among the extracted keys.

$\mathcal{H}_6$: For signing queries $(\mathsf{Sign}, m, \mathcal{R}, i^*)$ for i^* change the statement proofs are simulated for to $\phi = (\mathsf{PK}, m, \tau)$ where the tag is now sampled uniformly at random $\tau \leftarrow \mathcal{Y}$. Store τ, so that it may be retrieved and used if $\tau_{\mathsf{sk}_{i^*}}$ is queried on the same input again.

$\mathcal{H}_7$: Let $\mathcal{A}$ always *lose*.

As the adversary always loses in $\mathcal{H}_7$, we may use the distinguishing advantages of the sequence to bound the advantage in $\mathcal{H}_0$.

$\mathcal{H}_0$ and $\mathcal{H}_1$: The index i^* is chosen from $[u]$ uniform independently, giving probabilities $\Pr[i \in \mathcal{U}] = |\mathcal{U}|/u$ and

$$\Pr\left[i^* \notin Q_{\mathsf{corr}} \;\middle|\; i^* \in \mathcal{U}\right] = |\mathcal{U} \setminus Q_{\mathsf{corr}}|/|\mathcal{U}|.$$

Putting these together, we obtain

$$\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_1] \geq \frac{|\mathcal{U} \setminus Q_{\mathsf{corr}}|}{u} \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_0].$$

$\mathcal{H}_1 \approx \mathcal{H}_2$: Indistinguishability of these hybrids follows from zero-knowledge of NIZKAoK. For advantage bound $\epsilon^{\mathsf{NIZKAoK}}_{\mathsf{ZK}}$ in distinguishing simulated and real proofs,

$$\Pr[\mathcal{A} \textit{ wins } \mathcal{H}_2] + \epsilon^{\mathsf{NIZKAoK}}_{\mathsf{ZK}} \geq \Pr[\mathcal{A} \textit{ wins } \mathcal{H}_1].$$

As otherwise $\mathcal{A}$ would allow distinguishing with advantage greater than $\epsilon^{\mathsf{NIZKAoK}}_{\mathsf{ZK}}$ in the zero-knowledge game.

$\mathcal{H}_2 \approx \mathcal{H}_3$: (Sequence of t hybrids) For a forgery $\sigma = (\pi_1, \tau_1)|| \dots ||(\pi_{t^*}, \tau_{t^*})$. Let hybrid j in this sequence be as the previous, except the hybrid runs the extractor for statement (PK, m, τ_j) and proof π_j. The hybrid causes $\mathcal{A}$ to lose if $j \le t^*$ and extraction fails and no proof has previously been simulated for (PK, m, τ_j). Weak simulation extractability implies that it must be possible to extract from any proof for a statement where no proof has been simulated, except with negligible probability. Let $\mathcal{H}_2 = \mathcal{H}_3{}^{(0)}$ and $\mathcal{H}_3 = \mathcal{H}_3{}^{(t)}$, then for our sequence of hybrids $j \in [t]$

$$\Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_3{}^{(j)}] + \epsilon^{\mathsf{NIZKAoK}}_{\text{WEAK-SIM-EXT}} \ge \Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_3{}^{(j-1)}],$$

implying $\Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_3] + t \cdot \epsilon^{\mathsf{NIZKAoK}}_{\text{WEAK-SIM-EXT}} \ge \Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_2]$.

$\mathcal{H}_3 \approx \mathcal{H}_4$: These two cases are distinguishable exactly when the extractor prodcues output which would win the key collision game for P, i.e. x, k, k' subject to $P_k(x) = P_{k'}(x)$. Thus,

$$\Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_4] + \epsilon^{P}_{\text{COL}} \ge \Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_3].$$

$\mathcal{H}_4$ and $\mathcal{H}_5$: We will argue that a forgery must result in extracting the key of at least one honest party, it will then follow directly that this includes party i^* with probability at least $1/|\mathcal{U} \setminus Q_{\mathsf{corr}}|$. Let,

$$\mathcal{S} = \{j \in \mathcal{U} \mid \exists i \in \mathcal{E}, \exists \mathsf{pk} = (x, y) \in \mathsf{PK} : y = P_{k_i}(x)\}.$$

