



**HAL**  
open science

## **Fully Dynamic Attribute-Based Signatures for Circuits from Codes**

San Ling, Khoa Nguyen, Duong Hieu Phan, Khai Hanh Tang, Huaxiong Wang,  
Yanhong Xu

► **To cite this version:**

San Ling, Khoa Nguyen, Duong Hieu Phan, Khai Hanh Tang, Huaxiong Wang, et al.. Fully Dynamic Attribute-Based Signatures for Circuits from Codes. PKC 2024 : Public-Key Cryptography – PKC 2024, Apr 2024, Sydney, Australia. pp.37-73, <10.1007/978-3-031-57718-5\_2>. <hal-04577056>

**HAL Id: hal-04577056**

**<https://telecom-paris.hal.science/hal-04577056v1>**

Submitted on 15 May 2024








**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization


# Fully Dynamic Attribute-Based Signatures for Circuits from Codes

San Ling <sup>1</sup>, Khoa Nguyen <sup>2</sup>, Duong Hieu Phan <sup>3</sup>, Khai Hanh Tang <sup>1</sup>,  
Huaxiong Wang <sup>1</sup>, and Yanhong Xu <sup>4</sup>

<sup>1</sup> Nanyang Technological University, 50 Nanyang Ave, Singapore 639798, Singapore

<sup>2</sup> University of Wollongong, Northfields Avenue, Wollongong, NSW 2522, Australia

<sup>3</sup> Telecom Paris, Institut Polytechnique de Paris, 19 place Marguerite Perey CS  
20031, F-91123 Palaiseau Cedex, France

<sup>4</sup> Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai 200240, China,  
 [yanhong.xu@sjtu.edu.cn](mailto:yanhong.xu@sjtu.edu.cn)

**Abstract.** Attribute-Based Signature (ABS), introduced by Maji et al. (CT-RSA’11), is an advanced privacy-preserving signature primitive that has gained a lot of attention. Research on ABS can be categorized into three main themes: expanding the expressiveness of signing policies, enabling new functionalities, and providing more diversity in terms of computational assumptions. We contribute to the development of ABS in all three dimensions, by providing a fully dynamic ABS scheme for arbitrary circuits from codes. The scheme is the first ABS from code-based assumptions and also the first ABS system offering the **full dynamicity** functionality (i.e., attributes can be enrolled and revoked simultaneously). Moreover, the scheme features much shorter signature size than a lattice-based counterpart proposed by El Kaafarani and Katsumata (PKC’18). In the construction process, we put forward a new theoretical abstraction of Stern-like zero-knowledge (ZK) protocols, which are the major tools for privacy-preserving cryptography from codes. Our main insight here actually lies in the questions we ask about the fundamental principles of Stern-like protocols that have remained unchallenged since their conception by Stern at CRYPTO’93. We demonstrate that these long-established principles are not essential, and then provide a refined framework generalizing existing Stern-like techniques and enabling enhanced constructions.

## 1 Introduction

**Attribute-Based Signatures.** Introduced by Maji et al. [68], attribute-based signature (ABS) is an advanced signature primitive that simultaneously provides fine-grained authentications and protects the privacy of signers. In an ABS scheme, a user possessing an attribute  $x$  certified by an authority can anonymously sign message  $M$  along with a policy  $P$ , as long as  $x$  satisfies the given policy, i.e.,  $P(x) = 1$ . Thanks to its versatility and privacy-preserving features,

ABS may find applications in various contexts, such as attribute-based messaging, attribute-based authentication and trust negotiation, leaking secrets and non-transferable access control (see [68,54] for comprehensive discussions). Since the pioneering work of Maji et al. [68], significant attention has been paid to the developments of ABS systems, which can be categorized into the following three major research themes.

Similar to other access-control primitives, a prominent line of work in ABS is devoted to expanding the expressiveness of the class of signing policies a given ABS can allow. Okamoto and Takashima [75] proposed efficient ABS schemes for non-monotone access structures, improving Maji et al.'s schemes that can only handle monotone ones. Systems allowing more expressive policy families, such as bounded-depth circuits [85], unbounded arithmetic branching programs [31] and non-deterministic finite automata [80], were subsequently developed. This line of research reached high success with constructions supporting very general policies, in the form of arbitrary circuits [79,37] and Turing machines [80].

The second important direction focuses on defining and designing ABS systems with new functionalities. Examples of these features include decentralization [76] (that removes the need for a central authority), traceability [38,36] (that incorporates a group-signature-like [27] opening mechanism), linkability [35,87] (that allows to link two ABS signatures under certain conditions), forward security [89], hierarchy [33] and revocability [55,84,13,83,46]. Among them, revocability is arguably one of the most non-trivial functionalities to achieve. In the original model of ABS [68], the authority can enroll new signing keys for attributes, but the model does not support revocations of certified attributes. In fact, for advanced multi-user systems, efficient key revocation is a desirable feature (e.g., to address situations such as membership terminations or key misuses), yet typically challenging to realize, since one has to ensure that revoked keys are no longer usable without having to reinitialize the system or affecting other key owners. While in the related context of group signatures [27], nice solutions have been proposed, e.g., [61,60,16], existing proposals for ABS are still somewhat unsatisfactory. In a nutshell, they either suggest to revoke users' identities (which is an artificial and unnecessary concept in the ABS setting) instead of attributes; or they do not propose a clear model for handling revocations.

The third major research direction aims to provide more diversity regarding the pool of computational assumptions used to instantiate ABS. A long line of pairing-based constructions started with Maji et al.'s work [68]. The first scheme that does not rely on pairings was suggested by Herranz [49]. The recent emergence of post-quantum cryptography raised interest in designing ABS from post-quantum assumptions, and lattice-based constructions [7,37,46] have been introduced. To our knowledge, no other ABS from alternative post-quantum foundations, e.g., codes, multivariates, isogenies, has been proposed.

In this work, we aim to contribute to the development of ABS systems in all three dimensions discussed above. In terms of supporting techniques, we will also enhance the area of code-based zero-knowledge protocols, specifically, those that operate in Stern's framework [81,82].

**Code-Based and Stern-like ZK Protocols.** A beautiful and influential cryptographic notion introduced by Goldwasser, Micali and Rackoff [48], zero-knowledge (ZK) protocols allow to prove the truth of a statement without leaking any additional information. In the last three decades or so, ZK protocols have become a foundational subject of study and essential building blocks in the development of countless cryptographic constructions. Specifically, via the Fiat-Shamir transform [45], ZK protocols have been the basis for developing ordinary signature schemes, including those in NIST’s Post-Quantum Cryptography project<sup>5</sup>, such as Dilithium [66,34] and Picnic [25,52,90]. An equally important application domain of ZK protocols comprises privacy-preserving authentication systems, such as group signatures [27], ring signatures [78], anonymous credentials [26], functional signatures [20], policy-based signatures [8], mesh signatures [19], and – of our particular interest – attribute-based signatures [68].

In the context of code-based ZK, a recent line of work [43,42,14,24] has obtained efficient Fiat-Shamir-based ordinary signatures by cleverly employing the MPC-in-the-head paradigm [50] for variants of the Syndrome Decoding problem. However, for developing advanced code-based privacy-preserving cryptosystems, where sophisticated algebraic structures are required, the major technical stepping stone is still Stern’s protocol [81,82].

The original protocol of Stern addresses the following relation

$$\mathbf{R}_{\text{Stern}} = \left\{ \left( (\mathbf{M}, \mathbf{v}), \mathbf{w} \right) \in (\mathbb{Z}_2^{D_0 \times D} \times \mathbb{Z}_2^{D_0}) \times \mathbf{B}(D, \omega) : \mathbf{M} \cdot \mathbf{w} = \mathbf{v} \right\},$$

where  $D, \omega \in \mathbb{Z}^+$  such that  $D > \max\{D_0, \omega\}$ , and  $\mathbf{B}(D, \omega)$  denotes the set of all vectors in  $\{0, 1\}^D$  having Hamming weight  $\omega$ . The protocol is a  $\Sigma$ -protocol [29] in the generalized sense defined in [51,11] (where three valid transcripts are needed for extraction, instead of just two), and is based on the following two main ideas.

- (i) To prove the linear equation  $\mathbf{M} \cdot \mathbf{w} = \mathbf{v}$ , use a uniformly random  $\mathbf{r} \in \{0, 1\}^D$ , and prove instead that  $\mathbf{M} \cdot \mathbf{z} = \mathbf{M} \cdot \mathbf{r} \oplus \mathbf{v}$ . Here,  $\mathbf{z} = \mathbf{w} \oplus \mathbf{r}$ , where  $\oplus$  denotes the addition modulo 2, is uniformly random over  $\{0, 1\}^D$ , as  $\mathbf{r}$  acts as a one-time pad.
- (ii) To prove that  $\mathbf{w} \in \mathbf{B}(D, \omega)$ , use a uniformly random permutation  $\phi$  in the symmetric group  $\mathcal{S}_D$  to permute the coordinates of  $\mathbf{w}$ . The permuted vector  $\mathbf{t} = \phi(\mathbf{w})$  is then uniformly distributed over  $\mathbf{B}(D, \omega)$ .

Moreover, the two ideas are compatible with each other thanks to the homomorphism  $\phi(\mathbf{z}) = \phi(\mathbf{w}) \oplus \phi(\mathbf{r})$ , since there are two different ways to compute  $\phi(\mathbf{z})$ , based on either  $(\phi, \mathbf{z})$  or  $(\phi(\mathbf{w}), \phi(\mathbf{r}))$ . That is, the prover can show the honest computation of  $\phi(\mathbf{z})$  via either of the pairs, depending on the verifier’s challenge. Since 1993, there have been a large number of works built upon Stern’s ideas. Most notably, they enabled privacy-preserving code-based constructions such as proofs of plaintext knowledge [70], proofs of valid openings for commitments and proofs for general relations [51] and committed Boolean functions [62],

<sup>5</sup> <https://csrc.nist.gov/Projects/post-quantum-cryptography>

ring signatures [30,69,21,74], group signatures [40,41,2,74], group encryption [73], accumulators and range proofs [74].

Stern’s ideas have also been shown useful in the lattice setting, including the first proposals [53,63] of “exact” ZK protocols<sup>6</sup> for the Short Integer Solution [1] and Learning With Errors [77] problems, as well as the first lattice-based group signatures [58,59] without trapdoors [47], the first policy-based signatures [28], accountable tracing signatures [64], multimodal private signatures [71], bicameral and auditably private signatures [72].

In terms of developing Stern’s framework, Libert et al. [56] proposed an abstraction of Stern-like protocols that captures the contexts where one’s witness vectors may simultaneously be involved in many linear equations. In Libert et al.’s formulation, the set of valid witnesses is generalized from  $B(D, \omega)$  to some set  $\text{VALID} \subset \{0, 1\}^D$ . Furthermore, to handle vectors  $\mathbf{w}$  whose coordinates are arranged according to certain patterns, the random permutations applied to  $\mathbf{w}$  are not necessarily uniform over  $\mathcal{S}_D$ , e.g., they could be uniformly random over some fine-grained subset  $\mathcal{S}'$  of  $\mathcal{S}_D$ .

However, Libert et al.’s work as well as all other existing Stern-like protocols all methodically adhere to the two original ideas, namely, use uniformly random permutations of coordinates and uniformly random masking vectors over the entire space  $\{0, 1\}^D$ . In this paper, we *challenge these fundamental ideas*. The new perspectives acquired during our revisiting process could then help us to move forward with new formulations, which in turn, could potentially inspire new and enhanced cryptographic applications, in particular, code-based ABS.

**OUR CONTRIBUTIONS.** We make several contributions to the area of attribute-based signatures. In the process, we also introduce new insights and techniques for Stern-like protocols, which help enhance our code-based ABS scheme and could be of independent interest. Our contributions are summarized as follows.

**Contributions to attribute-based signatures.** Our results for ABS subsume all the three discussed aspects: additional functionality, expressiveness of signing policies and diversity of (post-quantum) computational assumptions.

We first propose a model for fully dynamic attribute-based signatures (FD-ABS). By “full dynamicity”, we mean that the system simultaneously supports: (i) Dynamic enrollments of new attributes; (ii) Key updates for users who changed their attributes; and (iii) Revocations of expired/misused keys. Our model is equipped with rigorous definitions and stringent security requirements that extend the privacy and unforgeability notions put forward by Maji et al. [68] to the fully dynamic setting.

Next, we provide an instantiation of FDABS based on codes. Being the first code-based ABS, our scheme helps to enrich the pool of ABS from post-quantum assumptions. In terms of policy expressiveness, the scheme supports arbitrary Boolean circuits. In the quantum random oracle model (QROM), we prove that the scheme satisfies the proposed security requirements for FDABS, based on

<sup>6</sup> “Exactness” here roughly means that there is no difference between the language used for defining ZK and the one ensured by soundness. This is a desirable feature for lattice-based ZK proofs and arguments. See, e.g., [88,18,39].

a well-studied variant of the Syndrome Decoding problem [5,6,12]. In terms of efficiency, our scheme has signature size  $\tilde{\mathcal{O}}(C \cdot \lambda + \lambda^2)$ , where  $C$  denotes the size of a circuit representing the signed policy, and  $\lambda$  is the security parameter.

Prior to our work, the only ABS construction for arbitrary circuits based on well-studied post-quantum assumptions is the one introduced in [37], which has signature size  $\tilde{\mathcal{O}}(C \cdot \lambda^2 + \lambda^3)$ . Our signature size is smaller by a factor  $\tilde{\mathcal{O}}(\lambda)$ , which is quite a surprising and counterintuitive feat, as code-based signatures are generally considered to be much inferior to their lattice-based counterparts in terms of efficiency. Here, the improvement is achieved thanks to an enhanced approach that allows us to prove circuit satisfiability in a direct manner and that is compatible with our refined abstraction of Stern (discussed below). In a nutshell, in [37] one has to commit to all the input and output wires of the circuit and prove relations among them in zero-knowledge with communication cost  $\tilde{\mathcal{O}}(\lambda^2)$  bits for each gate. Here, in contrast, we prove the circuit satisfiability directly (without relying on commitments) by reducing it to  $C$  simple binary equations that involve witness vectors constructed in a specific way. Then, our new Stern-like techniques help us to prove with communication cost 4 bits per gate for soundness error  $2/3$  and  $\mathcal{O}(\lambda)$  bits per gate for soundness error  $2^{-\lambda}$ . Moreover, while the ABS scheme in [37] was only analyzed in the ROM [9], our scheme here is proven secure in the QROM [15] and it is technically the first post-quantum ABS for arbitrary circuits.<sup>7</sup>

In Table 1, we provide a comparison among known ABS schemes for expressive policy classes and from well-established assumptions. Note that, while we obtain noticeable improvements over previous work, our scheme is still far from being practical (with an estimated signature size of tens MBs). We therefore mainly view our work as a theoretical one and consider the problem of designing practically usable post-quantum ABS for circuits as a fascinating open question.

**Contributions to Stern-like and code-based ZK protocols.** We revisit the fundamental ideas Stern introduced in his seminal work [81,82], namely, the uses of uniformly random permutations of coordinates and uniformly random masking vectors. Viewing them from new angles, we observe a highly intriguing fact: these ideas are not essential for Stern-like protocols! Indeed, we show that there are examples where it is unnecessary (and sub-optimal) to use a uniformly random masking  $\mathbf{r}$  or a uniformly random permutation  $\phi$  to hide witness vector  $\mathbf{w}$ . These new understandings inspire us to formulate an abstract ZK protocol for proving linear relations of binary witness vectors, with the following nice features.

- It captures previous approaches and techniques for Stern-like protocols, including Stern’s original work [81,82], Libert et al.’s formulation [56], as well as those used in code-based constructions such as [69,70,51,40,2,74,73].
- It paves the way for the designs of more efficient protocols in which one may securely hide a witness vector  $\mathbf{w}$  via non-traditional methods: a mask-

<sup>7</sup> To be fair, it could also be possible make the scheme in [37] secure in the QROM. Intuitively, the scheme also employs Stern-like protocols, and can as well benefit from the variant of Unruh’s transform [86] presented in [44].

Scheme	Policy expressiveness	Assumptions	SM/ (Q)ROM	Signature size	Fully dynamic
OT11 [75]	Non-monotone access structures	pairings	SM	$\mathcal{O}(S \cdot \lambda)$	$\times$
SAH16 [79]	Arbitrary circuits	pairings	SM	$\mathcal{O}(C \cdot \lambda)$	$\times$
SKAH18-1 [80]	Turing machines	pairings	SM	$\mathcal{O}(T^2 \cdot \lambda)$	$\times$
SKAH18-2 [80]	Non-deterministic finite automata	pairings	SM	$\mathcal{O}(W \cdot \lambda)$	$\times$
DOT19 [31]	Branching programs	pairings	SM	$\mathcal{O}(L \cdot \lambda)$	$\times$
Tsa17 [85]	Bounded-depth circuits	lattices	SM	$\tilde{\mathcal{O}}(D \cdot \lambda)$	$\times$
EKK18 [37]	Arbitrary circuits	lattices	ROM	$\tilde{\mathcal{O}}(C \cdot \lambda^2 + \lambda^3)$	$\times$
Ours	Arbitrary circuits	codes	QROM	$\tilde{\mathcal{O}}(C \cdot \lambda + \lambda^2)$	$\checkmark$

**Table 1:** A summary of known ABS for expressive policy classes and from well-established computational assumptions. For all schemes,  $\lambda$  denotes the security parameter. For [75],  $S$  denotes the access structure size; For [80],  $T$  and  $W$  denote the running time of a Turing machine and the input length of a finite automaton, respectively; For [31],  $L$  denotes the branching program length; For [85],  $D$  denotes the circuit depth; For [79,37] and ours,  $C$  denotes the circuit size.

ing vector  $\mathbf{r}$  that is not necessarily uniform, or a function  $F(\cdot)$  that is not necessarily a permutation.

In particular, these new views allow us to obtain sub-protocols with enhanced efficiency for commonly seen statements in code-based privacy-preserving cryptography, such as proving knowledge of a binary vector of fixed Hamming weight [81,82] or with special arrangements of non-zero bits [40,58,74], or for handling products of secret bits [57]. These improved sub-routines are helpful for our instantiation of ABS from codes.

We would like to stress that our major innovation here actually lies in *the questions we ask* about the fundamental principles of Stern-like protocols that have been remaining unchallenged since their conception in 1993. Once the questions are spot on, the counterexamples as well as the refinements would come rather naturally and might look somewhat “simple” *in hindsight*. We also would like to admit that our refinements have not yielded a noteworthy efficiency improvement for the cryptographic applications we are currently aware of. In particular, our techniques of replacing uniform masks by odd-weight masks and replacing coordinate permutations by affine functions only give improvement factor  $\mathcal{O}(1)$ , which could be considered small – in theory and in practice. Nevertheless, we believe that our refined framework has the *potential* to enable more

significant improvements for cryptographic applications, which we have been unable to fully exploit.

**TECHNICAL OVERVIEW.** Let us give a high-level discussion for each of our technical contributions.

**Defining FDABS.** In order to bring full dynamicity into the context of ABS, we adapt the definitional framework of Bootle et al. [16,17] in their work on fully dynamic group signatures, which captures the two most commonly adopted approaches for revocations, namely, accumulator-based [23] and revocation-list-based [22]. Similar to [16,17], the lifetime of an FDABS system is divided into time epochs. We assume that the specification of the epochs (e.g., from 8:00:00am UTC on Monday to 7:59:59am UTC on next Monday) is publicly known. We also assume that the updated system information is announced at the beginning of each epoch and is publicly accessible. For example, such information may be appended to a public database at 8:00:00am UTC on each Monday. Hence, the signers and verifiers are required to download the updated information at most once per epoch. Such a requirement is natural and necessary to formulate the correctness and security properties in the fully dynamic setting. In fact, unlike the partially dynamic setting (in which users can join the system at any time but cannot be revoked) – where the public parameters do not need to change over time, all known fully dynamic signatures inherently demand some updates of system information per time period: either in the form of an updated keys or updated lists of active/revoked users.

In our model, a signature  $\Sigma$  is always attached to a message-epoch pair  $(M, \tau)$ . To verify  $\Sigma$ , one should use the corresponding system information at epoch  $\tau$  – which can be publicly downloaded if it has not been in one’s local storage. We formalize the stringent security requirements of privacy and unforgeability for FDABS, which are extended from those in Maji et al.’s model [68].

Regarding privacy, the adversary is provided with the strongest capability: it is allowed to maliciously generate the keys of the authority and even the challenge attributes. Furthermore, different flavors of privacy are presented to capture different computational powers of the adversary.

In terms of unforgeability, we formulate a strict notion, in which an attribute is not authorized to sign unless it is not revoked and also active at the epoch associated with the signature. Furthermore, a signature generated at epoch  $\tau$  should only be verified w.r.t.  $\tau$ , i.e., verification w.r.t. any different epoch  $\tau'$  should fail. This, in particular, eliminates the possibility that an attribute is used to sign before it is introduced into the system.