At this point we know that extraction can at most fail for $|Q^{m,\mathcal{R}}_{\mathsf{sig}} \setminus \mathcal{S}|$ of the indices $i \in [t]$, as extraction may only fail for tags where a proof has been simulated. As each extracted secret key corresponds to a distinct public key, it follows $|\mathcal{S}| = |\mathcal{E}| \ge t^* - |Q^{m,\mathcal{R}}_{\mathsf{sig}} \setminus \mathcal{S}|$.

For notational convenience, let $Q^{\mathcal{R}}_{\mathsf{corr}} = Q_{\mathsf{corr}} \cap \mathcal{R}$. We may show that at least one of the extracted keys must be from an honest party

$$\begin{aligned}|\mathcal{S} \setminus (Q^{m,\mathcal{R}}_{\mathsf{sig}} \cup Q^{\mathcal{R}}_{\mathsf{corr}})| &= |\mathcal{S}| - |\mathcal{S} \cap (Q^{m,\mathcal{R}}_{\mathsf{sig}} \cup Q^{\mathcal{R}}_{\mathsf{corr}})| \\ &= |\mathcal{S}| - |(Q^{m,\mathcal{R}}_{\mathsf{sig}} \setminus (Q^{m,\mathcal{R}}_{\mathsf{sig}} \setminus \mathcal{S})) \cup (\mathcal{S} \cap Q^{\mathcal{R}}_{\mathsf{corr}} \setminus Q^{m,\mathcal{R}}_{\mathsf{sig}})| \\ &= |\mathcal{S}| - |Q^{m,\mathcal{R}}_{\mathsf{sig}}| + |Q^{m,\mathcal{R}}_{\mathsf{sig}} \setminus \mathcal{S}| - |\mathcal{S} \cap Q^{\mathcal{R}}_{\mathsf{corr}} \setminus Q^{m,\mathcal{R}}_{\mathsf{sig}}|\end{aligned}$$

as $|\mathcal{S}| \ge t^* - |Q^{m,\mathcal{R}}_{\mathsf{sig}} \setminus \mathcal{S}|$ it follows

$$\begin{aligned}&\ge t^* - |Q^{m,\mathcal{R}}_{\mathsf{sig}}| - |\mathcal{S} \cap Q^{\mathcal{R}}_{\mathsf{corr}} \setminus Q^{m,\mathcal{R}}_{\mathsf{sig}}| \\ &\ge t^* - |\mathcal{R} \cap (Q^{m,\mathcal{R}}_{\mathsf{sig}} \cup Q^{\mathcal{R}}_{\mathsf{corr}})| \ge 1\end{aligned}$$

where the first inequality follows by $\mathcal{S} \subset \mathcal{R}$ and the second inequality follows as $|\mathcal{R} \cap (Q^{m,\mathcal{R}}_{\mathsf{sig}} \cup Q^{\mathcal{R}}_{\mathsf{corr}})| < t^*$ when $\mathcal{A}$ *wins*. We may therefore conclude,

$$\Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_5] \ge \frac{1}{|\mathcal{U} \setminus Q_{\mathsf{corr}}|} \Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_4].$$

$\mathcal{H}_5 \approx \mathcal{H}_6$: $\mathcal{A}$ may be reduced to an adversary winning the PRF game for P, T with advantage equal to the difference in winning probabilities between $\mathcal{H}_6$ and $\mathcal{H}_5$ by using the oracle in the PRF game to produce the tags for party i^*.

$$\Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_6] + \epsilon^{P,T}_{\text{PRF}} \ge \Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_5].$$

$\mathcal{H}_6 \approx \mathcal{H}_7$: At this we no longer need access to the secret key of party i^* as their tags are sampled randomly for signing queries. Therefore, we may replace the public key of i with a challenge from the one-way game of P, T. An adversary winning $\mathcal{H}_6$ may be reduced to win the OWF game with the same advantage.