**Designing code-based FDABS for circuits.** A typical approach for designing ABS is “sign-then-prove”, which relies on an ordinary signature scheme and a zero-knowledge protocol. The signing key for an attribute  $\mathbf{x}$  is then set as a signature  $\mathbf{s}$  of the authority on “message”  $\mathbf{x}$ . When signing with policy  $P$ , one proves knowledge of a message-signature pair  $(\mathbf{x}, \mathbf{s})$  that is valid under the authority’s public key, and additionally proves that  $P(\mathbf{x}) = 1$ . This approach, in particular, was used in existing lattice-based ABS in the ROM [7,37,46].

In the code-based setting, however, ordinary signature schemes compatible with ZK proofs of a message-signature pair are currently unavailable. Note that the state-of-the-art code-based signature schemes, such as [4,32,42,14,24], resort to the (Q)ROM, and are unsuitable for our purpose – since it is not known how to efficiently prove in ZK the knowledge of a message hashed by a RO. Thus, at a high level, the current lack of code-based signature schemes in the standard model is arguably the major reason why constructing ABS from codes remains an open question to date.

To overcome the above issue, we start with a somewhat disparate approach, which we name “commit-then-accumulate-then-prove”. Specifically, we employ different technical building blocks: an updatable Merkle-tree accumulator [74,73], a commitment scheme [74] and a companion ZK protocol. The general idea is inspired by the constructions from [65,73], but here, we put commitments  $\mathbf{d} = \text{com}(\mathbf{x}, \mathbf{r})$  to attributes  $\mathbf{x}$  at leaves of the tree. Then, when signing with policy  $P$ , one proves that: (i)  $\mathbf{d}$  is contained in the tree; (ii)  $\mathbf{d}$  is a valid commitment to some attribute  $\mathbf{x}$ ; (iii)  $\mathbf{x}$  satisfies the Boolean circuit relation  $P(\mathbf{x}) = 1$ . Here, (ii) can be viewed as a bridge connecting two layers (i) and (iii).

In fact, a ZK protocol for the combined relation above can be developed based on existing Stern-like techniques from [74] – for the Merkle trees and commitments, and [51] – for circuits. However, [51] requires to commit to all input and output wires in the circuit, which yields communication cost  $\mathcal{O}(C \cdot \lambda^2)$  for the circuit layer (similar to the situation in [37]).

Fortunately, as we will discuss below, our refined framework for Stern-like protocols together with our techniques for proving  $(x_1 \text{ NAND } x_2) \oplus x_3 = 0$  enable us to handle the NAND gates in the circuit and realize the sub-protocols in a much more efficient manner. In particular, we manage to reduce the communication cost for the circuit layer to  $\mathcal{O}(C \cdot \lambda)$ .

Let us now discuss how we enroll and revoke an attribute  $\mathbf{x}$ . Initially, all the leaves of the tree are associated with  $\mathbf{0}$ . To enroll  $\mathbf{x}$ , we commit it to  $\mathbf{d}$ , and add  $\mathbf{d}$  to the tree if  $\mathbf{d}$  has *odd Hamming weight* (which happens with probability negligibly close to  $1/2$ ). To revoke  $\mathbf{x}$ , we set  $\mathbf{d}$  back to  $\mathbf{0}$  and update the tree accordingly. When signing, one additionally proves that attribute  $\mathbf{x}$  is active (i.e., it has been enrolled and has not been revoked) by showing that  $\mathbf{d}$  has odd weight – which then can be done very efficiently by proving that the inner product of  $\mathbf{d}$  and the all-1 vector is 1. We remark that, in previous works [65,73], an active  $\mathbf{d}$  was set to be non-zero, causing a relatively inefficient sub-protocol for proving  $\mathbf{d} \neq \mathbf{0}$ . That sub-protocol relies on an extension trick from [63]. The latter requires the use of a random permutation of bit-size  $\mathcal{O}(n \log n)$ , where  $n$  is the dimension of  $\mathbf{d}$ . Here, in contrast, by working with odd-weight  $\mathbf{d}$ , our cost is only  $n$  bits.

**Revisiting Stern.** Recall that Stern’s two fundamental ideas [81,82] are to employ a one-time pad  $\mathbf{r} \in \{0, 1\}^D$  to additively mask witness  $\mathbf{w}$ , and to use a uniformly random permutation  $\phi \in \mathcal{S}_D$  to permute the coordinates of  $\mathbf{w}$ .

Our first observation is that  $\mathbf{r}$  does not necessarily have to be uniform over the entire space  $\{0, 1\}^D$ . This sounds particularly counterintuitive (and contrary to

the usual notion of one-time pads), but we can demonstrate that it is true when  $\mathbf{w}$  has a specific constraint. For instance, let us consider the case  $\mathbf{w} \in \mathcal{B}(D, \omega)$  as in [81,82], and denote by  $\mathcal{B}_{\text{even}}^D$  (resp.,  $\mathcal{B}_{\text{odd}}^D$ ) the set of all length- $D$  vectors with even (resp., odd) Hamming weight. Then, if we use  $\mathbf{r} \stackrel{\$}{\leftarrow} \mathcal{B}_{\text{odd}}^D$  to mask  $\mathbf{w}$ , the sum  $\mathbf{z} = \mathbf{w} \oplus \mathbf{r}$  will be uniformly random in the set  $\mathcal{B}_q^D$ , where  $q = \text{odd}$  if  $\omega$  is an even integer and  $q = \text{even}$  otherwise. Here, we note that elements of  $\mathcal{B}_{\text{even}}^D$  and  $\mathcal{B}_{\text{odd}}^D$  can be described by  $(D - 1)$  bits instead of  $D$  bits, yielding smaller complexity.

Another example is for vectors of the form  $\mathbf{w} = (\mathbf{w}_1 \| \dots \| \mathbf{w}_D) \in \{0, 1\}^{2D}$ , where each  $\mathbf{w}_i \in \{01, 10\}$ , as considered in code-based privacy-preserving constructions like [40,74]. If for each  $\mathbf{w}_i$ , we apply a mask  $\mathbf{r}_i \stackrel{\$}{\leftarrow} \{01, 10\}$ , then the sum  $\mathbf{z}_i = \mathbf{w}_i \oplus \mathbf{r}_i$  will be uniformly random over  $\{00, 11\}$ . Because elements of  $\{01, 10\}$  and  $\{00, 11\}$  can be represented by just 1 bit, we can reduce the communication cost for proving knowledge of  $\mathbf{w}$  from  $2D$  bits to  $D$  bits.

Inspired by the above observation, we suggest a new abstraction for Stern-like ZK protocols, in which we assume the existence of (possibly proper) subsets  $\text{VALID}$ ,  $\mathcal{R}$ ,  $\mathcal{Z}$  of  $\{0, 1\}^D$ , such that for all witness vectors  $\mathbf{w} \in \text{VALID}$ , the distributions of  $\{\mathbf{z} = \mathbf{w} \oplus \mathbf{r} \mid \mathbf{r} \stackrel{\$}{\leftarrow} \mathcal{R}\}$  and  $\{\mathbf{z} \stackrel{\$}{\leftarrow} \mathcal{Z}\}$  are identical. We remark that, while this refined masking framework only helps to save a small portion of overall communication cost for our zero-knowledge protocols (Section 4), the underlying motive could potentially be more useful in related contexts. For example, if one only requires statistical closeness between the two distributions, then one might possibly work with sets  $\mathcal{R}, \mathcal{Z}$  of smaller cardinalities, via techniques such as rejection samplings [66,32].

Next, we ask the question of whether it is really essential to employ permutations of coordinates to prove the membership of witness  $\mathbf{w}$  in some predetermined set  $\text{VALID}$ , as done in all previous Stern-like protocols. To answer this question, we abstract out the properties we would need for a function  $F$  that can be used to prove  $\mathbf{w} \in \text{VALID}$  and that is compatible with our linear masking framework discussed above. Let us define  $F$  as  $F : \mathcal{S} \times \{0, 1\}^D \rightarrow \{0, 1\}^D$ , where  $\mathcal{S}$  is a finite set. Then, we would need  $F$  to satisfy the following requirements.

First,  $F$  must “behave nicely” w.r.t. the set  $\text{VALID}$ , namely, for all  $\phi \in \mathcal{S}$ , we require that  $\mathbf{t} = F(\phi, \mathbf{w}) \in \text{VALID}$  if and only if  $\mathbf{w} \in \text{VALID}$  and furthermore,  $\mathbf{t}$  is uniform whenever  $\phi$  is uniform (in their respective sets). Second, we define functions  $F' : \mathcal{S} \times \mathcal{R} \rightarrow \mathcal{R}$  (for the mask  $\mathbf{r}$ ) and  $F'' : \mathcal{S} \times \mathcal{Z} \rightarrow \mathcal{Z}$  (for the sum  $\mathbf{z}$ ), and demand that they together with  $F$  satisfy a “homomorphism” property:

$$F'(\phi, \mathbf{r}) \oplus F''(\phi, \mathbf{z}) = F(\phi, \mathbf{r} \oplus \mathbf{z}), \quad \forall (\phi, \mathbf{r}, \mathbf{z}) \in \mathcal{S} \times \mathcal{R} \times \mathcal{Z}.$$

Our new formulations capture the case of Stern’s original protocol, with  $\mathcal{S} := \mathcal{S}_D$ ,  $\mathcal{R} = \mathcal{Z} = \{0, 1\}^D$  and  $F(\phi, \cdot) = F'(\phi, \cdot) = F''(\phi, \cdot) = \phi(\cdot)$ . These formulations are particularly helpful for a technical step in our protocol of Section 4.3, where we would like to prove knowledge of bits  $x_1, x_2, x_3$  such that  $(x_1 \text{ NAND } x_2) \oplus x_3 = 0$  and  $x_1, x_2, x_3$  may satisfy other relations (e.g., they are committed and may appear in other wires in the considered circuit).

To the above end, we would need highly non-trivial definitions of the corresponding sets and functions. Specifically, letting  $\bar{b} = b \oplus 1$  for any bit  $b$ , we encode  $(x_1, x_2, x_3)$  as  $\mathbf{w} \in \{0, 1\}^4$  of the form

$$\text{ENC}(x_1, x_2, x_3) := [\bar{x}_1 \cdot \bar{x}_2 \oplus x_3 \mid \bar{x}_1 \cdot x_2 \oplus x_3 \mid x_1 \cdot \bar{x}_2 \oplus x_3 \mid x_1 \cdot x_2 \oplus x_3]^\top.$$

Next, we define  $\text{valid} := \{\text{ENC}(x_1, x_2, x_3) \mid (x_1, x_2, x_3) \in \{0, 1\}^3\}$ , and prove instead that  $\mathbf{w} \in \text{valid}$  and its last coordinate is equal to 1.

In the process, we note that  $\text{valid} = \mathcal{B}_{\text{odd}}^4$ , allowing us to use our refined masking framework (with  $\mathcal{R} = \mathcal{B}_{\text{odd}}^4$ ,  $\mathcal{Z} = \mathcal{B}_{\text{even}}^4$ ) for proving linear equation  $[0 \mid 0 \mid 0 \mid 1] \cdot \mathbf{w} = 1$  – which implies that the last coordinate of  $\mathbf{w}$  is 1.

Meanwhile, we would need specifically designed functions  $F, F', F''$  to prove that  $\mathbf{w}$  is a well-formed element of  $\text{valid}$ . To this end, we employ a permuting function  $T : \{0, 1\}^2 \times \{0, 1\}^4 \rightarrow \{0, 1\}^4$ , first suggested in [57], that, on input  $(e_1, e_2)$  and  $\mathbf{y} = [y_{0,0} \mid y_{0,1} \mid y_{1,0} \mid y_{1,1}]^\top$ , outputs  $[y_{e_1, e_2} \mid y_{e_1, \bar{e}_2} \mid y_{\bar{e}_1, e_2} \mid y_{\bar{e}_1, \bar{e}_2}]^\top$ . Then, we let  $\mathcal{S} = \{0, 1\}^3$  and define

$$\begin{aligned} F : \mathcal{S} \times \{0, 1\}^4 &\rightarrow \{0, 1\}^4, & F((e_1, e_2, e_3), \mathbf{y}) &= T((e_1, e_2), \mathbf{y}) + [e_3 \mid e_3 \mid e_3 \mid e_3]^\top; \\ F' : \mathcal{S} \times \mathcal{R} &\rightarrow \mathcal{R}, & F'((e_1, e_2, e_3), \mathbf{r}) &= T((e_1, e_2), \mathbf{r}); \\ F'' : \mathcal{S} \times \mathcal{Z} &\rightarrow \mathcal{Z}, & F''((e_1, e_2, e_3), \mathbf{z}) &= T((e_1, e_2), \mathbf{z}) + [e_3 \mid e_3 \mid e_3 \mid e_3]^\top. \end{aligned}$$

Now, it is crucial to remark that  $F$  does **not** act as a permutation of coordinates (due to the shift of  $[e_3 \mid e_3 \mid e_3 \mid e_3]^\top$ ), but it interacts well with the set  $\text{valid}$ , since we have

$$F((e_1, e_2, e_3), \text{ENC}(x_1, x_2, x_3)) = \text{ENC}(x_1 \oplus e_1, x_2 \oplus e_2, x_3 \oplus e_3).$$

Moreover,  $F, F', F''$  satisfy the described homomorphism property. As a result, we obtain a sub-protocol for proving  $(x_1 \text{ NAND } x_2) \oplus x_3 = 0$ , which can further be extended to additionally prove that  $x_1, x_2, x_3$  satisfy other relations (by using the same bits  $e_1, e_2, e_3$  at those places). We stress that techniques from [57] could also lead to a sub-protocol achieving the same goals, but our refinements here allow to save 1/2 of the communication cost.

Encouraged by the above new insights, new formulations and their usefulness, we put forward a new abstract Stern-like ZK protocol. It serves as a blueprint for the development of the ZK protocol supporting our code-based FDABS construction. It also enables more efficient methods for proving knowledge of binary vectors satisfying various different constraints, and can be used to improve the efficiency of existing and future code-based privacy-preserving constructions.

## 2 Fully Dynamic Attribute-Based Signatures

In this section, we formalize the primitive of fully dynamic attribute-based signature (FDABS). An FDABS scheme involves the following entities: a trusted authority who initializes the system; an attribute-issuing authority who generates attribute keys and periodically announces updated system information; users/signers and signature verifiers.

We let  $\mathcal{X}$  be the universe of possible attributes and  $\mathcal{P} = \{P : \mathcal{X} \rightarrow \{0, 1\}\}$  be a policy family. We say an attribute  $x \in \mathcal{X}$  satisfies a policy  $P \in \mathcal{P}$  if  $P(x) = 1$ .

## 2.1 Syntax

An FDABS scheme consists of the following polynomial-time algorithms.

- Setup<sub>init</sub>**( $1^\lambda$ ): This algorithm, run by a trusted authority, takes as input the security parameter  $1^\lambda$  and generates public parameter  $\mathbf{pp}$ . We assume that  $\mathbf{pp}$  contains the description of an attribute space  $\mathcal{X}$ , a policy family  $\mathcal{P}$ , a time space  $\mathcal{T}$ , and a message space  $\mathcal{M}$ .
- Setup<sub>auth</sub>**( $\mathbf{pp}$ ): This algorithm is run by an attribute-issuing authority. It takes as input  $\mathbf{pp}$  and outputs a key pair ( $\mathbf{mpk}, \mathbf{msk}$ ). The authority also initializes the system information  $\mathbf{info}_0$  and a public registration table  $\mathbf{reg}$ . Note that the record stored in  $\mathbf{reg}$  depends on the scheme specification and may be used by the authority for updating the system information. For simplicity,  $\mathbf{pp}$  and  $\mathbf{mpk}$  are inputs of the following algorithms even not explicitly written.
- AttrGen**( $\mathbf{msk}, x, \mathbf{info}_{\tau_{\text{current}}}, \mathbf{reg}$ ): This algorithm is run by the authority when receiving attribute key generation request from a user. It takes as input the authority's secret key  $\mathbf{msk}$ , an attribute  $x$  and current system information  $\mathbf{info}_{\tau_{\text{current}}}$  and outputs an attribute key (or a signing key)  $\mathbf{sk}_x$  to the user. The authority will then add a new record to the table  $\mathbf{reg}$ .
- Update**( $\mathbf{msk}, \mathcal{S}, \mathbf{info}_{\tau_{\text{current}}}, \mathbf{reg}$ ): This algorithm is run by the authority who will advance the epoch and update system information. Given  $\mathbf{msk}$ , a set  $\mathcal{S} \subset \mathcal{X}$  of to-be-revoked attributes,  $\mathbf{info}_{\tau_{\text{current}}}$ , and  $\mathbf{reg}$ , the authority computes new system information  $\mathbf{info}_{\tau_{\text{new}}}$  and may also update  $\mathbf{reg}$ . If there is no change to the system or  $\mathcal{S}$  contains inactive attributes (which either have not been issued signing keys or have been revoked previously), this algorithm aborts.
- Sign**( $\mathbf{sk}_x, M, P, \mathbf{info}_\tau$ ): This algorithm is run by the user who possesses the signing key  $\mathbf{sk}_x$ . Given  $\mathbf{sk}_x$ , a message  $M \in \mathcal{M}$ , a policy  $P \in \mathcal{P}$ , and  $\mathbf{info}_\tau$ , it returns a signature  $\Sigma$ .
- Verify**( $M, P, \mathbf{info}_\tau, \Sigma$ ): This algorithm is run by any verifier. Given the inputs, it outputs a bit indicating the validity of signature  $\Sigma$  on message  $M$  with respect to policy  $P$  and system information  $\mathbf{info}_\tau$ .

**Additional algorithm.** To ease the notion, we introduce another algorithm that will only be used in the definitions of security requirements.

**IsActive**( $x, \mathbf{info}_\tau$ ): This algorithm returns 1 if  $x$  has been issued a signing key and has not been revoked at time  $\tau$ . We call  $x$  an active attribute. Otherwise, it returns 0 and we call  $x$  an inactive attribute.

*Remark 1.* Algorithm **Update** intends to capture all the changes to the activeness of attributes, which occur between epochs  $\tau_{\text{current}}$  and  $\tau_{\text{new}}$ . Since **Update** takes the registration table  $\mathbf{reg}$  as an input, the changes it makes may include not only the revocations but also the enrollments and re-enrollments of attributes.

An attribute  $x$  could be active at epoch  $\tau_1$ , revoked at epoch  $\tau_2$ , and active again at epoch  $\tau_3$ . To handle such dynamicity, we demand that each signature is bound to a specific epoch: a signature generated by  $x$  in epoch  $\tau_1$  should be rejected when verified w.r.t. to either  $\tau_2$  or  $\tau_3$  (even if  $x$  is active at  $\tau_3$ ).

**Correctness.** Basically, correctness of FDABS demands that: if  $x$  is active at time epoch  $\tau$  and if  $P(x) = 1$ , then a signature  $\Sigma \leftarrow \text{Sign}(\text{sk}_x, M, P, \text{info}_\tau)$  should be accepted by  $\text{Verify}(M, P, \text{info}_\tau, \Sigma)$ . We model this requirement in an adversarial experiment  $\text{Exp}_\mathcal{A}^{\text{correct}}(1^\lambda)$  in Figure 1. Below, we define some oracles available to the adversary.

**AddHX( $x$ ):** This oracle issues a signing key for an attribute  $x$  at current time epoch  $\tau_{\text{current}}$  when invoked by the adversary  $\mathcal{A}$ . If  $x$  has never been issued any signing key, it runs  $\text{sk}_x \leftarrow \text{AttrGen}(\text{msk}, x, \text{info}_{\tau_{\text{current}}}, \mathbf{reg})$  and adds  $x$  to an honestly maintained list  $\text{HL}$ . Otherwise, it returns  $\perp$ . Note that as in the algorithm  $\text{AttrGen}$ , a new record is also added to  $\mathbf{reg}$ .

**Update( $\mathcal{S}$ ):** This oracle allows  $\mathcal{A}$  to remove a set  $\mathcal{S}$  of active attributes from the system at current epoch. It executes  $\text{Update}(\text{msk}, \mathcal{S}, \text{info}_{\tau_{\text{current}}}, \mathbf{reg})$  to get  $\text{info}_{\tau_{\text{new}}}$ , and may also update  $\mathbf{reg}$  as specified in  $\text{Update}$ .

**Definition 1.** Let  $\text{Adv}_\mathcal{A}^{\text{correct}}(1^\lambda) = \Pr[\text{Exp}_\mathcal{A}^{\text{correct}}(1^\lambda) = 1]$  be the advantage of an adversary  $\mathcal{A}$  against correctness of an FDABS scheme in experiment  $\text{Exp}_\mathcal{A}^{\text{correct}}(1^\lambda)$ . An FDABS scheme is correct if, for any PPT adversary  $\mathcal{A}$ , its advantage is negligible in  $\lambda$ .

## 2.2 Formulation of the Security Requirements

Security requirements for static attribute-based signatures (see e.g., [68,79]) are perfect privacy and unforgeability. The former requires signatures not to reveal any information on the attribute beyond the fact that the attribute satisfies the policy. The latter requires that no colluding set of signers (even being able to see signatures on messages of their choices) can create valid signatures under a policy that is not satisfied by any individual attribute in the collusion. Below we carefully extend these two security requirements to the fully dynamic case.

**Privacy.** This notion is adapted from the static case and requires that signatures do not leak the underlying attributes. It protects the signer from a malicious adversary who tries to extract the attribute information from signatures. We model this requirement in adversarial experiments  $\text{Exp}_\mathcal{A}^{\text{privacy-}b}(1^\lambda)$  for  $b \in \{0, 1\}$ . In the following, we define some oracles that will be used in the experiments.

**SndToHX( $x$ ):** This oracle simulates an honest user who requests an attribute key for  $x$  at  $\tau_{\text{current}}$  from an adversarially controlled authority. It maintains a list  $\text{CL}$ . Let the output of this oracle be  $\text{sk}_x$ . It adds  $x$  to  $\text{CL}$  if  $\text{sk}_x$  is valid.

**Chal $_b(x_0, x_1, M, P, \tau)$ :** This is a challenge oracle that is called only once. It computes  $\Sigma \leftarrow \text{Sign}(\text{sk}_{x_b}, M, P, \text{info}_\tau)$  and outputs  $\Sigma$  if and only if  $x_0, x_1 \in \mathcal{X}$ ,  $M \in \mathcal{M}$  or  $P \in \mathcal{P}$ ,  $\tau \in \mathcal{T}$ ,  $\text{info}_\tau \neq \perp$ , and  $x_b \in \text{CL}$ ,  $P(x_b) = 1$ ,  $\text{IsActive}(x_b, \text{info}_\tau) = 1$  for  $b \in \{0, 1\}$ .

In experiment  $\mathbf{Expt}_{\mathcal{A}}^{\text{privacy-}b}(1^\lambda)$ , the adversary  $\mathcal{A}$  fully controls the authority and enrolls honest users to the system by interacting with the oracle  $\text{SndToHX}$ . We require  $\mathcal{A}$  to output the randomness  $\mathbf{r}$  used for generating the authority's key pair to verify its well-formedness. The adversary is also allowed to introduce fully corrupted users to and remove existing users from the system by updating  $\text{info}$  and  $\text{reg}$  at its will, so long as  $\text{info}$  and  $\text{reg}$  are well-formed.

Note that the two challenge attributes  $x_0, x_1$  are required to be active at the challenge time  $\tau$ . However,  $\mathcal{A}$  could update the system (since it fully controls the authority) by revoking either attribute at an arbitrary time period  $\tau' \neq \tau$ . This does not help  $\mathcal{A}$  to win the experiment, since revocation of  $\text{sk}_{x_0}$  or  $\text{sk}_{x_1}$  at time  $\tau'$  does not affect the validity of signatures generated at challenge time  $\tau$ .

<p>Experiment <math>\mathbf{Expt}_{\mathcal{A}}^{\text{correct}}(1^\lambda)</math>  <math>\text{pp} \leftarrow \text{Setup}_{\text{init}}(1^\lambda)</math>;  <math>(\text{mpk}, \text{msk}) \leftarrow \text{Setup}_{\text{auth}}(\text{pp})</math>.  <math>\text{HL} \leftarrow \emptyset</math>.  <math>(x, M, P, \tau) \leftarrow \mathcal{A}^{\text{AddHX, Update}}(\text{pp}, \text{mpk})</math>.            If <math>x \notin \mathcal{X}</math> or <math>M \notin \mathcal{M}</math> or <math>P \notin \mathcal{P}</math>            or <math>\tau \notin \mathcal{T}</math>, return 0.            If <math>x \notin \text{HL}</math> or <math>\text{info}_\tau = \perp</math> or <math>P(x) = 0</math> or  <math>\text{IsActive}(x, \text{info}_\tau) = 0</math>, return 0.  <math>\Sigma \leftarrow \text{Sign}(\text{sk}_x, M, P, \text{info}_\tau)</math>.            If <math>\text{Verify}(M, P, \text{info}_\tau, \Sigma) = 1</math>, return 0.            Else return 1.</p>	<p>If <math>\text{info}</math> or <math>\text{reg}</math> is not well-formed            at any epoch, return 0.  <math>b' \leftarrow \mathcal{A}^{\text{SndToHX, Chal}_b}(\text{aux})</math>.            Return <math>b'</math>.</p>
<p>Experiment <math>\mathbf{Expt}_{\mathcal{A}}^{\text{privacy-}b}(1^\lambda)</math>  <math>\text{pp} \leftarrow \text{Setup}_{\text{init}}(1^\lambda)</math>;  <math>((\text{mpk}, \mathbf{r}), \text{aux}) \leftarrow \mathcal{A}(\text{pp})</math>; <math>\text{CL} \leftarrow \emptyset</math>.            If <math>\text{mpk}</math> is not well-formed, return 0.</p>	<p>Experiment <math>\mathbf{Expt}_{\mathcal{A}}^{\text{unforge}}(1^\lambda)</math>  <math>\text{pp} \leftarrow \text{Setup}_{\text{init}}(1^\lambda)</math>;  <math>(\text{mpk}, \text{msk}) \leftarrow \text{Setup}_{\text{auth}}(\text{pp})</math>.  <math>\text{HL} \leftarrow \emptyset, \text{BL} \leftarrow \emptyset, \text{SL} \leftarrow \emptyset</math>.  <math>(M, P, \tau, \Sigma) \leftarrow \mathcal{A}^{\text{AddHX, RevealX, Sign, Update}}(\text{pp}, \text{mpk})</math>.            If <math>M \notin \mathcal{M}</math> or <math>P \notin \mathcal{P}</math> or <math>\tau \notin \mathcal{T}</math> return 0.            If <math>\text{Verify}(M, P, \text{info}_\tau, \Sigma) = 0</math>, return 0.            If <math>(M, P, \tau, \Sigma) \in \text{SL}</math>, return 0.            If <math>\exists x \in \text{BL}</math> so that <math>P(x) = 1</math> and  <math>\text{IsActive}(x, \text{info}_\tau) = 1</math>, return 0.            Else return 1.</p>

**Fig. 1:** Definitions of correctness, privacy and unforgeability of FDABS.

**Definition 2.** Let the advantage of an adversary  $\mathcal{A}$  against privacy be defined as  $\text{Adv}_{\mathcal{A}}^{\text{privacy}} = |\Pr[\mathbf{Expt}_{\mathcal{A}}^{\text{privacy-}1}(1^\lambda) = 1] - \Pr[\mathbf{Expt}_{\mathcal{A}}^{\text{privacy-}0}(1^\lambda) = 1]|$ . An FD-ABS scheme is perfectly private (statistically private) if for any computationally unbounded adversary  $\mathcal{A}$ , the advantage of  $\mathcal{A}$  is 0 (negligible in  $\lambda$ ).

*Remark 2.* Note that the above definitions on privacy are in the strongest sense. Slightly weaker requirement such as computational privacy, where the adversary is any PPT algorithm, might also be useful for most applications.

**Unforgeability.** This notion extends from the static case carefully to capture the full dynamicity. It protects the verifier from accepting a signature with respect to a policy not satisfied by any attribute in the colluding set. We model this requirement in experiment  $\mathbf{Expt}_{\mathcal{A}}^{\text{unforge}}(1^\lambda)$  which utilizes oracles  $\text{AddHX}$  and  $\text{Update}$ , and the following oracles  $\text{RevealX}$  and  $\text{Sign}$ .

**RevealX( $x$ ):** This oracle allows the adversary to learn an honest attribute key. It maintains a list  $\text{BL}$ . When an attribute  $x$  is queried, it returns the corresponding attribute key  $\text{sk}_x$  and adds  $x$  to  $\text{BL}$  if  $x \in \text{HL}$ , and aborts otherwise.

**Sign( $M, P, \tau$ ):** This oracle allows the adversary to see a signature on any message with respect to any policy and any time epoch of the adversary's choices. It maintains a signature list  $\text{SL}$ . When a tuple  $(M, P, \tau)$  is queried, it returns  $\Sigma \leftarrow \text{Sign}(\text{sk}_x, M, P, \text{info}_\tau)$  with arbitrary  $\text{sk}_x$  such that  $P(x) = 1$  and  $\text{lsActive}(x, \text{info}_\tau) = 1$  and then adds  $(M, P, \tau, \Sigma)$  to the list  $\text{SL}$ . If no such  $x$  is found, it aborts.

**Definition 3.** Let  $\text{Adv}_{\mathcal{A}}^{\text{unforge}} = \Pr[\text{Expt}_{\mathcal{A}}^{\text{unforge}}(1^\lambda) = 1]$  be the advantage of an adversary  $\mathcal{A}$  against unforgeability in an FDABS scheme. The scheme is unforgeable, if for any PPT adversary  $\mathcal{A}$ , the advantage of  $\mathcal{A}$  is negligible in  $\lambda$ .

### 3 Code-Based FDABS for Boolean Circuits

#### 3.1 Preliminaries on Code-Based Cryptographic Tools

**NOTATIONS.** Let  $a, b \in \mathbb{Z}$ . Denote  $[a, b]$  as the set  $\{a, \dots, b\}$ . We simply write  $[b]$  when  $a = 1$ . Let  $\oplus$  denote the bit-wise addition operation modulo 2. If  $S$  is a finite set, then  $x \xleftarrow{\$} S$  means that  $x$  is chosen uniformly at random from  $S$ . Throughout this paper, all vectors are column vectors. When concatenating vectors  $\mathbf{x} \in \{0, 1\}^m$  and  $\mathbf{y} \in \{0, 1\}^k$ , for simplicity, we use  $(\mathbf{x} \parallel \mathbf{y}) \in \{0, 1\}^{m+k}$  instead of  $(\mathbf{x}^\top \parallel \mathbf{y}^\top)^\top$ . The Hamming weight of vector  $\mathbf{x} \in \{0, 1\}^m$  is denoted by  $\text{wt}(\mathbf{x})$ . The Hamming distance between vectors  $\mathbf{x}, \mathbf{y} \in \{0, 1\}^m$  is denoted by  $d_H(\mathbf{x}, \mathbf{y})$ , and is equal to  $\text{wt}(\mathbf{x} \oplus \mathbf{y})$ . Denote by  $\text{B}(n, \omega)$  the set of all binary vectors of length  $n$  with Hamming weight  $\omega$ .

Let  $\mathbb{Z}^+$  be the set consisting of all positive integers. For  $c \in \mathbb{Z}^+$  and  $k$  divisible by  $c$ , define the following.

$\text{Regular}(k, c)$  is the set containing all vectors of the form  $\mathbf{w} = (\mathbf{w}_1 \parallel \dots \parallel \mathbf{w}_{\frac{k}{c}}) \in \{0, 1\}^{2^{c \cdot \frac{k}{c}}}$  that consists of  $\frac{k}{c}$  blocks, each of which is an element of  $\text{B}(2^c, 1)$ .

We call  $\mathbf{w}$  *regular word* if  $\mathbf{w} \in \text{Regular}(k, c)$  for some  $k, c$ .

$\text{2-Regular}(k, c)$  is the set of all  $\mathbf{x} \in \{0, 1\}^{2^{c \cdot \frac{k}{c}}}$  such that there exist regular words  $\mathbf{v}, \mathbf{w} \in \text{Regular}(k, c)$  satisfying  $\mathbf{x} = \mathbf{v} \oplus \mathbf{w}$ . We call  $\mathbf{x}$  a *2-regular word* if  $\mathbf{x} \in \text{2-Regular}(k, c)$  for some  $k, c$ .

$\text{RE} : \{0, 1\}^k \rightarrow \{0, 1\}^{2^{c \cdot \frac{k}{c}}}$  is a regular encoding function mapping  $\mathbf{x} \in \{0, 1\}^k$  to  $\text{RE}(\mathbf{x}) \in \{0, 1\}^{2^{c \cdot \frac{k}{c}}}$ . Let  $\mathbf{x} = (\mathbf{x}_1 \parallel \dots \parallel \mathbf{x}_{\frac{k}{c}})$ , where  $\mathbf{x}_j = [x_{j,1} \dots x_{j,c}]^\top$  for all  $j \in [1, \frac{k}{c}]$ . Then, compute  $t_j = \sum_{i=1}^c 2^{c-i} \cdot x_{j,i}$ . Let  $\text{re}(\mathbf{x}_j) \in \text{B}(2^c, 1)$  whose sole 1 entry is at the  $t_j$ -th position for some  $t_j \in [0, 2^c - 1]$ .  $\text{RE}(\mathbf{x})$  is then defined as  $(\text{re}(\mathbf{x}_1) \parallel \text{re}(\mathbf{x}_2) \parallel \dots \parallel \text{re}(\mathbf{x}_{\frac{k}{c}})) \in \text{Regular}(k, c)$ .

We now recall the  $\text{2-RNSD}_{n,k,c}$  problem, introduced by Augot, Finiasz and Sendrier (AFS) [6]. The problem asks to find low-weight 2-regular codewords

in random binary linear codes, and is closely related to the Small Codeword Problem [67] and binary Shortest Vector Problem [3], with an additional constraint that the solution codeword must be 2-regular.

**Definition 4 (2-RNSD $_{n,k,c}$  Problem).** *Given a uniformly random matrix  $\mathbf{B} \in \mathbb{Z}_2^{n \times m}$ , where  $m = 2^c \cdot k/c$ , find a non-zero vector  $\mathbf{z} \in 2\text{-Regular}(k, c) \subseteq \{0, 1\}^m$  such that  $\mathbf{B} \cdot \mathbf{z} = \mathbf{0}$ .*

The problem is known to be NP-complete in the worst case [6]. In practice, for appropriate choices of  $n$ ,  $k$ ,  $c$ , the best known algorithms require exponential time in the security parameter. See [12] for a comprehensive discussion of known attacks and parameter settings.

Our construction below also employs an efficiently updatable code-based Merkle-tree accumulator and a commitment scheme from [74,73]. The accumulator consists of five algorithms TSetup, TAcc, TWitGen, TVerify, and TUpdate that can efficiently accumulate values, generate witnesses and update a value. The commitment scheme achieves statistical hiding by choosing appropriate parameters. Security of these two schemes relies on the security of 2-RNSD problems. For completeness, we also recall them in Appendix A.

### 3.2 Description of the Scheme

We now present our construction of a code-based ABS for arbitrary Boolean circuits. In the QROM, the scheme satisfies the security requirements defined in Section 2. The scheme makes use of the code-based updatable Merkle-tree accumulator and commitment scheme from [74,73] (also recalled in Appendix A), together with ZK protocols operating within our new framework of Stern from Section 4.1. Initially, the attribute-issuing authority maintains an all-zero Merkle tree. When a user possessing attribute  $\mathbf{x}$  requests an attribute key, the authority computes a commitment  $\mathbf{d} \in \{0, 1\}^n$  to  $\mathbf{x}$  and adds  $\mathbf{d}$  to the tree by associating it with a leaf. When signing a message with respect to policy  $P$  and time  $\tau$ , the signer generates a NIZK argument to prove that (i)  $\mathbf{d}$  is correctly accumulated to the current tree root; (ii)  $\mathbf{d}$  is a valid commitment to  $\mathbf{x}$ ; (iii)  $\mathbf{x}$  satisfies  $P$ . Here the commitment acts as a connecting layer between (i) and (iii). Note that, our NIZK argument is obtained by applying the Unruh transform [86,44] to the refined Stern protocol, resulting in QROM security.

As in [65,73], we set the leaf nodes to all zero values  $\mathbf{0}^n$  initially and change it back to  $\mathbf{0}^n$  when an attribute is revoked. However, we enroll an attribute  $\mathbf{x}$  to the system iff the commitment  $\mathbf{d}$  has odd Hamming weight<sup>8</sup>. Next, when signing a message, the signer additionally proves that  $\mathbf{d}$  has an odd Hamming weight. This is different from [65,73], where an active  $\mathbf{d}$  is set to be non-zero. Our advantage here is that we can prove knowledge of an odd-weight  $\mathbf{d}$  with

<sup>8</sup> For the commitment scheme from [74], a commitment  $\mathbf{d}$  to an arbitrary  $\mathbf{x}$  is statistically close to uniform over  $\mathbb{Z}_2^n$ , thanks to a left-over hash lemma. Hence, it is expected to repeat the process around 2 times to obtain an odd-weight  $\mathbf{d}$ .

cost only  $\mathcal{O}(n)$  bits, while proving knowledge of a non-zero  $\mathbf{d}$  would incur cost  $\mathcal{O}(n \log n)$  bits.

**Setup<sub>init</sub>**( $1^\lambda$ ): Given the security parameter  $1^\lambda$ , this algorithm performs the following steps.

- Let  $L = \text{poly}(\lambda)$  be a positive integer. It then specifies the time space  $\mathcal{T} = \{0, 1, 2, 3, \dots\}$ , the message space  $\mathcal{M} = \{0, 1\}^*$ , the attribute space  $\mathcal{X} = \{0, 1\}^L$ , and the policy family  $\mathcal{P} = \{P : \mathcal{X} \rightarrow \{0, 1\}\}$  that consists of all possible polynomial-size Boolean circuits with  $L$ -bit input.
- Specify an integer  $\ell = \ell(\lambda)$  that determines the maximum number  $N = 2^\ell = \text{poly}(\lambda)$  of potential attributes.
- Choose  $n = \mathcal{O}(\lambda)$ ,  $c = \mathcal{O}(1)$  such that  $c$  divides both  $L$  and  $n$  and set  $m = 2 \cdot 2^c \cdot \frac{n}{c}$ .
- Sample a random matrix  $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_2^{n \times m}$  that specifies a hash function  $h_{\mathbf{B}}$  as in Definition 5.
- Choose  $k \geq n + 2\lambda + \mathcal{O}(1)$  such that  $c$  divides  $k$ . Let  $m_0 = 2^c \cdot L/c$  and  $m_1 = 2^c \cdot k/c$ . Sample  $\mathbf{C}_0 \xleftarrow{\$} \mathbb{Z}_2^{n \times m_0}$  and  $\mathbf{C}_1 \xleftarrow{\$} \mathbb{Z}_2^{n \times m_1}$  that specifies a statistically hiding and computationally binding commitment scheme.
- Let  $\text{COM} : \{0, 1\}^* \times \{0, 1\}^k \rightarrow \{0, 1\}^n$  be the extended commitment scheme as described in Appendix A.1, which will be used in our zero-knowledge argument system (though not explicitly in the description below).
- Pick two secure hash functions  $\mathcal{H}_G : \{0, 1\}^T \rightarrow \{0, 1\}^T$ , for some positive integer  $T$  such that the bit size of each input queried to  $\mathcal{H}_G$  is upper-bounded by  $T$  and  $T$  is super-logarithmic in  $\lambda$ , and  $\mathcal{H}_{\text{FS}} : \{0, 1\}^* \rightarrow \{1, 2, 3\}^\kappa$ , where  $\kappa = \mathcal{O}(\lambda)$ , to be modeled as random oracles in the Unruh transform [86,44].

It then outputs public parameter

$$\text{pp} = \{L, \mathcal{T}, \mathcal{M}, \mathcal{X}, \mathcal{P}, \ell, N, n, c, m, k, m_0, m_1, \mathbf{B}, \mathbf{C}_0, \mathbf{C}_1, \text{COM}, \mathcal{H}_G, \mathcal{H}_{\text{FS}}\}.$$

**Setup<sub>auth</sub>**( $\text{pp}$ ): This algorithm is run by the attribute-issuing authority. On input parameter  $\text{pp}$ , it runs  $(\text{mpk}, \text{msk}) \leftarrow \text{AuthGen}(\text{pp})^9$ . In addition, it initializes the following.

- A registration table  $\mathbf{reg} := (\mathbf{reg}[0], \dots, \mathbf{reg}[N-1])$  so that  $\mathbf{reg}[i][1] = \mathbf{0}^n$ ,  $\mathbf{reg}[i][2] = -1$ , and  $\mathbf{reg}[i][3] = -1$  for all  $i \in [0, N-1]$ . Looking ahead,  $\mathbf{reg}[i][1]$  will store a (non-zero) commitment of an attribute while  $\mathbf{reg}[i][2]$  and  $\mathbf{reg}[i][3]$  represent the epochs an attribute is enrolled in and removed from the system, respectively.
- A Merkle tree  $\mathcal{MT}$  built on top of  $\mathbf{reg}[0][1], \mathbf{reg}[1][1], \dots, \mathbf{reg}[N-1][1]$ . We remark that this  $\mathcal{MT}$  is all-zero at this stage. However, it will be eventually updated either when an attribute is enrolled in or revoked from the system.