$$\Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_7] + \epsilon^{P,T}_{\text{OWF}} \ge \Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_6].$$

By definition $\Pr[\mathcal{A}\ \mathit{wins}\ \mathcal{H}_7] = 0$, let $h = |\mathcal{U} \setminus Q_{\mathsf{corr}}|$, collecting our bounds we obtain,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{Unforge}}_{\mathsf{TRS},\mathcal{A}} &\le \frac{u}{h} \cdot \left(\epsilon^{\mathsf{NIZKAoK}}_{\text{ZK}} + t \cdot \epsilon^{\mathsf{NIZKAoK}}_{\text{WEAK-SIM-EXT}} + \epsilon^{P}_{\text{COLL}} + h \cdot (\epsilon^{P,T}_{\text{PRF}} + \epsilon^{P,T}_{\text{OWF}})\right) \\ &\le u \cdot \left(\epsilon^{\mathsf{NIZKAoK}}_{\text{ZK}} + t \cdot \epsilon^{\mathsf{NIZKAoK}}_{\text{WEAK-SIM-EXT}} + \epsilon^{P}_{\text{COLL}} + \epsilon^{P,T}_{\text{PRF}} + \epsilon^{P,T}_{\text{OWF}}\right).\end{aligned}$$

□

B.3 Omitted proofs from Section 6.2

Proof of Theorem 6.2. *Completeness.* Follows from the completeness of ALBA and Read only outputting elements of weight 1.

Knowledge-soundness. Consider the extractor defined in Figure 12.[6] Following the same reasoning as [14, Theorem 21] we may argue

```
Extract^{H,W}(𝒜)
1 :  $\Sigma \leftarrow \varnothing$
2 :  do
3 :      $H \xleftarrow{\$} \mathcal{H}$
4 :      $\pi \leftarrow \mathcal{A}^H()$
5 :      if $\mathsf{ELBA.Verify}^{H,W}(\pi) \neq 1$ then continue
6 :      $\Sigma \leftarrow \Sigma \cup (\mathsf{ELBA.Read}(\pi) \cap W)$
7 :  while $|\Sigma| \le n_f$
8 :  return $\Sigma$
```

Figure 12: The extractor for ELBA.

that each run of the loop will result in at least one new signature with probability $\varepsilon = \mathsf{Adv}^{\mathsf{sound}}_{\mathsf{ELBA},\mathcal{A}} - 2^{-\lambda_{sound}}$, where $\mathsf{Adv}^{\mathsf{sound}}_{\mathsf{ELBA},\mathcal{A}} = \Pr[H \xleftarrow{\$} \mathcal{H}, \pi \leftarrow \mathcal{A}^{H,W}; V^{H,W}(\pi) = 1]$. We recall their analysis for the sake of completeness. Fix a set $\Sigma \subset W$ where $|\Sigma| \le n_f$, let $\Sigma_\pi = \mathsf{ELBA.Read}^{H,W}(\pi)$, let V be the random variable taking the value of $\mathsf{ELBA.Verify}^{H,W}(\pi)$.

$$\begin{aligned}\Pr[\exists s \in (\Sigma_\pi \cap W) \setminus \Sigma] &\ge \Pr[\Sigma_\pi \not\subset \Sigma, \Sigma_\pi \subset W] \\ &\ge \Pr[\Sigma_\pi \subset W, \Sigma_\pi \not\subset \Sigma, V = 1] \\ &= \Pr[(\Sigma_\pi \subset W \wedge V = 1), \neg(\Sigma_\pi \subset \Sigma \wedge V = 1)] \\ &\ge \Pr[\Sigma_\pi \subset W, V = 1] - \Pr[\Sigma_\pi \subset \Sigma \wedge V = 1] \\ &\ge \mathsf{Adv}^{\mathsf{sound}}_{\mathsf{ELBA},\mathcal{A}} - 2^{-\lambda_{sound}}\end{aligned}$$