<sup>9</sup> Here, we assume the existence of a functionality  $\text{AuthGen}$ , which enables an attribute-issuing authority's key pair to be derived as  $(\text{mpk}, \text{msk}) \leftarrow \text{AuthGen}(1^\lambda)$ . In practice, the authority may use an ordinary signature with verifying-signing key pair  $(\text{mpk}, \text{msk})$  to authenticate the updated system information.

- A counter of enrolled attributes  $j := 0$ , initial time epoch  $\tau = 0$ , and initial system information  $\text{info}_0 = \emptyset$ .

The authority will then publish public key  $\text{mpk}$  and broadcast  $\text{reg}$  and  $\text{info}_0$  while keeping  $\mathcal{MT}$  and  $j$  for itself. We assume that both  $\text{reg}$  and  $\text{info}$  are visible to everyone but only editable by a party who owns  $\text{msk}$ . It is further required that one can efficiently verify the well-formedness of  $\text{reg}$  and  $\text{info}$ .

**AttrGen( $\text{msk}, \mathbf{x}, \text{info}_{\tau_{\text{current}}}, \text{reg}$ ):** When a user requests an attribute key for his provided attribute  $\mathbf{x} \in \{0, 1\}^L$  at current epoch  $\tau_{\text{current}}$ , the authority executes this algorithm and proceeds as follows.

1. Issue an identifier for this attribute  $\mathbf{x}$  as the binary representation of  $j$ , denoted as  $\text{bin}(j) \in \{0, 1\}^\ell$ .
2. Sample randomness  $\mathbf{r} \xleftarrow{\$} \{0, 1\}^k$  and compute a commitment of  $\mathbf{x}$  as  $\mathbf{d} = \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}) \oplus \mathbf{C}_1 \cdot \text{RE}(\mathbf{r})$ . Repeat the process until the weight of  $\mathbf{d}$  is odd. Return  $\text{sk}_{\mathbf{x}} = (\mathbf{x}, \mathbf{r}, \text{bin}(j))$  to the user. From now on, we write  $\text{sk}_{\mathbf{x}_j} = (\mathbf{x}_j, \mathbf{r}_j, \text{bin}(j))$  to distinguish signing keys of different attributes.
3. Update  $\mathcal{MT}$  by running the algorithm  $\text{TUpdate}_{\mathbf{B}}(\text{bin}(j), \mathbf{d})$ , register the attribute to  $\text{reg}$  as  $\text{reg}[j][1] = \mathbf{d}$ ,  $\text{reg}[j][2] = \tau_{\text{current}}$ , and increase the counter  $j$  to  $j + 1$ .

**Update( $\text{msk}, \mathcal{S}, \text{info}_{\tau_{\text{current}}}, \text{reg}$ ):** This algorithm is run by the authority to update the system and advance the epoch. Let  $\mathcal{S} = \{\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_r}\}$  contain attributes to be revoked. If there exists  $t \in [1, r]$  so that  $\text{lsActive}(\mathbf{x}_{i_t}, \text{info}_{\tau_{\text{current}}}) = 0$ , this algorithm aborts. Otherwise it performs the following steps.

1. For each  $t \in [1, r]$ , run  $\text{TUpdate}_{\mathbf{B}}(\text{bin}(i_t), \mathbf{0}^n)$  to update  $\mathcal{MT}$  and set  $\text{reg}[i_t][3] = \tau_{\text{new}}$ .
2. Note that all the zero leaves in updated  $\mathcal{MT}$  are associated with either revoked attributes or potential attributes that have not registered to the system yet. In other words, only active attributes have their odd-weight commitments, denoted as  $\{\mathbf{d}_j\}$ , accumulated in the root  $\mathbf{u}_{\tau_{\text{new}}}$  of the updated tree.

For each  $j$ , let  $w^{(j)} \in \mathbb{Z}_2^\ell \times (\mathbb{Z}_2^n)^\ell$  be the witness for the fact that  $\mathbf{d}_j$  is accumulated in  $\mathbf{u}_{\tau_{\text{new}}}$ . (This can be obtained by running algorithm  $\text{TWitGen}_{\mathbf{B}}$  as described in Appendix A.2). The authority then announces the updated system information as

$$\text{info}_{\tau_{\text{new}}} = (\mathbf{u}_{\tau_{\text{new}}}, \{w^{(j)}\}_j).$$

We remark that unnecessary for a signer or a verifier to download  $\text{info}_{\tau_{\text{new}}}$  as a whole. In fact, as we describe below, a signer with an active attribute only needs to download its corresponding witness  $w^{(j)}$  of  $\mathcal{O}(\lambda \cdot \ell)$  bits once so as to sign messages at time  $\tau_{\text{new}}$ . Meanwhile, it suffices for a verifier to download  $\mathbf{u}_{\tau_{\text{new}}}$  of  $\mathcal{O}(\lambda)$  bits to verify all signatures associated with  $\tau_{\text{new}}$ .

**Sign( $\text{sk}_{\mathbf{x}_j}, M, P, \text{info}_{\tau}$ ):** This algorithm is run by a user possessing an attribute key  $\text{sk}_{\mathbf{x}_j} = (\mathbf{x}_j, \mathbf{r}_j, \text{bin}(j))$  who wishes to sign a message with respect to a policy  $P$ . It aborts if  $P(\mathbf{x}_j) = 0$  or  $\text{info}_{\tau}$  does not include a witness containing  $\text{bin}(j)$ . Otherwise, it proceeds as below.

1. Download  $\mathbf{u}_\tau$  and the witness  $w^{(j)} = (\text{bin}(j), (\mathbf{w}_\ell, \dots, \mathbf{w}_1))$  from  $\text{info}_\tau$ .
2. Generate a proof to show the possession of tuple

$$\xi = (\mathbf{d}_j, \mathbf{x}_j, \mathbf{r}_j, \text{bin}(j), \mathbf{w}_\ell, \dots, \mathbf{w}_1) \quad (1)$$

such that

- (a)  $\mathbf{d}_j$  is correctly accumulated in the root  $\mathbf{u}_\tau$ , i.e.,

$$\text{TVerify}_{\mathbf{B}}(\mathbf{u}_\tau, \mathbf{d}_j, \text{bin}(j), (\mathbf{w}_\ell, \dots, \mathbf{w}_1)) = 1.$$

- (b)  $\mathbf{d}_j$  is an odd-weight commitment of  $\mathbf{x}_j$ , i.e.,

$$\mathbf{d}_j = \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}_j) \oplus \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}_j) \text{ and } wt(\mathbf{d}_j) = 1 \pmod{2}.$$

- (c) The attribute  $\mathbf{x}_j$  satisfies the claimed policy  $P$ . In other words,  $P(\mathbf{x}_j) = 1$ .

To this end, we run the Stern-like protocol in Section 4.3. It is repeated  $\kappa$  times to achieve negligible soundness error and made non-interactive via Unruh transform. The resultant NIZK proof is

$$\Pi = (\{\text{CMT}_i\}_{i=1}^\kappa, \{\overline{\text{RSP}}_{i,j}\}_{i \in [1,\kappa], j \in \{1,2,3\} \setminus \{\text{ch}_i\}}, \text{CH}, \{\text{RSP}_{i,\text{ch}_i}\}_{i=1}^\kappa)$$

where, for all  $i \in [1, \kappa]$  and all  $j \in \{1, 2, 3\}$ ,  $\text{RSP}_{i,j}$  is a response with respect to  $\text{CMT}_i$  and challenge  $j$ ,  $\overline{\text{RSP}}_{i,j} = \mathcal{H}_G(\text{RSP}_{i,j})$ , and

$$\begin{aligned} \text{CH} &= [\text{ch}_1 | \dots | \text{ch}_\kappa]^\top \\ &:= \mathcal{H}_{\text{FS}}(\{\text{CMT}_i\}_{i=1}^\kappa, \{\overline{\text{RSP}}_{i,j}\}_{i \in [1,\kappa], j \in \{1,2,3\}}, M, P, \tau, \mathbf{u}_\tau, \mathbf{B}, \mathbf{C}_0, \mathbf{C}_1) \\ &\in \{1, 2, 3\}^\kappa. \end{aligned}$$

Notice that  $\overline{\text{RSP}}_{i,\text{ch}_i}$ , for all  $i \in [1, \kappa]$ , are excluded from  $\Pi$  since it can be recovered by invoking  $\mathcal{H}_G(\text{RSP}_{i,\text{ch}_i})$ .

3. Return signature as  $\Sigma = \Pi$ .

**Verify**( $M, P, \text{info}_\tau, \Sigma$ ): This algorithm checks the validity of the message signature pair  $(M, \Sigma)$  with respect to the policy  $P$  and time epoch  $\tau$ . It parses  $\Sigma = (\{\text{CMT}_i\}_{i=1}^\kappa, \{\overline{\text{RSP}}_{i,j}\}_{i \in [1,\kappa], j \in \{1,2,3\} \setminus \{\text{ch}_i\}}, \text{CH}, \{\text{RSP}_{i,\text{ch}_i}\}_{i=1}^\kappa)$  and performs the following steps.

1. Download  $\mathbf{u}_\tau$  from  $\text{info}_\tau$ .
2. Compute  $\overline{\text{RSP}}_{i,\text{ch}_i} := \mathcal{H}_G(\text{RSP}_{i,\text{ch}_i})$  for all  $i \in [1, \kappa]$ .
3. If  $\text{CH} \neq \mathcal{H}_{\text{FS}}(\{\text{CMT}_i\}_{i=1}^\kappa, \{\overline{\text{RSP}}_{i,j}\}_{i \in [1,\kappa], j \in \{1,2,3\}}, M, P, \tau, \mathbf{u}_\tau, \mathbf{B}, \mathbf{C}_0, \mathbf{C}_1)$ , return 0.
4. Parse  $\text{CH} = [\text{ch}_1 | \dots | \text{ch}_\kappa]^\top$ .
5. For  $i \in [1, \kappa]$ , verify the validity of  $\text{RSP}_{i,\text{ch}_i}$  with respect to commitment  $\text{CMT}_i$  and challenge value  $\text{ch}_i$ . If any of the verifications does not hold, return 0. Else return 1.

### 3.3 Analysis of the Scheme

**Efficiency.** The efficiency of our construction is summarized as follows.

- The public parameter includes several matrices and has bit size  $\mathcal{O}(\lambda^2 + \lambda \cdot L)$ .
- The attribute key  $\text{sk}_{\mathbf{x}}$  has bit size  $\mathcal{O}(\lambda + L)$ .
- At each epoch, the signer downloads data of bit size  $\mathcal{O}(\lambda \cdot \ell) = \mathcal{O}(\lambda \cdot \log \lambda)$  while the verifier downloads data of bit size  $\mathcal{O}(\lambda)$ .
- The bit-size of signature  $\Sigma$  is  $\mathcal{O}(\zeta + T) \cdot \kappa = \mathcal{O}(L + |P| + \lambda \cdot \log \lambda + T) \cdot \mathcal{O}(\lambda)$ , where  $T$  is the output size of  $\mathcal{H}_G$ ,  $\zeta$  is the average communication cost of the protocol in Section 4.3 and  $|P|$  is the size of a circuit representing  $P$ .

**Correctness.** We show that the construction is correct with probability 1. In fact, experiment  $\text{Expt}_{\mathcal{A}}^{\text{correct}}(1^\lambda)$  outputs 1, i.e., the adversary breaks correctness, iff  $\mathcal{A}$  outputs a tuple  $(\mathbf{x}, M, P, \tau)$  with  $P(\mathbf{x}) = 1$  and  $\text{IsActive}(\mathbf{x}, \text{info}_\tau) = 1$  such that an honestly generated signature  $\Sigma$  is invalid. However, due to perfect correctness of the underlying NIZK protocol, the signature  $\Sigma$  is always valid.

**Security.** In Theorem 1, we prove that our construction satisfies the security requirements proposed in Section 2.2, based on the hardness of the 2-RNSD problems associated with parameters  $(n, 2n, c)$  and  $(n, L + k, c)$ .

**Theorem 1.** *In the QROM, the described FDABS scheme satisfies statistical privacy and unforgeability assuming the hardness of 2-RNSD problems.*

The proof of Theorem 1 relies on the following facts and is deferred to Appendix C.

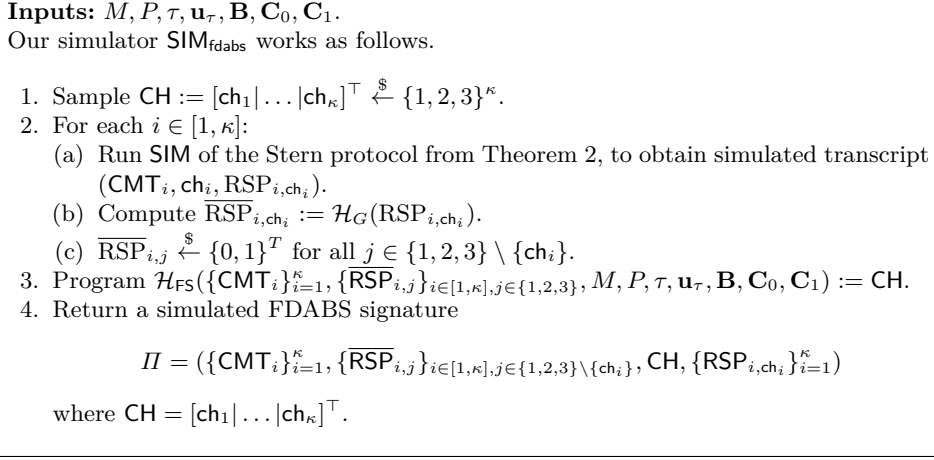
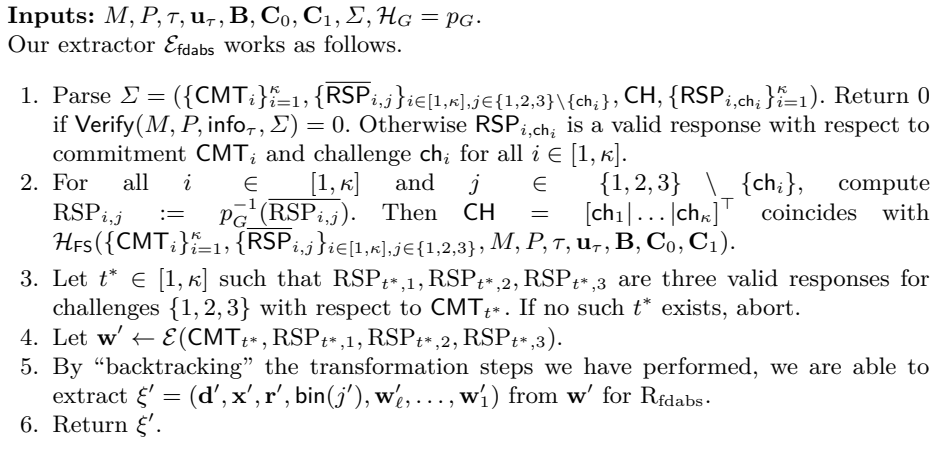
1. The employed Stern-like protocol described in Section 4.3 is statistical ZK and sound, based on the security of the underlying commitment COM.
2. The Merkle tree accumulator is secure as given in Lemma 3, which relies on the hardness of the 2-RNSD $_{n,2n,c}$  problem.
3. The commitment scheme used to commit the attributes is statistically hiding and computationally binding, which depends on the hardness of the 2-RNSD $_{n,L+k,c}$  problem.

We now present how to simulate an FDABS signature, without using a witness, that is statistically indistinguishable from a real one by programming the hash function  $\mathcal{H}_{\text{FS}}$ . Then we show how to extract a valid witness of form (1) from a valid signature by programming  $\mathcal{H}_G$ . These are essential in proving Theorem 1.

**Simulating an FDABS signature.** Let  $M, P, \tau, \mathbf{u}_\tau, \mathbf{B}, \mathbf{C}_0, \mathbf{C}_1$  be the public input involving a signing process and SIM is the simulator of the Stern protocol from Theorem 2. The simulator  $\text{SIM}_{\text{fdabs}}$  is described in Figure 2.

It is straightforward to verify that the above simulated signature is statistically indistinguishable from a real one, assuming the special honest-verifier ZK property of our Stern protocol in Figure 4.

**Extracting a witness from an FDABS signature.** To be able to extract a witness from an FDABS message-signature pair  $(M, \Sigma)$  with respect to a policy  $P$ , we follow [86,44] to simulate random oracle  $\mathcal{H}_G$  as a polynomial  $p_G$  over

**Fig. 2:** Simulator  $\text{SIM}_{\text{fdabs}}$  of FDABS.**Fig. 3:** Extractor  $\mathcal{E}_{\text{fdabs}}$  of FDABS.

$\text{GF}(2^T)$  of a sufficiently large degree, in particular, degree at least  $2q_G - 1$ , such that it is perfectly indistinguishable from a random function. Here  $q_G$  is an upper bound on the number of queries from all possible parties to  $\mathcal{H}_G$ , and  $T$  should be super-logarithmic in  $\lambda$ . From previous results [10], the inverse function  $p_G^{-1}$  can be efficiently computed.

Let  $M, P, \tau, \mathbf{u}_\tau, \mathbf{B}, \mathbf{C}_0, \mathbf{C}_1, \Sigma$  be public input and  $\mathcal{E}$  be the extractor of the protocol from Theorem 2. Our extractor  $\mathcal{E}_{\text{fdabs}}$  is formally described in Figure 3.

Assuming the online extractability of the Unruh transform [86,44], and the special soundness of our Stern protocol in Figure 4, the above extractor  $\mathcal{E}_{\text{fdabs}}$  outputs  $\xi'$  such that all the conditions specified in  $\text{Sign}$  algorithm from Section 3.2. In particular, the existence of  $t^*$  at Step 3 is guaranteed by the online extractability of the Unruh transform.

## 4 Supporting Zero-Knowledge Protocols

This section provides the supporting ZK layer for the FDABS scheme described in Section 3. We first present our refined abstraction of Stern’s protocol in Section 4.1. Then, in Section 4.2, we discuss our enhancement of previous Stern-like techniques in light of the refined abstraction. Then, the ZK protocol used in the signing algorithm of the proposed FDABS is presented in Section 4.3, as a special instance of the abstraction.

### 4.1 A Refined Abstraction of Stern’s Protocol

Here, we present our refined framework for Stern-like protocols. As discussed in Section 1, our approach significantly departs from Stern’s original work [81,82] and Libert et al.’s abstraction [56], in two fundamental aspects: we do not require that additive masking vectors to be uniformly random in the whole space, and we do not even assume that the functions applied to witness vectors are random permutations of coordinates.

Let  $D \in \mathbb{Z}^+$  and  $\text{VALID}, \mathcal{R}, \mathcal{Z}$  be subsets of  $\{0, 1\}^D$ . Let  $\mathcal{S}$  be a finite set, and  $F, F', F''$  be functions with domains/ranges defined as follows:

$$F : \mathcal{S} \times \{0, 1\}^D \rightarrow \{0, 1\}^D; \quad F' : \mathcal{S} \times \mathcal{R} \rightarrow \mathcal{R}; \quad F'' : \mathcal{S} \times \mathcal{Z} \rightarrow \mathcal{Z}.$$

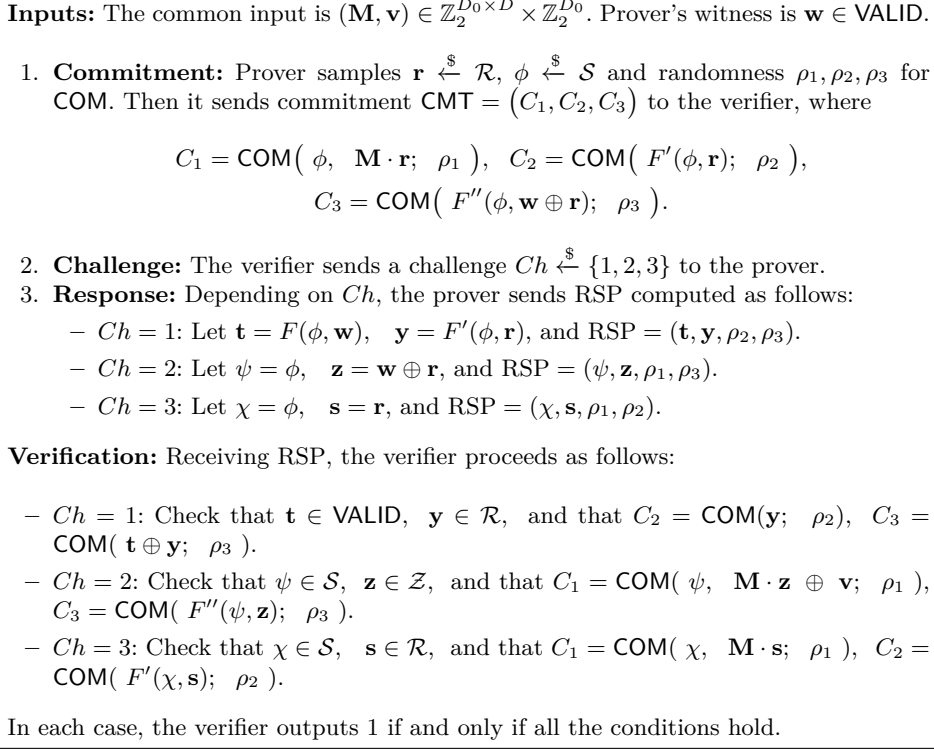
We assume  $\text{VALID}, \mathcal{R}, \mathcal{Z}, \mathcal{S}$ , and  $F, F', F''$  satisfy the following 4 conditions.

- (1) “Homomorphism”: For all  $(\phi, \mathbf{r}, \mathbf{z}) \in \mathcal{S} \times \mathcal{R} \times \mathcal{Z}$ ,  $F'(\phi, \mathbf{r}) \oplus F''(\phi, \mathbf{z}) = F(\phi, \mathbf{r} \oplus \mathbf{z})$ .
- (2) “Closure of  $F$  w.r.t.  $\text{VALID}$ ”: For all  $(\phi, \mathbf{w}) \in \mathcal{S} \times \{0, 1\}^D$ ,  $\mathbf{w} \in \text{VALID} \iff F(\phi, \mathbf{w}) \in \text{VALID}$ .
- (3) For all  $\mathbf{w} \in \text{VALID}$ , the distributions of  $\{\mathbf{t} = F(\phi, \mathbf{w}) \mid \phi \xleftarrow{\$} \mathcal{S}\}$  and  $\{\mathbf{t} \xleftarrow{\$} \text{VALID}\}$  are identical.
- (4) For all  $\mathbf{w} \in \text{VALID}$ , the distributions of  $\{\mathbf{z} = \mathbf{w} \oplus \mathbf{r} \mid \mathbf{r} \xleftarrow{\$} \mathcal{R}\}$  and  $\{\mathbf{z} \xleftarrow{\$} \mathcal{Z}\}$  are identical.

Let  $D_0 \in \mathbb{Z}^+$  such that  $D_0 \leq D$ . We aim to construct a  $\Sigma$ -protocol for the following abstract relation:

$$\mathbf{R}_{\text{abstract}} = \{((\mathbf{M}, \mathbf{v}), \mathbf{w}) \in (\mathbb{Z}_2^{D_0 \times D} \times \mathbb{Z}_2^{D_0}) \times \text{VALID} : \mathbf{M} \cdot \mathbf{w} = \mathbf{v}\}.$$

The interaction between prover  $\mathcal{P}$  and verifier  $\mathcal{V}$  is described in Figure 4. The protocol employs an auxiliary string commitment scheme  $\text{COM}$  that is statistically hiding and computationally binding, and that has commitment size  $n$  bits and randomness size  $r$  bits. (Such a commitment scheme can be obtained based on codes, see [74, Section 3.1] and Appendix A.1.)



**Fig. 4:** A  $\Sigma$ -protocol for the relation  $R_{\text{abstract}}$ .

**Theorem 2.** *Assume that the auxiliary string commitment scheme COM is statistically hiding and computationally binding. Then the protocol in Figure 4 is a  $\Sigma$ -protocol for the relation  $R_{\text{abstract}}$  with perfect completeness and average communication cost  $\zeta$  bits, where*

$$\zeta = 3n + 2r + \frac{\log(|\text{VALID}|) + 2 \log(|\mathcal{R}|) + \log(|\mathcal{Z}|) + 2 \log(|\mathcal{S}|)}{3}. \quad (2)$$

*In particular, the protocol satisfies*

- **Special honest-verifier ZK.** *There exists a PPT simulator that, on input  $(\mathbf{M}, \mathbf{v})$  and  $Ch \in \{1, 2, 3\}$ , outputs an accepted transcript statistically close to that produced by the real prover.*
- **Special soundness.** *There exists a PPT extractor  $\mathcal{E}$  that, on input a commitment CMT and 3 valid responses  $(\text{RSP}_1, \text{RSP}_2, \text{RSP}_3)$  to all 3 possible values of  $Ch$ , outputs  $\mathbf{w}' \in \text{VALID}$  such that  $\mathbf{M} \cdot \mathbf{w}' = \mathbf{v}$ .*

*Proof.* The proof uses the conditions (1)-(4) specified above, as well as the statistical hiding and computational binding properties of COM.

**Completeness.** An honest prover with witness  $\mathbf{w} \in \text{VALID}$  such that  $\mathbf{M} \cdot \mathbf{w} = \mathbf{v}$  will always get accepted by the verifier. Apart from the checks for well-formedness of  $C_1, C_2, C_3$ , it suffices to note the following points.

- **Case  $Ch = 1$ .** We have  $\mathbf{t} = F(\phi, \mathbf{w}) \in \text{VALID}$  by condition (2), and  $\mathbf{y} = F'(\phi, \mathbf{r}) \in \mathcal{R}$ , by the definition of  $F'$ . Furthermore, by condition (4), we know that  $\mathbf{z} = \mathbf{w} \oplus \mathbf{r} \in \mathcal{Z}$ , and then, by condition (1), we have

$$\mathbf{y} \oplus F''(\phi, \mathbf{w} \oplus \mathbf{r}) = F'(\phi, \mathbf{r}) \oplus F''(\phi, \mathbf{z}) = F(\phi, \mathbf{r} \oplus \mathbf{z}) = F(\phi, \mathbf{w}) = \mathbf{t},$$

which implies that  $F''(\phi, \mathbf{w} \oplus \mathbf{r}) = \mathbf{t} \oplus \mathbf{y}$ .

- **Case  $Ch = 2$ .** We have  $\mathbf{z} = \mathbf{w} \oplus \mathbf{r} \in \mathcal{Z}$ . Since  $\mathbf{v} = \mathbf{M} \cdot \mathbf{w}$ , it holds that

$$\mathbf{M} \cdot \mathbf{z} \oplus \mathbf{v} = \mathbf{M} \cdot (\mathbf{w} \oplus \mathbf{r}) \oplus \mathbf{v} = \mathbf{M} \cdot \mathbf{w} \oplus \mathbf{M} \cdot \mathbf{r} \oplus \mathbf{v} = \mathbf{M} \cdot \mathbf{r}.$$

**Honest-verifier ZK.** We construct a simulator SIM that, on input  $(\mathbf{M}, \mathbf{v})$  and  $Ch \in \{1, 2, 3\}$ , returns a transcript statistically close to the real one produced by an honest prover. Depending on the value of  $Ch$ , SIM proceeds as follows.

- **Case  $Ch = 1$ .** SIM samples  $\mathbf{t} \xleftarrow{\$} \text{VALID}$ ,  $\phi \xleftarrow{\$} \mathcal{S}$ ,  $\mathbf{r} \xleftarrow{\$} \mathcal{R}$  and randomness  $\rho_1, \rho_2, \rho_3$  for COM. Then it computes  $\mathbf{y} = F'(\phi, \mathbf{r})$  and commitment  $\text{CMT} = (C'_1, C'_2, C'_3)$  where  $C'_1 = \text{COM}(\mathbf{0}; \rho_1)$ ,  $C'_2 = \text{COM}(\mathbf{y}; \rho_2)$  and  $C'_3 = \text{COM}(\mathbf{t} \oplus \mathbf{y}; \rho_3)$ . It defines  $\text{RSP} = (\mathbf{t}, \mathbf{y}, \rho_2, \rho_3)$ . By condition (3), the distribution of  $\mathbf{t}$  is identical to that of the real transcript. Also,  $\mathbf{y}$  is computed in the same way as in the real transcript.
- **Case  $Ch = 2$ .** SIM samples  $\psi \xleftarrow{\$} \mathcal{S}$ ,  $\mathbf{z} \xleftarrow{\$} \mathcal{Z}$  and randomness  $\rho_1, \rho_2, \rho_3$  for COM. Then it computes commitment  $\text{CMT} = (C'_1, C'_2, C'_3)$  where  $C'_1 = \text{COM}(\psi, \mathbf{M} \cdot \mathbf{z} \oplus \mathbf{v}; \rho_2)$ ,  $C'_2 = \text{COM}(\mathbf{0}; \rho_2)$ ,  $C'_3 = \text{COM}(F''(\psi, \mathbf{z}); \rho_3)$ , and response  $\text{RSP} = (\psi, \mathbf{z}, \rho_1, \rho_3)$ . Here, note that the simulated  $\mathbf{z}$  and the one in real transcript are both uniformly random over  $\mathcal{Z}$ , by condition (4).
- **Case  $Ch = 3$ .** SIM samples  $\psi \xleftarrow{\$} \mathcal{S}$ ,  $\mathbf{s} \xleftarrow{\$} \mathcal{R}$  and randomness  $\rho_1, \rho_2, \rho_3$  for COM. Then it computes commitment  $\text{CMT} = (C'_1, C'_2, C'_3)$  where  $C'_1 = \text{COM}(\chi, \mathbf{M} \cdot \mathbf{s}; \rho_1)$ ,  $C'_2 = \text{COM}(F'(\chi, \mathbf{s}); \rho_2)$  and  $C'_3 = \text{COM}(\mathbf{0}; \rho_3)$ , and response  $\text{RSP} = (\chi, \mathbf{s}, \rho_1, \rho_2)$ .

In every case, SIM outputs the simulated transcript  $(\text{CMT}, Ch, \text{RSP})$ . It then follows from the statistical hiding property of COM that the distributions of simulated and real transcripts are statistically close.

**Special Soundness.** Let  $\text{RSP}_1 = (\mathbf{t}, \mathbf{y}, \rho_2, \rho_3)$ ,  $\text{RSP}_2 = (\psi, \mathbf{z}, \rho_1, \rho_3)$  and  $\text{RSP}_3 = (\chi, \mathbf{s}, \rho_1, \rho_2)$  be the 3 valid responses to the same commitment  $\text{CMT} = (C_1, C_2, C_3)$  with respect to all 3 values of  $Ch$ . Then, the following conditions hold.

$$\begin{cases} \mathbf{t} \in \text{VALID}, \mathbf{y} \in \mathcal{R}, \mathbf{z} \in \mathcal{Z}, \mathbf{s} \in \mathcal{R}, \psi \in \mathcal{S}, \chi \in \mathcal{S}, \\ C_1 = \text{COM}(\psi, \mathbf{M} \cdot \mathbf{z} \oplus \mathbf{v}; \rho_1) = \text{COM}(\chi, \mathbf{M} \cdot \mathbf{s}; \rho_1), \\ C_2 = \text{COM}(\mathbf{y}; \rho_2) = \text{COM}(F'(\chi, \mathbf{s}); \rho_2), \\ C_3 = \text{COM}(\mathbf{t} \oplus \mathbf{y}; \rho_3) = \text{COM}(F''(\psi, \mathbf{z}); \rho_3). \end{cases}$$

Since COM is computationally binding, we deduce that

$$\mathbf{t} \in \text{VALID}, \quad \psi = \chi, \quad \mathbf{M} \cdot \mathbf{z} \oplus \mathbf{v} = \mathbf{M} \cdot \mathbf{s}, \quad \mathbf{y} = F'(\chi, \mathbf{s}), \quad \mathbf{t} \oplus \mathbf{y} = F''(\psi, \mathbf{z}).$$

By condition (1), we have

$$\mathbf{t} = \mathbf{y} \oplus (\mathbf{t} \oplus \mathbf{y}) = \mathbf{y} \oplus F''(\psi, \mathbf{z}) = F'(\psi, \mathbf{s}) \oplus F''(\psi, \mathbf{z}) = F(\psi, \mathbf{s} \oplus \mathbf{z}).$$

Since  $\mathbf{t} \in \text{VALID}$ , condition (2) implies that  $\mathbf{w}' := \mathbf{s} \oplus \mathbf{z} \in \text{VALID}$ . Moreover, as  $\mathbf{M} \cdot \mathbf{z} \oplus \mathbf{v} = \mathbf{M} \cdot \mathbf{s}$ , we deduce that  $\mathbf{M} \cdot \mathbf{w}' = \mathbf{M} \cdot (\mathbf{s} \oplus \mathbf{z}) = \mathbf{v}$ . In other words, we have  $((\mathbf{M}, \mathbf{v}), \mathbf{w}') \in \mathcal{R}_{\text{abstract}}$ .  $\square$

## 4.2 Stern-like Techniques: Previous Ideas and Our Enhancements

**Techniques for handling arbitrary binary vectors.** To prove knowledge of a vector  $\mathbf{x} \in \{0, 1\}^n$ , define the following extension  $\text{Encode}$  and a function  $F_{\text{bin}}$ .

- For  $\mathbf{x} = [x_1 | \dots | x_n]^\top$ , let  $\text{Encode}(\mathbf{x}) = [\bar{x}_1 | x_1 | \dots | \bar{x}_n | x_n]^\top \in \{0, 1\}^{2n}$ .
- Let  $\mathbf{I}_n^* \in \mathbb{Z}_2^{n \times 2n}$  be an extension of the identity matrix  $\mathbf{I}_n$ , obtained by inserting a zero-column  $\mathbf{0}^n$  right before each column of  $\mathbf{I}_n$ . Form a new matrix  $\mathbf{M} = \mathbf{M}_0 \cdot \mathbf{I}_n^* \in \mathbb{Z}_2^{D_0 \times 2n}$ . Therefore,  $\mathbf{x} = \mathbf{I}_n^* \cdot \text{Encode}(\mathbf{x})$  and  $\mathbf{M}_0 \cdot \mathbf{x} = \mathbf{v}$  is equivalent to  $\mathbf{M} \cdot \text{Encode}(\mathbf{x}) = \mathbf{v}$ .
- For  $\mathbf{b} = [b_1 | \dots | b_n]^\top \in \{0, 1\}^n$  and  $\mathbf{w} = [w_{1,0} | w_{1,1} | \dots | w_{n,0} | w_{n,1}]^\top \in \{0, 1\}^{2n}$ , define a function  $F_{\text{bin}} : \{0, 1\}^n \times \{0, 1\}^{2n} \rightarrow \{0, 1\}^{2n}$  as  $F_{\text{bin}}(\mathbf{b}, \mathbf{w}) = [w_{1,b_1} | w_{1,\bar{b}_1} | \dots | w_{n,b_n} | w_{n,\bar{b}_n}]^\top$ . It can be verified that for all  $\mathbf{b} \in \{0, 1\}^n$  and all  $\mathbf{w}, \mathbf{w}' \in \{0, 1\}^{2n}$ , the following holds:

$$F_{\text{bin}}(\mathbf{b}, \mathbf{w}) \oplus F_{\text{bin}}(\mathbf{b}, \mathbf{w}') = F_{\text{bin}}(\mathbf{b}, \mathbf{w} \oplus \mathbf{w}'). \quad (3)$$

Define  $\text{valid}_{\text{bin}} = \{\mathbf{w} : \exists \mathbf{x} \in \{0, 1\}^n \text{ s.t. } \mathbf{w} = \text{Encode}(\mathbf{x})\}$ . Then,  $\forall \mathbf{x}, \mathbf{b} \in \{0, 1\}^n$ ,

$$\mathbf{w} = \text{Encode}(\mathbf{x}) \in \text{valid}_{\text{bin}} \iff F_{\text{bin}}(\mathbf{b}, \mathbf{w}) = \text{Encode}(\mathbf{x} \oplus \mathbf{b}) \in \text{valid}_{\text{bin}}. \quad (4)$$

In Stern's framework (see e.g. [58]), to prove knowledge of  $\mathbf{x}$ , the prover extends  $\mathbf{x}$  to  $\mathbf{w} \in \text{valid}_{\text{bin}}$  and shows that  $\mathbf{w}$  is indeed from the set  $\text{valid}_{\text{bin}}$  by employing equivalence (4). In addition, if  $\mathbf{b}$  is chosen randomly, it perfectly hides  $\mathbf{w}$  and hence  $\mathbf{x}$ . Furthermore, to prove in ZK that the linear equation holds, the prover samples a masking vector  $\mathbf{r}_w \xleftarrow{\$} \mathcal{R} = \{0, 1\}^{2n}$  and convinces the verifier that  $\mathbf{M} \cdot (\mathbf{w} \oplus \mathbf{r}_w) = \mathbf{M} \cdot \mathbf{r}_w \oplus \mathbf{v}$ . We can see that the number of masking bits is  $2n$ . We now present our refined techniques so that  $n$  bits suffice.

**Our techniques.** Before presenting our techniques, let us define some notions.

- Let  $\mathcal{B}_{\text{odd}}^k = \{\mathbf{y} \in \{0, 1\}^k : wt(\mathbf{y}) = 1 \pmod{2}\}$  and  $\mathcal{B}_{\text{even}}^k = \{\mathbf{y} \in \{0, 1\}^k : wt(\mathbf{y}) = 0 \pmod{2}\}$ .
- For simplicity, denote by  $\mathcal{B} = (\mathcal{B}_{\text{odd}}^k \parallel \dots \parallel \mathcal{B}_{\text{odd}}^k) \subset \{0, 1\}^{k \cdot k_0}$  the set that contains all vectors of the form  $\mathbf{y} = (\mathbf{y}_1 \parallel \dots \parallel \mathbf{y}_{k_0})$ , where each  $\mathbf{y}_i \in \mathcal{B}_{\text{odd}}^k$ .

We aim to reduce the above statement, i.e., proving knowledge of  $\mathbf{w} = \text{Encode}(\mathbf{x})$  so that it satisfies the equation  $\mathbf{M} \cdot \mathbf{w} = \mathbf{v}$ , to an instance of the abstract relation  $\mathcal{R}_{\text{abstract}}$  from Section 4.1. To this end, let  $\mathcal{S}_{\text{bin}} = \{0, 1\}^n$  and

$$\mathcal{R}_{\text{bin}} = (\mathcal{B}_{\text{odd}}^2 \parallel \dots \parallel \mathcal{B}_{\text{odd}}^2) \subset \{0, 1\}^{2 \cdot n}, \quad \mathcal{Z}_{\text{bin}} = (\mathcal{B}_{\text{even}}^2 \parallel \dots \parallel \mathcal{B}_{\text{even}}^2) \subset \{0, 1\}^{2 \cdot n}.$$

$F'_{\text{bin}} : \mathcal{S}_{\text{bin}} \times \mathcal{R}_{\text{bin}} \rightarrow \mathcal{R}_{\text{bin}}$  and  $F''_{\text{bin}} : \mathcal{S}_{\text{bin}} \times \mathcal{Z}_{\text{bin}} \rightarrow \mathcal{Z}_{\text{bin}}$  are defined as  $F'_{\text{bin}}(\mathbf{b}, \mathbf{w}) = F_{\text{bin}}(\mathbf{b}, \mathbf{w})$  and  $F''_{\text{bin}}(\mathbf{b}, \mathbf{w}) = F_{\text{bin}}(\mathbf{b}, \mathbf{w})$ , respectively. Since  $F_{\text{bin}}(\mathbf{b}, \cdot)$  is indeed a permutation, then  $F'_{\text{bin}}(\mathbf{b}, \mathbf{w}) \in \mathcal{R}_{\text{bin}}$  if  $\mathbf{w} \in \mathcal{R}_{\text{bin}}$  and  $F''_{\text{bin}}(\mathbf{b}, \mathbf{w}) \in \mathcal{Z}_{\text{bin}}$  if  $\mathbf{w} \in \mathcal{Z}_{\text{bin}}$ . Thus, these two functions are well defined.

We now demonstrate that the four conditions specified in Section 4.1 are all satisfied. First, “homomorphism” and “closure of  $F$  with respect to  $\text{valid}_{\text{bin}}$ ” are satisfied due to (3) and (4), respectively. Next, observe that  $\mathcal{R}_{\text{bin}} = \text{valid}_{\text{bin}}$  and for any  $\mathbf{w} = \text{Encode}(\mathbf{x}) \in \text{valid}_{\text{bin}}$ , we have<sup>10</sup>

$$\begin{aligned} \{\mathbf{t} = F_{\text{bin}}(\mathbf{b}, \mathbf{w}) \mid \mathbf{b} \stackrel{\$}{\leftarrow} \mathcal{S}_{\text{bin}}\} &\equiv \{\mathbf{t} = \text{Encode}(\mathbf{b} \oplus \mathbf{x}) \mid \mathbf{b} \stackrel{\$}{\leftarrow} \{0, 1\}^n\} \\ &\equiv \{\mathbf{t} = \text{Encode}(\mathbf{b}) \mid \mathbf{b} \stackrel{\$}{\leftarrow} \{0, 1\}^n\} \\ &\equiv \{\mathbf{t} \mid \mathbf{t} \stackrel{\$}{\leftarrow} \text{valid}_{\text{bin}}\}; \\ \{\mathbf{z} = \mathbf{w} \oplus \mathbf{r} \mid \mathbf{r} \stackrel{\$}{\leftarrow} \mathcal{R}_{\text{bin}}\} &\equiv \{\mathbf{z} = \mathbf{w} \oplus (\mathbf{r}_1 \parallel \dots \parallel \mathbf{r}_n) \mid \forall i \in [1, n] : \mathbf{r}_i \stackrel{\$}{\leftarrow} \mathcal{B}_{\text{odd}}^2\} \\ &\equiv \{\mathbf{z} = (\mathbf{z}_1 \parallel \dots \parallel \mathbf{z}_n) \mid \forall i \in [1, n] : \mathbf{z}_i \stackrel{\$}{\leftarrow} \mathcal{B}_{\text{even}}^2\} \\ &\equiv \{\mathbf{z} \mid \mathbf{z} \stackrel{\$}{\leftarrow} \mathcal{Z}_{\text{bin}}\}. \end{aligned}$$

Thus, the remaining two conditions hold. Therefore, we have successfully reduced the above statement to an instance of  $\mathbf{R}_{\text{abstract}}$ . Now the prover can simply run the protocol as described in Figure 4, in which the cost of communicating masking vectors is  $\log(|\mathcal{R}_{\text{bin}}|) = n$  bits, reduced from  $2n$  bits.