Where the equality follows by $\Pr[(A \wedge B) \wedge \neg(C \wedge B)] = \Pr[(A \wedge B) \wedge (\neg C \vee \neg B)] = \Pr[A \wedge B \wedge \neg C]$, and the final inequality follows by

[6] History tells us that extraction from Elba is possible, we show the corresponding result for ELBA.

the soundness of ALBA. For $\varepsilon = \mathsf{Adv}^{\mathsf{sound}}_{\mathsf{ELBA},\mathcal{A}} - 2^{-\lambda_{sound}} > 0$ the loop will run in an expected time $\mathsf{poly}(T, n_f, 1/\varepsilon)$ before $|\Sigma| > n_f$. □

B.4 Omitted proofs from Section 7.1

Proof of Lemma 7.1. This follows from two hybrids. In the first hybrid, sample a fresh π^* when the adversary queries π on a new $r \in \mathcal{K}$. This is indistinguishable unless the adversary queries π_r, which is bounded by $q/|\mathcal{K}|$. In the second hybrid, replace π^* with a random function; $C^{\pi^*}(m, r)$ also appears uniform in this hybrid, as each of the n blocks are independent given disjoint inputs. First and second hybrids are indistinguishable by the PRP/PRF switching lemma [8, Lemma 1] with advantage $q(q-1)/2^{\ell+1}$ where q is the number of queries to π. The view of the adversary in the second hybrid is independent of the committed message. □

Proof of Lemma 7.2. Consider an adversary having made i previous queries $\{((k_j, m_{j,1}, \cdots, m_{j,l}, y_{j,1}, \ldots, y_{j,n})\}_{j\in[i]}$ such that $y_{j,l} = \pi_{k_j}(0\ldots0||\mathsf{bits}(l)||m_{j,l})$ for $l \in [n]$. The adversary finds a collision to break binding if it makes a fresh query $(k, m_1, ..., m_n)$ such that $(m_1, ..., m_n) \neq (m_{j,1}, ..., m_{j,n})$ for $j \in [i]$ where π_k matches π_{k_j} on $m = (m_1, ..., m_n)$ and $m_j = (m_{j,1}, ..., m_{j,n})$ respectively.

Without loss of generality, assume that the adversary has made i previous queries for keys $k_{j\in[i]}$ and for each query with a key, it obtains oracle responses $y_j^1, \ldots, y_j^{|m|}$ for all $m \in 2^d$ (constant message space). Then, the probability of a fresh query $(m = (m_1, ..., m_n), k)$ colliding with a prior query with key different k_j and a message $m' = m$ is simply bounded by the probability of a fresh query colliding with one of the prior oracle responses with a different key.

For each prior query $j \in [i]$, let $(y_{j,1}, ..., y_{j,n})$ denote the oracle response from any prior query (m_j, k_j). Then, the collision event between fresh query (m, k) and prior query $(m_j \neq m, k_j)$ is bounded by

$$\begin{aligned}
&\Pr\left[y_{j,1} = \pi_k(0\ldots0||\mathsf{bits}(1)||m_1),\ \ldots\ , y_{j,n} = \pi_k(0\ldots0||\mathsf{bits}(n)||m_n)\right] \\
&\quad\le \left(\frac{2^{d/n}}{2^\ell}\right)\left(\frac{2^{d/n}}{2^\ell - 2^{d/n}}\right)\cdots\left(\frac{2^{d/n}}{2^\ell - (n+2)2^{d/n}}\right) = 1 \Bigg/ \left(\prod_{r=0}^{n-2} 2^\ell/2^{d/n} - r\right) \\
&\quad= O\left(\left(2^{\ell-d/n}\right)^{-n}\right).
\end{aligned}$$

where the last equality holds for constant n.

For each new query we apply a union bound over the collision event probability with each previous query. Bounding across all queries, the combined collision probability is then at most $\sum_{i=2}^{q} i \cdot O\left((2^{\ell-d/n})^{-n}\right) = O(q^2(2^{\ell-d/n})^{-n})$. □