**Techniques for handling regular words.** Let  $\mathbf{M} \cdot \text{RE}(\mathbf{x}) = \mathbf{v}$  for  $\mathbf{M} \in \mathbb{Z}_2^{D_0 \times 2^{c \cdot n/c}}$ ,  $\mathbf{x} \in \mathbb{Z}_2^n$  and  $\mathbf{v} \in \mathbb{Z}_2^{D_0}$ . To prove knowledge of a regular word  $\mathbf{w} = \text{RE}(\mathbf{x})$  (as defined in Appendix A), we recall the following notions from [74].

- For  $\mathbf{b} = [b_1 \dots b_c]^\top \in \{0, 1\}^c$ ,  $\mathbf{w} = [w_{0,0,\dots,0} \dots |w_{i_1,i_2,\dots,i_c}| \dots |w_{1,1,\dots,1}]^\top \in \{0, 1\}^{2^c}$ , define a function  $f_{\text{re}} : \{0, 1\}^c \times \{0, 1\}^{2^c} \rightarrow \{0, 1\}^{2^c}$  as  $f_{\text{re}}(\mathbf{b}, \mathbf{w}) = [w'_{0,0,\dots,0} \dots |w'_{i_1,i_2,\dots,i_c}| \dots |w'_{1,1,\dots,1}]^\top$  satisfying

$$w'_{i_1,i_2,\dots,i_c} = w_{i_1 \oplus b_1, i_2 \oplus b_2, \dots, i_c \oplus b_c}$$

for each  $[i_1 \dots i_c]^\top \in \{0, 1\}^c$ . It is verifiable that for any  $\mathbf{x}, \mathbf{b} \in \{0, 1\}^c$ ,

$$\mathbf{w} = \text{re}(\mathbf{x}) \iff f_{\text{re}}(\mathbf{b}, \mathbf{w}) = \text{re}(\mathbf{x} \oplus \mathbf{b}). \quad (5)$$

- For  $\mathbf{b} = (\mathbf{b}_1 \parallel \dots \parallel \mathbf{b}_{n/c}) \in \{0, 1\}^n$  containing  $n/c$  blocks of size  $c$  and for  $\mathbf{w} = (\mathbf{w}_1 \parallel \dots \parallel \mathbf{w}_{n/c}) \in \{0, 1\}^{2^{c \cdot n/c}}$  containing  $n/c$  blocks of size  $2^c$ , define  $F_{\text{re}} : \{0, 1\}^n \times \{0, 1\}^{2^{c \cdot n/c}} \rightarrow \{0, 1\}^{2^{c \cdot n/c}}$  as

$$F_{\text{re}}(\mathbf{b}, \mathbf{w}) = (f_{\text{re}}(\mathbf{b}_1, \mathbf{w}_1) \parallel \dots \parallel f_{\text{re}}(\mathbf{b}_{n/c}, \mathbf{w}_{n/c})).$$

One can check that for all  $\mathbf{b} \in \{0, 1\}^n$ , all  $\mathbf{w}, \mathbf{w}' \in \{0, 1\}^{2^{c \cdot n/c}}$ ,

$$F_{\text{re}}(\mathbf{b}, \mathbf{w}) \oplus F_{\text{re}}(\mathbf{b}, \mathbf{w}') = F_{\text{re}}(\mathbf{b}, \mathbf{w} \oplus \mathbf{w}'). \quad (6)$$

<sup>10</sup> We use the “ $\equiv$ ” sign to represent that two distributions are identical.

Define  $\text{valid}_{\text{re}} = \{\mathbf{w} : \exists \mathbf{x} \in \{0, 1\}^n \text{ s.t. } \mathbf{w} = \text{RE}(\mathbf{x})\}$ . For any  $\mathbf{x}, \mathbf{b} \in \{0, 1\}^n$ , it follows immediately from (5) that the following equivalence holds,

$$\mathbf{w} = \text{RE}(\mathbf{x}) \in \text{valid}_{\text{re}} \iff F_{\text{re}}(\mathbf{b}, \mathbf{w}) = \text{RE}(\mathbf{x} \oplus \mathbf{b}) \in \text{valid}_{\text{re}}. \quad (7)$$

Equivalence (7) is fundamental in Stern’s framework for proving knowledge of a regular word  $\mathbf{w} = \text{RE}(\mathbf{x})$ . The prover first samples a random  $\mathbf{b}$  and shows to the verifier that  $F_{\text{re}}(\mathbf{b}, \mathbf{w}) \in \text{valid}_{\text{re}}$ . Due to the equivalence, the verifier should be convinced that  $\mathbf{w} \in \text{valid}_{\text{re}}$  as well. In addition, the uniformity of  $\mathbf{b}$  perfectly hides  $\mathbf{w}$ . Furthermore, to prove in ZK that the equation holds, the prover randomly chooses a masking vector  $\mathbf{r}_w \xleftarrow{\$} \mathcal{R} = \{0, 1\}^{2^c \cdot n/c}$  and persuades the verifier that  $\mathbf{M} \cdot (\mathbf{w} \oplus \mathbf{r}_w) = \mathbf{M} \cdot \mathbf{r}_w \oplus \mathbf{v}$ .

From the above description, we see that the number of bits to represent  $\mathbf{r}_w$  is  $2^c \cdot n/c$ . Let us now present our refined techniques to reduce the cost of communicating masking vectors.

**Our techniques.** As in the case of handling arbitrary binary vectors, the goal is to reduce the considered statement to an instance of  $\text{R}_{\text{abstract}}$ . Let

$$\begin{aligned} \mathcal{R}_{\text{re},n} &= (\mathcal{B}_{\text{odd}}^{2^c} \parallel \dots \parallel \mathcal{B}_{\text{odd}}^{2^c}) \subset \{0, 1\}^{2^c \cdot n/c}, \\ \mathcal{Z}_{\text{re},n} &= (\mathcal{B}_{\text{even}}^{2^c} \parallel \dots \parallel \mathcal{B}_{\text{even}}^{2^c}) \subset \{0, 1\}^{2^c \cdot n/c}, \quad \mathcal{S}_{\text{re}} = \{0, 1\}^n. \end{aligned}$$

Define  $F'_{\text{re}} : \mathcal{S}_{\text{re}} \times \mathcal{R}_{\text{re}} \rightarrow \mathcal{R}_{\text{re}}$ ,  $F''_{\text{re}} : \mathcal{S}_{\text{re}} \times \mathcal{Z}_{\text{re}} \rightarrow \mathcal{Z}_{\text{re}}$  as  $F'_{\text{re}}(\mathbf{b}, \mathbf{w}) = F_{\text{re}}(\mathbf{b}, \mathbf{w})$  and  $F''_{\text{re}}(\mathbf{b}, \mathbf{w}) = F_{\text{re}}(\mathbf{b}, \mathbf{w})$ , respectively. These functions are well defined because  $F_{\text{re}}$  is a permutation.

We now verify that the requirements in Section 4.1 all hold. Due to (6) and (7), “homomorphism” and “closure of  $F$  with respect to  $\text{valid}_{\text{re}}$ ” are met. Next, let  $\mathbf{x} = (\mathbf{x}_1 \parallel \dots \parallel \mathbf{x}_{n/c}) \in \{0, 1\}^n$  where all  $\mathbf{x}_i$  have the same length. For  $\mathbf{w} = \text{RE}(\mathbf{x}) = (\text{re}(\mathbf{x}_1) \parallel \dots \parallel \text{re}(\mathbf{x}_{n/c})) \in \text{valid}_{\text{re}}$ , we have  $\text{re}(\mathbf{x}_i) \in \mathcal{B}_{\text{odd}}^{2^c}$  and:

$$\begin{aligned} \{\mathbf{t} = F_{\text{re}}(\mathbf{b}, \mathbf{w}) \mid \mathbf{b} \xleftarrow{\$} \mathcal{S}_{\text{re}}\} &\equiv \{\mathbf{t} = \text{RE}(\mathbf{b} \oplus \mathbf{x}) \mid \mathbf{b} \xleftarrow{\$} \{0, 1\}^n\} \\ &\equiv \{\mathbf{t} = \text{RE}(\mathbf{b}) \mid \mathbf{b} \xleftarrow{\$} \{0, 1\}^n\} \\ &\equiv \{\mathbf{t} \mid \mathbf{t} \xleftarrow{\$} \text{valid}_{\text{re}}\}; \\ \{\mathbf{z} = \mathbf{w} \oplus \mathbf{r} \mid \mathbf{r} \xleftarrow{\$} \mathcal{R}_{\text{re},n}\} &\equiv \{\mathbf{z} = (\text{re}(\mathbf{x}_i) \oplus \mathbf{r}_i)_{i=1}^{n/c} \mid \forall i \in [1, n/c] : \mathbf{r}_i \xleftarrow{\$} \mathcal{B}_{\text{odd}}^{2^c}\} \\ &\equiv \{\mathbf{z} = (\mathbf{z}_1 \parallel \dots \parallel \mathbf{z}_{n/c}) \mid \forall i \in [1, n/c] : \mathbf{z}_i \xleftarrow{\$} \mathcal{B}_{\text{even}}^{2^c}\} \\ &\equiv \{\mathbf{z} \mid \mathbf{z} \xleftarrow{\$} \mathcal{Z}_{\text{re},n}\}. \end{aligned}$$

Therefore, the last two conditions are satisfied as well. The prover and verifier now interact as in Figure 4, which incurs masking vectors of length  $(2^c - 1) \cdot n/c$  rather than  $2^c \cdot n/c$  since  $\log(|\mathcal{R}_{\text{re},n}|) = (2^c - 1) \cdot n/c$ .

At the first glance, the size of  $\mathcal{R}_{\text{re},n}$  is not lowered significantly. We, however, remark that this is indeed optimal. Let  $\mathcal{B}^{2^c} \subset \{0, 1\}^{2^c}$  and  $\mathcal{R}' = (\mathcal{B}^{2^c} \parallel \dots \parallel \mathcal{B}^{2^c}) \subset \{0, 1\}^{2^c \cdot n/c}$ . Consider two arbitrary  $\mathbf{w}, \mathbf{w}' \in \text{valid}_{\text{re}}$ , the fourth condition required by  $\text{R}_{\text{abstract}}$  implies that for all  $i \in [1, n/c]$ :

$$\{\mathbf{z}_i = \text{re}(\mathbf{x}_i) \oplus \mathbf{r}_i \mid \mathbf{r}_i \xleftarrow{\$} \mathcal{B}^{2^c}\} \equiv \{\mathbf{z}'_i = \text{re}(\mathbf{x}'_i) \oplus \mathbf{r}'_i \mid \mathbf{r}'_i \xleftarrow{\$} \mathcal{B}^{2^c}\}. \quad (8)$$

Also note that  $\{\text{re}(\mathbf{x}_0) : \mathbf{x}_0 \in \{0, 1\}^c\}$  forms a basis of the vector space  $\{0, 1\}^{2^c}$ . Note that if  $\mathcal{B}^{2^c}$  ever contains a single vector with odd weight, then  $\mathcal{B}^{2^c}$  contains all vectors with odd weight, i.e.,  $\mathcal{B}^{2^c} \supseteq \mathcal{B}_{\text{odd}}^{2^c}$ . Thus, the size of  $\mathcal{R}_{\text{re},n}$  is optimal.

We also remark that the size of  $\mathcal{R}_{\text{re},n}$  is optimal only because we transform the considered statement to an instance of  $\mathbf{R}_{\text{abstract}}$ . It remains open if we can further reduce the size of  $\mathcal{R}_{\text{re},n}$  by considering a different abstract relation.

**Techniques for handling bit multiplication.** Let  $\mathbf{M}_0 \cdot (x_1 \cdot x_2) = \mathbf{v}$  for  $\mathbf{M}_0 \in \mathbb{Z}_2^{D_0 \times 1}$ ,  $x_1, x_2 \in \{0, 1\}$  and  $\mathbf{v} \in \mathbb{Z}_2^{D_0}$ . To prove knowledge of  $x_3 = x_1 \cdot x_2$ , Libert et al. [57] proposed the following notations and techniques.

- For bits  $x_1, x_2, x_3 = x_1 \cdot x_2$ , denote  $\text{ext}(x_1, x_2) = [\bar{x}_1 \cdot \bar{x}_2 \mid \bar{x}_1 \cdot x_2 \mid x_1 \cdot \bar{x}_2 \mid x_1 \cdot x_2]^\top \in \{0, 1\}^4$  as the extension of the bit product  $x_3$ . Then  $x_3 = [0 \mid 0 \mid 0 \mid 1] \cdot \text{ext}(x_1, x_2)$ .
- For  $b = [b_1 \mid b_2]^\top \in \{0, 1\}^2$  and  $\mathbf{w} = [w_{0,0} \mid w_{0,1} \mid w_{1,0} \mid w_{1,1}]^\top \in \{0, 1\}^4$ , define a function  $F_{\text{mult}} : \{0, 1\}^2 \times \{0, 1\}^4 \rightarrow \{0, 1\}^4$  as

$$F_{\text{mult}}(\mathbf{b}, \mathbf{w}) = [w_{b_1, b_2} \mid w_{b_1, \bar{b}_2} \mid w_{\bar{b}_1, b_2} \mid w_{\bar{b}_1, \bar{b}_2}]^\top.$$

Define  $\text{valid}_{\text{mult}} = \{\mathbf{w} : \exists x_1, x_2 \in \{0, 1\} \text{ s.t. } \mathbf{w} = \text{ext}(x_1, x_2)\}$ . Then for any  $\mathbf{x} = [x_1 \mid x_2]^\top \in \{0, 1\}^2$ , and  $\mathbf{b} = [b_1 \mid b_2]^\top \in \{0, 1\}^2$ ,

$$\mathbf{w} = \text{ext}(x_1, x_2) \in \text{valid}_{\text{mult}} \Leftrightarrow F_{\text{mult}}(\mathbf{b}, \mathbf{w}) = \text{ext}(b_1 \oplus x_1, b_2 \oplus x_2) \in \text{valid}_{\text{mult}}. \quad (9)$$

Equivalence (9) is essential in Stern's framework to prove knowledge of  $x_3$  such that  $x_3 = x_1 \cdot x_2$ . In addition, the prover samples a random masking vector  $\mathbf{r} \in \mathcal{R} = \{0, 1\}^4$  and shows to the verifier that  $\mathbf{M} \cdot (\text{ext}(x_1, x_2) \oplus \mathbf{r}) = \mathbf{M} \cdot \mathbf{r} \oplus \mathbf{v}$ , where  $\mathbf{M} = \mathbf{M}_0 \cdot [0 \mid 0 \mid 0 \mid 1] \in \mathbb{Z}_2^{D_0 \times 4}$ .

Here, 4 bits are needed to represent  $\mathbf{r}$ . We will show how to reduce to 3 bits.

**Our techniques.** Let  $\mathcal{R}_{\text{mult}} = \mathcal{B}_{\text{odd}}^4$ ,  $\mathcal{Z}_{\text{mult}} = \mathcal{B}_{\text{even}}^4$ ,  $\mathcal{S}_{\text{mult}} = \mathbb{Z}_2^2$ . Define

$$\begin{aligned} F'_{\text{mult}} : \mathcal{S}_{\text{mult}} \times \mathcal{R}_{\text{mult}} &\rightarrow \mathcal{R}_{\text{mult}} : F'_{\text{mult}}(\mathbf{b}, \mathbf{w}) = F_{\text{mult}}(\mathbf{b}, \mathbf{w}); \\ F''_{\text{mult}} : \mathcal{S}_{\text{mult}} \times \mathcal{Z}_{\text{mult}} &\rightarrow \mathcal{Z}_{\text{mult}} : F''_{\text{mult}}(\mathbf{b}, \mathbf{w}) = F_{\text{mult}}(\mathbf{b}, \mathbf{w}). \end{aligned}$$

These two functions are well defined since  $F_{\text{mult}}$  is actually a permutation. Now we show that the four requirements in Section 4.1 are satisfied. For all  $\mathbf{b} \in \mathcal{S}_{\text{mult}}$ ,  $\mathbf{w} \in \mathcal{R}_{\text{mult}}$  and  $\mathbf{w}' \in \mathcal{Z}_{\text{mult}}$ , it is not hard to see that  $F'_{\text{mult}}(\mathbf{b}, \mathbf{w}) \oplus F''_{\text{mult}}(\mathbf{b}, \mathbf{w}') = F_{\text{mult}}(\mathbf{b}, \mathbf{w} \oplus \mathbf{w}')$ . Next, equivalence (9) shows that  $F_{\text{mult}}$  is closed with respect to  $\text{valid}_{\text{mult}}$ . For all  $\mathbf{w} \in \text{valid}_{\text{mult}}$ , note that  $\{\mathbf{t} = F_{\text{mult}}(\mathbf{b}, \mathbf{w}) : \mathbf{b} \in \mathcal{S}_{\text{mult}}\} = \text{valid}_{\text{mult}}$ . It then follows that the third condition holds. Finally,  $wt(\mathbf{w}) = 1 \pmod 2$  for all  $\mathbf{w} \in \text{valid}_{\text{mult}}$ . Thus, the distributions  $\{\mathbf{z} = \mathbf{w} \oplus \mathbf{r} \mid \mathbf{r} \stackrel{\$}{\leftarrow} \mathcal{R}_{\text{mult}}\}$  and  $\{\mathbf{z} \mid \mathbf{z} \stackrel{\$}{\leftarrow} \mathcal{Z}_{\text{mult}}\}$  are identical for all  $\mathbf{w} \in \text{valid}_{\text{mult}}$ . Therefore, we have successfully reduced the above statement to an instance of  $\mathbf{R}_{\text{abstract}}$ . Therefore, to prove knowledge of  $x_3$  such that  $x_3 = x_1 \cdot x_2$  for  $x_1, x_2 \in \{0, 1\}$  it suffices to sample a masking value  $\mathbf{r}_w$  from  $\mathcal{R}_{\text{mult}}$ . As a result, 3 bits are sufficient to represent  $\mathbf{r}_w$  since  $\log |\mathcal{R}_{\text{mult}}| = 3$ .

### 4.3 Supporting Zero-Knowledge Protocol for Algorithm Sign

This section presents the zero-knowledge protocol invoked by the signer who executes the algorithm `Sign` in Section 3.2. Let  $L, \ell, N, n, c, m, k, m_0, m_1$  as specified in Section 3.2. For completeness, the protocol is summarized below.

- The public input consists of  $P, \mathbf{u}_\tau, \mathbf{B}, \mathbf{C}_0, \mathbf{C}_1$ .
- The secret input consists of  $\xi = (\mathbf{d}_j, \mathbf{x}_j, \mathbf{r}_j, \text{bin}(j), \mathbf{w}_\ell, \dots, \mathbf{w}_1)$ .
- The goal of the prover is to prove in ZK that
  - $\mathbf{d}_j$  is correctly accumulated in the root  $\mathbf{u}_\tau$ , i.e.,

$$\text{TVerify}_{\mathbf{B}}(\mathbf{u}_\tau, \mathbf{d}_j, \text{bin}(j), (\mathbf{w}_\ell, \dots, \mathbf{w}_1)) = 1. \quad (10)$$

- $\mathbf{d}_j$  is an odd-weight commitment of  $\mathbf{x}_j$ , i.e.,

$$\mathbf{d}_j = \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}_j) \oplus \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}_j) \text{ and } wt(\mathbf{d}_j) = 1 \pmod{2}.$$

- The attribute  $\mathbf{x}_j$  satisfies the claimed policy  $P$ , i.e.,  $P(\mathbf{x}_j) = 1$ .

Let  $R_{\text{fdabs}}$  be the corresponding relation. Our strategy is to reduce  $R_{\text{fdabs}}$  to an instance of  $R_{\text{abstract}}$  from Section 4.1. We remark that we will transform the secret input to a secret vector  $\mathbf{w}_{\text{fdabs}} \in \{0, 1\}^{D_{\text{fdabs}}}$  and design a masking vector space  $\mathcal{R}_{\text{fdabs}}$  such that the  $\log(|\mathcal{R}_{\text{fdabs}}|)$  is strictly smaller than  $D_{\text{fdabs}}$ .

**Handling the accumulator layer.** In [74], Nguyen et al. devised a statistical zero-knowledge argument of knowledge that allows a prover  $\mathcal{P}$  to convince a verifier  $\mathcal{V}$  in ZK that  $\mathcal{P}$  knows a value that was honestly accumulated in the root of the Merkle tree. Using the techniques presented in Section 4.2, we are able to obtain a protocol with fewer communication cost. Specifically, proving knowledge of  $\mathbf{d}_j, \text{bin}(j), \mathbf{w}_\ell, \dots, \mathbf{w}_1$  such that (10) holds is the same as proving knowledge of  $\mathbf{w}_{\text{acc}} \in \{0, 1\}^{D_{\text{acc}}}$  such that  $\mathbf{M}_{\text{acc}} \cdot \mathbf{w}_{\text{acc}} = \mathbf{v}_{\text{acc}}$  for  $\mathbf{M}_{\text{acc}} \in \mathbb{Z}_2^{\ell n \times D_{\text{acc}}}$ ,  $\mathbf{v}_{\text{acc}} \in \mathbb{Z}_2^{\ell n}$ ,  $\mathbf{w}_{\text{acc}} \in \{0, 1\}^{D_{\text{acc}}}$ , and  $D_{\text{acc}} = 2\ell m + 2(\ell - 1)n$ . In addition, the sets  $\text{valid}_{\text{acc}}, \mathcal{R}_{\text{acc}}, \mathcal{Z}_{\text{acc}}, \mathcal{S}_{\text{acc}}$ , and the functions  $F_{\text{acc}}, F'_{\text{acc}}, F''_{\text{acc}}$  are given so that the accumulator layer is reduced to an instance of  $R_{\text{abstract}}$ . Particularly,  $\log|\mathcal{R}_{\text{acc}}| = 2\ell m + (\ell - 1)n < D_{\text{acc}}$ . Details are given in Appendix B.1.

**Handling the commitment layer.** We will employ the techniques presented in Section 4.2 to prove knowledge of regular words  $\text{RE}(\mathbf{x}_j)$  and  $\text{RE}(\mathbf{r}_j)$ . We now show how to prove that  $\mathbf{d}_j$  has odd weight. To this end, observe that

$$wt(\mathbf{d}_j) = 1 \pmod{2} \iff \mathbf{1}^{1 \times n} \cdot \mathbf{d}_j = 1 \pmod{2}. \quad (11)$$

Let  $\mathbf{y}_\ell = \text{Encode}(\mathbf{d}_j) \in \mathbb{Z}_2^{2n}$ . We have  $\mathbf{d}_j = \mathbf{I}_n^* \cdot \mathbf{y}_\ell$  and  $\mathbf{M}_1 \cdot \mathbf{y}_\ell = 1$ , where  $\mathbf{M}_1 = \mathbf{1}^{1 \times n} \cdot \mathbf{I}_n^* \in \mathbb{Z}_2^{1 \times 2n}$ . Therefore, the commitment layer is now equivalent to proving knowledge of  $\mathbf{y}_\ell, \text{RE}(\mathbf{x}_j), \text{RE}(\mathbf{r}_j)$  such that

$$\mathbf{M}_1 \cdot \mathbf{y}_\ell = 1 \text{ and } \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}_j) \oplus \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}_j) \oplus \mathbf{I}_n^* \cdot \mathbf{y}_\ell = \mathbf{0}^n. \quad (12)$$

Through some basic algebra, one can form public matrix  $\mathbf{M}_{\text{com}} \in \mathbb{Z}_2^{(n+1) \times D_{\text{com}}}$  and vector  $\mathbf{v}_{\text{com}} \in \mathbb{Z}_2^{n+1}$ , secret  $\mathbf{w}_{\text{com}} = (\mathbf{y}_\ell \parallel \text{RE}(\mathbf{x}_j) \parallel \text{RE}(\mathbf{r}_j)) \in \mathbb{Z}_2^{D_{\text{com}}}$  with  $D_{\text{com}} = 2n + m_0 + m_1$  such that (12) is equivalent to  $\mathbf{M}_{\text{com}} \cdot \mathbf{w}_{\text{com}} = \mathbf{v}_{\text{com}}$ .

**Handling the policy layer.** To prove knowledge of  $\mathbf{x}_j$  so that it satisfies the policy  $P$ , we first convert  $P$  to a Boolean circuit  $C$  entirely represented by NAND gates. Next we show knowledge of each wire value so that the output of  $C$  is 1 and that for each triple  $(x_1, x_2, x_3)$  connected by a NAND gate,  $x_1 \cdot x_2 \oplus x_3 = 1$ .

Note that the above transformation of the policy circuit  $P$  to  $C$  is for presentation purpose. In fact, for a triple  $(x_1, x_2, x_3)$  connected by an AND gate and an OR gate, we have  $x_1 \cdot x_2 \oplus x_3 = 0$  and  $x_1 \cdot x_2 \oplus x_3 \oplus x_1 \oplus x_2 = 0$ , respectively; for  $(x_1, x_2)$  connected by a NOT gate, we have  $x_1 \oplus x_2 = 1$ . Therefore, we can use the same techniques that handle the NAND gates and arbitrary binary vectors to handle  $x_1 \cdot x_2 \oplus x_3$  and the linear terms  $x_1, x_2$ , respectively. Therefore, it will not be an issue even if there is a blowup in  $C$ 's size due to the transformation.

Now, let us present the notations and techniques for handling NAND triples.

- For  $x_1, x_2, x_3 \in \{0, 1\}$ , let

$$X = \text{ENC}(x_1, x_2, x_3) = [\bar{x}_1 \cdot \bar{x}_2 \oplus x_3 \mid \bar{x}_1 \cdot x_2 \oplus x_3 \mid x_1 \cdot \bar{x}_2 \oplus x_3 \mid x_1 \cdot x_2 \oplus x_3]^\top \in \mathbb{Z}_2^4.$$

It is easy to see that  $[0 \mid 0 \mid 0 \mid 1] \cdot X = x_1 \cdot x_2 \oplus x_3$ .

- Define  $\text{valid}_{\text{nand}} = \{X : \exists x_1, x_2, x_3 \in \mathbb{Z}_2 \text{ s.t. } X = \text{ENC}(x_1, x_2, x_3)\}$ . It is worth noting that  $\text{valid}_{\text{nand}} = \mathcal{B}_{\text{odd}}^4$ , i.e., the set that contains all length-4 binary vectors with odd weight. Set  $\mathcal{R}_{\text{nand}} = \mathcal{B}_{\text{odd}}^4$ ,  $\mathcal{Z}_{\text{nand}} = \mathcal{B}_{\text{even}}^4$ ,  $\mathcal{S}_{\text{nand}} = \mathbb{Z}_2^3$ .
- Let  $\mathbf{b} = [b_1 \mid b_2 \mid b_3]^\top \in \mathbb{Z}_2^3$  and  $X = [x_{0,0} \mid x_{0,1} \mid x_{1,0} \mid x_{1,1}]^\top \in \mathbb{Z}_2^4$ , define  $F_{\text{nand}} : \mathcal{S}_{\text{nand}} \times \mathbb{Z}_2^4 \rightarrow \mathbb{Z}_2^4$  as

$$F_{\text{nand}}(\mathbf{b}, X) = [x_{b_1, b_2} \oplus b_3 \mid x_{b_1, \bar{b}_2} \oplus b_3 \mid x_{\bar{b}_1, b_2} \oplus b_3 \mid x_{\bar{b}_1, \bar{b}_2} \oplus b_3]^\top,$$

$$F'_{\text{nand}} : \mathcal{S}_{\text{nand}} \times \mathcal{R}_{\text{nand}} \rightarrow \mathcal{R}_{\text{nand}} \text{ as } F'_{\text{nand}}(\mathbf{b}, X) = [x_{b_1, b_2} \mid x_{b_1, \bar{b}_2} \mid x_{\bar{b}_1, b_2} \mid x_{\bar{b}_1, \bar{b}_2}]^\top,$$

and  $F''_{\text{nand}} : \mathcal{S}_{\text{nand}} \times \mathcal{Z}_{\text{nand}} \rightarrow \mathcal{Z}_{\text{nand}}$  as

$$F''_{\text{nand}}(\mathbf{b}, X) = F_{\text{nand}}(\mathbf{b}, X) = [x_{b_1, b_2} \mid x_{b_1, \bar{b}_2} \mid x_{\bar{b}_1, b_2} \mid x_{\bar{b}_1, \bar{b}_2}]^\top \oplus [b_3 \mid b_3 \mid b_3 \mid b_3]^\top.$$

First, we show that  $F'_{\text{nand}}$  and  $F''_{\text{nand}}$  are well defined. Note that  $F'_{\text{nand}}(\mathbf{b}, \cdot)$  is indeed a permutation, thus  $X \in \mathcal{R}_{\text{nand}}$  implies  $F'_{\text{nand}}(\mathbf{b}, X) \in \mathcal{R}_{\text{nand}}$ . Regarding  $F''_{\text{nand}}$ , it suffices to see that  $[b_3 \mid b_3 \mid b_3 \mid b_3]^\top \in \mathcal{Z}_{\text{nand}}$ .

Next, we will demonstrate that the four conditions specified in Section 4.1 with respect to  $\text{valid}_{\text{nand}}$ ,  $\mathcal{R}_{\text{nand}}$ ,  $\mathcal{Z}_{\text{nand}}$ ,  $\mathcal{S}_{\text{nand}}$  and  $F_{\text{nand}}$ ,  $F'_{\text{nand}}$ ,  $F''_{\text{nand}}$  are satisfied. Looking ahead, this is crucial in reducing  $\text{R}_{\text{fdabs}}$  to an instance of  $\text{R}_{\text{abstract}}$ .

For all  $\mathbf{b} = [b_1 \mid b_2 \mid b_3]^\top \in \mathcal{S}_{\text{nand}}$ , and  $X \in \mathcal{R}_{\text{nand}}$ ,  $X' \in \mathcal{Z}_{\text{nand}}$ ,

$$\begin{aligned} F_{\text{nand}}(\mathbf{b}, X \oplus X') &= [x_{b_1, b_2} \mid x_{b_1, \bar{b}_2} \mid x_{\bar{b}_1, b_2} \mid x_{\bar{b}_1, \bar{b}_2}]^\top \\ &\quad \oplus [x'_{b_1, b_2} \mid x'_{b_1, \bar{b}_2} \mid x'_{\bar{b}_1, b_2} \mid x'_{\bar{b}_1, \bar{b}_2}]^\top \oplus [b_3 \mid b_3 \mid b_3 \mid b_3]^\top \\ &= F'_{\text{nand}}(\mathbf{b}, X) \oplus F''_{\text{nand}}(\mathbf{b}, X'). \end{aligned}$$

Thus, ‘‘homomorphism’’ is satisfied. Also, due to the design of  $F_{\text{nand}}$ , it is not hard to verify that for all  $\mathbf{b} \in \mathcal{S}_{\text{nand}}$ , all  $X \in \mathbb{Z}_2^4$ , the equivalence  $X \in \text{valid}_{\text{nand}} \iff$

$F_{\text{nand}}(\mathbf{b}, X) \in \text{valid}_{\text{nand}}$  holds. Therefore, “closure of  $F_{\text{nand}}$  with respect to  $\text{valid}_{\text{nand}}$ ” is also fulfilled. Finally, observe that for all  $X \in \text{valid}_{\text{nand}}$ ,  $\{F_{\text{nand}}(\mathbf{b}, X) : \mathbf{b} \in \mathcal{S}_{\text{nand}}\} = \text{valid}_{\text{nand}}$ ,  $\{\mathbf{z} = X + \mathbf{r} : \mathbf{r} \in \mathcal{B}_{\text{odd}}^4 = \mathcal{R}_{\text{nand}}\} = \mathcal{B}_{\text{even}}^4 = \mathcal{Z}_{\text{nand}}$ . Therefore, the last two conditions are satisfied as well. Being prepared with the above technique, we are ready to describe the transformations for the policy layer.

Let  $K$  be the number of NAND gates in  $C$ . We distinguish the input wires, the internal wires, and the output wire by  $1, \dots, L, L+1, \dots, L+K$ , where  $1, \dots, L$  are the input wires,  $L+1, \dots, L+K-1$  are the internal wires while  $L+K$  is the output wire. The topology of  $C$  is specified by two publicly known functions  $g, h$  mapping  $\{1, \dots, K\} \rightarrow \{1, \dots, L+K-1\}$ . Given an  $L$ -bit input  $(x_1, \dots, x_L)$ , the assignments to non-input wires in  $C$  are denoted as  $x_{L+1}, \dots, x_{L+K}$  and are as computed  $x_{L+i} = x_{g(i)} \text{ NAND } x_{h(i)}$  for  $i \in [1, K]$ . Equivalently,

$$\begin{cases} x_{g(1)} \cdot x_{h(1)} \oplus x_{L+1} = 1, \\ \dots \\ x_{g(K-1)} \cdot x_{h(K-1)} \oplus x_{L+K-1} = 1, \\ x_{g(K)} \cdot x_{h(K)} \oplus x_{L+K} = 1. \end{cases} \quad (13)$$

Thus, the policy layer is equivalent to proving knowledge of  $x_1, x_2, \dots, x_{L+K-1}$ , and  $x_{L+K} = 1$  such that equations in (13) hold. For every  $i \in [1, K-1]$ , let  $X_i = \text{ENC}(x_{g(i)}, x_{h(i)}, x_{L+i})$  and  $X_K = \text{ext}(x_{g(K)}, x_{h(K)})$ . Denote  $\mathbf{h}_4 = [0 \mid 0 \mid 0 \mid 1] \in \mathbb{Z}_2^{1 \times 4}$ . Then (13) is equivalent to

$$\mathbf{h}_4 \cdot X_1 = 1, \dots, \mathbf{h}_4 \cdot X_{K-1} = 1, \mathbf{h}_4 \cdot X_K = 0. \quad (14)$$

Through some basic algebra, one can form public matrix  $\mathbf{M}_{\text{cir}} \in \mathbb{Z}_2^{K \times 4K}$  and vector  $\mathbf{v}_{\text{cir}} \in \mathbb{Z}_2^K$ , secret vector  $\mathbf{w}_{\text{cir}} = (X_1 \parallel \dots \parallel X_{K-1} \parallel X_K) \in \mathbb{Z}_2^{4K}$  such that (14) is equivalent to  $\mathbf{M}_{\text{cir}} \cdot \mathbf{w}_{\text{cir}} = \mathbf{v}_{\text{cir}}$ .

**Putting it together.** We are ready to transform the considered relation  $\text{R}_{\text{fdabs}}$  into one equation of the desired form  $\mathbf{M}_{\text{fdabs}} \cdot \mathbf{w}_{\text{fdabs}} = \mathbf{v}_{\text{fdabs}}$ . The first step is to form appropriate public  $\mathbf{M}_{\text{fdabs}} \in \mathbb{Z}_2^{D_0 \times D_{\text{fdabs}}}$  and  $\mathbf{v}_{\text{fdabs}} \in \mathbb{Z}_2^{D_0}$ , and secret  $\mathbf{w}_{\text{fdabs}} \in \{0, 1\}^{D_{\text{fdabs}}}$ , with  $D_0 = \ell n + n + 1 + K$  and  $D_{\text{fdabs}} = D_{\text{acc}} + D_{\text{com}} + 4K$ , and  $\mathbf{w}_{\text{fdabs}}$  being the following form:

$$\mathbf{w}_{\text{fdabs}} = (\mathbf{w}_{\text{acc}} \parallel \mathbf{y}_\ell \parallel \text{RE}(\mathbf{x}_j) \parallel \text{RE}(\mathbf{r}_j) \parallel X_1 \parallel \dots \parallel X_{K-1} \parallel X_K), \quad (15)$$

so that  $\mathbf{M}_{\text{fdabs}} \cdot \mathbf{w}_{\text{fdabs}} = \mathbf{v}_{\text{fdabs}}$ . The next step is to specify the sets  $\text{valid}_{\text{fdabs}}$ ,  $\mathcal{R}_{\text{fdabs}}$ ,  $\mathcal{Z}_{\text{fdabs}}$ ,  $\mathcal{S}_{\text{fdabs}}$ , and  $F_{\text{fdabs}}, F'_{\text{fdabs}}, F''_{\text{fdabs}}$  such that the four constraints in Section 4.1 are satisfied. Once we have reduced the considered statement to an instance of  $\text{R}_{\text{abstract}}$ , the prover and verifier can run the protocol in Figure 4. Due to space limit, details are in Appendix B.2.

**Acknowledgements.** We thank the anonymous reviewers for their helpful comments and suggestions. The work of Yanhong Xu was supported in part by the National Key Research and Development Program under Grant 2022YFA1004900. San Ling, Khai Hanh Tang and Huaxiong Wang were supported by Singapore Ministry of Education Academic Research Fund Tier 2 Grant MOE2019-T2-2-083. Duong Hieu Phan was supported in part by the “Banque Publique

d’Investissement” under the VisioConfiance project. This research is supported by the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

## References

1. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Symposium on the Theory of Computing - STOC 1996*, pages 99–108. ACM, 1996.
2. Q. Alamélou, O. Blazy, S. Cauchie, and P. Gaborit. A code-based group signature scheme. *Des. Codes Cryptogr.*, 82(1-2):469–493, 2017.
3. B. Applebaum, N. Haramaty, Y. Ishai, E. Kushilevitz, and V. Vaikuntanathan. Low-complexity cryptographic hash functions. In *Innovations in Theoretical Computer Science Conference - ITCS 2017*, volume 67 of *LIPICs*, pages 7:1–7:31. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.
4. N. Aragon, O. Blazy, P. Gaborit, A. Hauteville, and G. Zémor. Durandal: A rank metric based signature scheme. In *Advances in Cryptology - EUROCRYPT 2019*, volume 11478 of *Lecture Notes in Computer Science*, pages 728–758. Springer, 2019.
5. D. Augot, M. Finiasz, and N. Sendrier. A fast provably secure cryptographic hash function. *IACR Cryptol. ePrint Arch.* 2003/230, 2003.
6. D. Augot, M. Finiasz, and N. Sendrier. A family of fast syndrome based cryptographic hash functions. In *Progress in Cryptology - Mycrypt 2005*, volume 3715 of *Lecture Notes in Computer Science*, pages 64–83. Springer, 2005.
7. R. E. Bansarkhani and A. El Kaafarani. Post-quantum attribute-based signatures from lattice assumptions. *IACR Cryptol. ePrint Arch.* 2016/823, 2016.
8. M. Bellare and G. Fuchsbauer. Policy-based signatures. In *Public-Key Cryptography - PKC 2014*, volume 8383 of *Lecture Notes in Computer Science*, pages 520–537. Springer, 2014.
9. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In D. E. Denning, R. Pyle, R. Ganesan, R. S. Sandhu, and V. Ashby, editors, *Conference on Computer and Communications Security - CCS 1993*, pages 62–73. ACM, 1993.
10. M. Ben-Or. Probabilistic algorithms in finite fields. In *Annual Symposium on Foundations of Computer Science - FOCS 1981*, pages 394–398. IEEE Computer Society, 1981.
11. F. Benhamouda, J. Camenisch, S. Krenn, V. Lyubashevsky, and G. Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *Advances in Cryptology - ASIACRYPT 2014*, volume 8873 of *Lecture Notes in Computer Science*, pages 551–572. Springer, 2014.
12. D. J. Bernstein, T. Lange, C. Peters, and P. Schwabe. Faster 2-regular information-set decoding. In *Coding and Cryptology - IWCC 2011*, volume 6639 of *Lecture Notes in Computer Science*, pages 81–98. Springer, 2011.
13. O. Biçer and A. Küpçü. Versatile ABS: usage limited, revocable, threshold traceable, authority hiding, decentralized attribute based signatures. *IACR Cryptol. ePrint Arch.* 2019/203, 2019.
14. L. Bidoux, P. Gaborit, M. Kulkarni, and V. Mateu. Code-based signatures from new proofs of knowledge for the syndrome decoding problem. *Des. Codes Cryptogr.*, 91(2):497–544, 2023.

15. D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. Random oracles in a quantum world. In *Advances in Cryptology - ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 41–69. Springer, 2011.
16. J. Bootle, A. Cerulli, P. Chaidos, E. Ghadafi, and J. Groth. Foundations of fully dynamic group signatures. In *Applied Cryptography and Network Security - ACNS 2016*, volume 9696 of *Lecture Notes in Computer Science*, pages 117–136. Springer, 2016.
17. J. Bootle, A. Cerulli, P. Chaidos, E. Ghadafi, and J. Groth. Foundations of fully dynamic group signatures. *J. Cryptol.*, 33(4):1822–1870, 2020.
18. J. Bootle, V. Lyubashevsky, and G. Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In *Advances in Cryptology - CRYPTO 2019*, volume 11692 of *Lecture Notes in Computer Science*, pages 176–202. Springer, 2019.
19. X. Boyen. Mesh signatures. In *Advances in Cryptology - EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 210–227. Springer, 2007.
20. E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In *Public-Key Cryptography - PKC 2014*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519. Springer, 2014.
21. P. Branco and P. Mateus. A code-based linkable ring signature scheme. In *Provable Security - ProvSec 2018*, volume 11192 of *Lecture Notes in Computer Science*, pages 203–219. Springer, 2018.
22. E. Bresson and J. Stern. Efficient revocation in group signatures. In *Public Key Cryptography - PKC 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 190–206. Springer, 2001.
23. J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76. Springer, 2002.
24. E. Carozza, G. Couteau, and A. Joux. Short signatures from regular syndrome decoding in the head. In *Advances in Cryptology - EUROCRYPT 2023*, volume 14008 of *Lecture Notes in Computer Science*, pages 532–563. Springer, 2023.
25. M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, and G. Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In *Conference on Computer and Communications Security - CCS 2017*, pages 1825–1842. ACM, 2017.
26. D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Commun. ACM*, 28(10):1030–1044, 1985.
27. D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology - EUROCRYPT 1991*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer, 1991.
28. S. Cheng, K. Nguyen, and H. Wang. Policy-based signature scheme from lattices. *Des. Codes Cryptogr.*, 81(1):43–74, 2016.
29. R. Cramer. *Modular Design of Secure yet Practical Cryptographic Protocols*. PhD thesis, Jan. 1997.
30. L. Dallet and D. Vergnaud. Provably secure code-based threshold ring signatures. In *Cryptography and Coding - IMACC 2009*, volume 5921 of *Lecture Notes in Computer Science*, pages 222–235. Springer, 2009.
31. P. Datta, T. Okamoto, and K. Takashima. Efficient attribute-based signatures for unbounded arithmetic branching programs. In *Public-Key Cryptography - PKC*

- 2019, volume 11442 of *Lecture Notes in Computer Science*, pages 127–158. Springer, 2019.
32. T. Debris-Alazard, N. Sendrier, and J. Tillich. Wave: A new family of trapdoor one-way preimage sampleable functions based on codes. In *Advances in Cryptology - ASIACRYPT 2019*, volume 11921 of *Lecture Notes in Computer Science*, pages 21–51. Springer, 2019.
  33. C. C. Dragan, D. Gardham, and M. Manulis. Hierarchical attribute-based signatures. In *Cryptology and Network Security - CANS 2018*, volume 11124 of *Lecture Notes in Computer Science*, pages 213–234. Springer, 2018.
  34. L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(1):238–268, 2018.
  35. A. El Kaafarani, L. Chen, E. Ghadafi, and J. H. Davenport. Attribute-based signatures with user-controlled linkability. In *Cryptology and Network Security - CANS 2014*, volume 8813 of *Lecture Notes in Computer Science*, pages 256–269. Springer, 2014.
  36. A. El Kaafarani, E. Ghadafi, and D. Khader. Decentralized traceable attribute-based signatures. In *Topics in Cryptology - CT-RSA 2014*, volume 8366 of *Lecture Notes in Computer Science*, pages 327–348. Springer, 2014.
  37. A. El Kaafarani and S. Katsumata. Attribute-based signatures for unbounded circuits in the ROM and efficient instantiations from lattices. In *Public-Key Cryptography - PKC 2018*, volume 10770 of *Lecture Notes in Computer Science*, pages 89–119. Springer, 2018.
  38. A. Escala, J. Herranz, and P. Morillo. Revocable attribute-based signatures with adaptive security in the standard model. In *Progress in Cryptology - AFRICACRYPT 2011*, volume 6737 of *Lecture Notes in Computer Science*, pages 224–241. Springer, 2011.
  39. M. F. Esgin, N. K. Nguyen, and G. Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In *Advances in Cryptology - ASIACRYPT 2020*, volume 12492 of *Lecture Notes in Computer Science*, pages 259–288. Springer, 2020.
  40. M. F. Ezerman, H. T. Lee, S. Ling, K. Nguyen, and H. Wang. A provably secure group signature scheme from code-based assumptions. In *Advances in Cryptology - ASIACRYPT 2015*, volume 9452 of *Lecture Notes in Computer Science*, pages 260–285. Springer, 2015.
  41. M. F. Ezerman, H. T. Lee, S. Ling, K. Nguyen, and H. Wang. Provably secure group signature schemes from code-based assumptions. *IEEE Trans. Inf. Theory*, 66(9):5754–5773, 2020.
  42. T. Feneuil, A. Joux, and M. Rivain. Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs. In *Advances in Cryptology - CRYPTO 2022*, volume 13508 of *Lecture Notes in Computer Science*, pages 541–572. Springer, 2022.
  43. T. Feneuil, A. Joux, and M. Rivain. Shared permutation for syndrome decoding: new zero-knowledge protocol and code-based signature. *Des. Codes Cryptogr.*, 91(2):563–608, 2023.
  44. H. Feng, J. Liu, and Q. Wu. Secure stern signatures in quantum random oracle model. In *Information Security - ISC 2019*, volume 11723 of *Lecture Notes in Computer Science*, pages 425–444. Springer, 2019.
  45. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO 1986*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

46. D. Gardham and M. Manulis. Revocable hierarchical attribute-based signatures from lattices. In *Applied Cryptography and Network Security - ACNS 2022*, volume 13269 of *Lecture Notes in Computer Science*, pages 459–479. Springer, 2022.
47. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Symposium on Theory of Computing - STOC 2008*, pages 197–206. ACM, 2008.
48. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.
49. J. Herranz. Attribute-based signatures from RSA. *Theor. Comput. Sci.*, 527:73–82, 2014.
50. Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge from secure multiparty computation. In *Symposium on Theory of Computing - STOC 2007*, pages 21–30. ACM, 2007.
51. A. Jain, S. Krenn, K. Pietrzak, and A. Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *Advances in Cryptology - ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 663–680. Springer, 2012.
52. J. Katz, V. Kolesnikov, and X. Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In *Conference on Computer and Communications Security - CCS 2018*, pages 525–537. ACM, 2018.
53. A. Kawachi, K. Tanaka, and K. Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *Advances in Cryptology - ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*, pages 372–389. Springer, 2008.
54. J. Li, M. H. Au, W. Susilo, D. Xie, and K. Ren. Attribute-based signature and its applications. In *Symposium on Information, Computer and Communications Security - ASIACCS 2010*, pages 60–69. ACM, 2010.
55. Y. Lian, L. Xu, and X. Huang. Attribute-based signatures with efficient revocation. In *International Conference on Intelligent Networking and Collaborative Systems - INCoS 2013*, pages 573–577. IEEE, 2013.
56. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In *Advances in Cryptology - ASIACRYPT 2016*, volume 10032 of *Lecture Notes in Computer Science*, pages 373–403, 2016.
57. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. *Theor. Comput. Sci.*, 759:72–97, 2019.
58. B. Libert, S. Ling, K. Nguyen, and H. Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *Advances in Cryptology - EUROCRYPT 2016*, volume 9666 of *Lecture Notes in Computer Science*, pages 1–31. Springer, 2016.
59. B. Libert, S. Ling, K. Nguyen, and H. Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. *J. Cryptol.*, 36(3):23, 2023.
60. B. Libert, T. Peters, and M. Yung. Group signatures with almost-for-free revocation. In *Advances in Cryptology - CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 571–589. Springer, 2012.
61. B. Libert, T. Peters, and M. Yung. Scalable group signatures with revocation. In *Advances in Cryptology - EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 609–627. Springer, 2012.

62. S. Ling, K. Nguyen, D. H. Phan, H. Tang, and H. Wang. Zero-knowledge proofs for committed symmetric boolean functions. In *Post-Quantum Cryptography - PQCrypto 2021*, volume 12841 of *Lecture Notes in Computer Science*, pages 339–359. Springer, 2021.
63. S. Ling, K. Nguyen, D. Stehlé, and H. Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In *Public-Key Cryptography - PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 107–124. Springer, 2013.
64. S. Ling, K. Nguyen, H. Wang, and Y. Xu. Accountable tracing signatures from lattices. In *Topics in Cryptology - CT-RSA 2019*, volume 11405 of *Lecture Notes in Computer Science*, pages 556–576. Springer, 2019.
65. S. Ling, K. Nguyen, H. Wang, and Y. Xu. Lattice-based group signatures: Achieving full dynamicity (and deniability) with ease. *Theor. Comput. Sci.*, 783:71–94, 2019.
66. V. Lyubashevsky. Lattice signatures without trapdoors. In *Advances in Cryptology - EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 738–755. Springer, 2012.
67. V. Lyubashevsky and D. Micciancio. Asymptotically efficient lattice-based digital signatures. *J. Cryptol.*, 31(3):774–797, 2018.
68. H. K. Maji, M. Prabhakaran, and M. Rosulek. Attribute-based signatures. In *Topics in Cryptology - CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 376–392. Springer, 2011.
69. C. A. Melchor, P. Cayrel, P. Gaborit, and F. Laguillaumie. A new efficient threshold ring signature scheme based on coding theory. *IEEE Trans. Inf. Theory*, 57(7):4833–4842, 2011.
70. K. Morozov and T. Takagi. Zero-knowledge protocols for the mceliece encryption. In *Information Security and Privacy - ACISP 2012*, volume 7372 of *Lecture Notes in Computer Science*, pages 180–193. Springer, 2012.
71. K. Nguyen, F. Guo, W. Susilo, and G. Yang. Multimodal private signatures. In *Advances in Cryptology - CRYPTO 2022*, volume 13508 of *Lecture Notes in Computer Science*, pages 792–822. Springer, 2022.
72. K. Nguyen, P. S. Roy, W. Susilo, and Y. Xu. Bicameral and auditably private signatures. In *Advances in Cryptology - ASIACRYPT 2023*, volume 14439 of *Lecture Notes in Computer Science*, pages 313–347. Springer, 2023.
73. K. Nguyen, R. Safavi-Naini, W. Susilo, H. Wang, Y. Xu, and N. Zeng. Group encryption: Full dynamicity, message filtering and code-based instantiation. In *Public-Key Cryptography - PKC 2021*, volume 12711 of *Lecture Notes in Computer Science*, pages 678–708. Springer, 2021.
74. K. Nguyen, H. Tang, H. Wang, and N. Zeng. New code-based privacy-preserving cryptographic constructions. In *Advances in Cryptology - ASIACRYPT 2019*, volume 11922 of *Lecture Notes in Computer Science*, pages 25–55. Springer, 2019.
75. T. Okamoto and K. Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In *Public Key Cryptography - PKC 2011*, volume 6571 of *Lecture Notes in Computer Science*, pages 35–52. Springer, 2011.
76. T. Okamoto and K. Takashima. Decentralized attribute-based signatures. In *Public-Key Cryptography - PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 125–142. Springer, 2013.
77. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009.

78. R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.
79. Y. Sakai, N. Attrapadung, and G. Hanaoka. Attribute-based signatures for circuits from bilinear map. In *Public-Key Cryptography - PKC 2016*, volume 9614 of *Lecture Notes in Computer Science*, pages 283–300. Springer, 2016.
80. Y. Sakai, S. Katsumata, N. Attrapadung, and G. Hanaoka. Attribute-based signatures for unbounded languages from standard assumptions. In *Advances in Cryptology - ASIACRYPT 2018*, volume 11273 of *Lecture Notes in Computer Science*, pages 493–522. Springer, 2018.
81. J. Stern. A new identification scheme based on syndrome decoding. In *Advances in Cryptology - CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*, pages 13–21. Springer, 1993.
82. J. Stern. A new paradigm for public key identification. *IEEE Trans. Inf. Theory*, 42(6):1757–1768, 1996.
83. Q. Su, R. Zhang, R. Xue, and P. Li. Revocable attribute-based signature for blockchain-based healthcare system. *IEEE Access*, 8:127884–127896, 2020.
84. S. R. Tate and R. Vishwanathan. Expiration and revocation of keys for attribute-based signatures. In *Data and Applications Security and Privacy - DBSec 2015*, volume 9149 of *Lecture Notes in Computer Science*, pages 153–169. Springer, 2015.
85. R. Tsabary. An equivalence between attribute-based signatures and homomorphic signatures, and new constructions for both. In *Theory of Cryptography - TCC 2017*, volume 10678 of *Lecture Notes in Computer Science*, pages 489–518. Springer, 2017.
86. D. Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *Advances in Cryptology - EUROCRYPT 2015*, volume 9057 of *Lecture Notes in Computer Science*, pages 755–784. Springer, 2015.
87. M. Urquidi, D. Khader, J. Lancrenon, and L. Chen. Attribute-based signatures with controllable linkability. In *Trusted Systems - INTRUST 2015*, volume 9565 of *Lecture Notes in Computer Science*, pages 114–129. Springer, 2015.
88. R. Yang, M. H. Au, Z. Zhang, Q. Xu, Z. Yu, and W. Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In *Advances in Cryptology - CRYPTO 2019*, volume 11692 of *Lecture Notes in Computer Science*, pages 147–175. Springer, 2019.
89. T. H. Yuen, J. K. Liu, X. Huang, M. H. Au, W. Susilo, and J. Zhou. Forward secure attribute-based signatures. In *Information and Communications Security - ICICS 2012*, volume 7618 of *Lecture Notes in Computer Science*, pages 167–177. Springer, 2012.
90. G. Zaverucha, M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, J. Katz, X. Wang, et al. Picnic. *NIST Post-Quantum Standardization Project Round 3*, 2017.

## A Additional Code-Based Cryptographic Tools

**The AFS hash functions.** Let  $\lambda$  be the security parameter. The AFS family of hash functions  $\mathcal{H}_{\text{afs}}$  maps  $\{0, 1\}^k$  to  $\{0, 1\}^n$ , where  $n, k = \Omega(\lambda)$  and  $k > n$ . Each function in the family is associated with a matrix  $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_2^{n \times 2^c \cdot k/c}$ , for some properly chosen constant  $c$  dividing  $k$ . To compute the hash value of  $\mathbf{x} \in \{0, 1\}^k$ , one encodes it to the corresponding regular word  $\text{RE}(\mathbf{x}) \in \{0, 1\}^{2^c \cdot k/c}$  and outputs  $\mathbf{B} \cdot \text{RE}(\mathbf{x})$ .

The above hash functions are collision-resistant assuming the hardness of the 2-RNSD $_{n,k,c}$  problem. Suppose that the adversary can produce distinct  $\mathbf{x}_0, \mathbf{x}_1$  such that  $\mathbf{B} \cdot \text{RE}(\mathbf{x}_0) = \mathbf{B} \cdot \text{RE}(\mathbf{x}_1)$ . Let  $\mathbf{z} = \text{RE}(\mathbf{x}_0) \oplus \text{RE}(\mathbf{x}_1) \neq \mathbf{0}$  then we have  $\mathbf{z} \in 2\text{-Regular}(k, c)$  and  $\mathbf{B} \cdot \mathbf{z} = \mathbf{0}$ . In other words,  $\mathbf{z}$  is a solution to the 2-RNSD $_{n,k,c}$  problem associated with matrix  $\mathbf{B}$ .

**The Modified AFS Hash Function.** Nguyen et al. [74] recently modified the AFS hash function family [6] so that it takes 2 inputs (instead of just 1) and hence is suitable for building Merkle hash trees. The definition is given below.

**Definition 5.** Let  $m = 2 \cdot 2^c \cdot n/c$ . The function family  $\mathcal{H}$  mapping  $\{0, 1\}^n \times \{0, 1\}^n$  to  $\{0, 1\}^n$  is defined as  $\mathcal{H} = \{h_{\mathbf{B}} \mid \mathbf{B} \in \mathbb{Z}_2^{n \times m}\}$ , where for  $\mathbf{B} = [\mathbf{B}_0 \mid \mathbf{B}_1]$  with  $\mathbf{B}_0, \mathbf{B}_1 \in \mathbb{Z}_2^{n \times m/2}$ , and for any  $(\mathbf{u}_0, \mathbf{u}_1) \in \{0, 1\}^n \times \{0, 1\}^n$ , we have:

$$h_{\mathbf{B}}(\mathbf{u}_0, \mathbf{u}_1) = \mathbf{B}_0 \cdot \text{RE}(\mathbf{u}_0) \oplus \mathbf{B}_1 \cdot \text{RE}(\mathbf{u}_1) \in \{0, 1\}^n.$$

The collision resistance of the hash function family relies on the hardness of the 2-RNSD problem.

**Lemma 1 ([74]).** The function family  $\mathcal{H}$ , defined in Definition 5 is collision-resistant, assuming the hardness of the 2-RNSD $_{n,2n,c}$  problem.

### A.1 Code-Based Commitment Scheme

Let us recall the statistically hiding and computationally binding commitment scheme proposed in [74].

**CSetup( $1^\lambda$ ):** Given the security parameter  $1^\lambda$ , it chooses  $n = \mathcal{O}(\lambda)$ ,  $k \geq n + 2\lambda + \mathcal{O}(1)$ , and specifies the message space  $\mathcal{X} = \{0, 1\}^L$ . It also chooses  $c = \mathcal{O}(1)$  that divides both  $k$  and  $L$ . Let  $m_0 = 2^c \cdot L/c$  and  $m_1 = 2^c \cdot k/c$ .

Sample  $\mathbf{C}_0 \xleftarrow{\$} \mathbb{Z}_2^{n \times m_0}$  and  $\mathbf{C}_1 \xleftarrow{\$} \mathbb{Z}_2^{n \times m_1}$ . Output public parameter  $\text{pp} = \{\lambda, n, k, L, c, m_0, m_1, \mathbf{C}_0, \mathbf{C}_1\}$ .

**CCom(pp,  $\mathbf{x}$ ):** To commit to a message  $\mathbf{x} \in \{0, 1\}^L$ , this algorithm samples a randomness  $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_2^{n \times k}$ , computes  $\mathbf{c} = \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}) \oplus \mathbf{C}_1 \cdot \text{RE}(\mathbf{r})$ , and outputs commitment  $\mathbf{c}$  as well as the opening  $\mathbf{r}$ .

**COpen(pp,  $\mathbf{c}$ , ( $\mathbf{x}$ ,  $\mathbf{r}$ )):** Given the inputs, it outputs 1 if  $\mathbf{c} = \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}) \oplus \mathbf{C}_1 \cdot \text{RE}(\mathbf{r})$  and 0 otherwise.

**Lemma 2 ([74]).** The above commitment scheme is correct. For any  $\mathbf{x} \in \{0, 1\}^L$ , the distribution of commitment  $\mathbf{c}$  is statistically close to the uniform distribution over  $\mathbb{Z}_2^n$ . In particular, the scheme satisfies the statistical hiding property. Moreover, if 2-RNSD $_{n,L+k,c}$  problem is hard, then the scheme is also computationally binding.

As pointed out in [74], the above commitment scheme can be extended to commit messages of arbitrary length using Merkle-Damgård technique. In this work, we will employ the above commitment scheme to commit to attributes and the extended one as a building block for Stern's ZK protocols.

## A.2 Efficiently Updatable Code-Based Merkle-tree Accumulator

We now recall the updatable code-based Merkle-tree accumulator [74,73].

**TSetup**( $1^\lambda$ ): This algorithm first chooses  $n = \mathcal{O}(\lambda)$ ,  $c = \mathcal{O}(1)$  so that  $c$  divides  $n$ . Set  $m = 2 \cdot 2^c \cdot n/c$ . It then samples  $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_2^{n \times m}$ , and outputs the public parameter  $pp = \{\lambda, n, c, m, \mathbf{B}\}$ .

**TAcc<sub>B</sub>**( $R = \{\mathbf{d}_0, \dots, \mathbf{d}_{N-1}\} \subseteq (\{0,1\}^n)^N$ ): Assume  $N = 2^\ell$  without loss of generality. Re-write  $\mathbf{d}_j$  as  $\mathbf{u}_{\ell,j}$  and call  $\mathbf{d}_j$  the leaf value of the leaf node  $\text{bin}(j)$  for  $j \in [0, N-1]$ . Build a binary tree upon  $N$  leaves  $\mathbf{u}_{\ell,0}, \dots, \mathbf{u}_{\ell,2^\ell-1}$  in the following way. For  $k \in \{\ell-1, \ell-2, \dots, 1, 0\}$  and  $i \in [0, 2^k-1]$ , compute  $\mathbf{u}_{k,i} = h_{\mathbf{B}}(\mathbf{u}_{k+1,2i}, \mathbf{u}_{k+1,2i+1})$ . Output the accumulated value  $\mathbf{u} = \mathbf{u}_{0,0}$ .

**TWitGen<sub>B</sub>**( $R, \mathbf{d}$ ): If  $\mathbf{d} \notin R$ , the algorithm outputs  $\perp$ . Otherwise, it outputs the witness  $w$  for  $\mathbf{d}$  as follows.

1. Set  $\mathbf{d} = \mathbf{d}_j$  for some  $j \in [0, N-1]$ . Re-write  $\mathbf{d}_j = \mathbf{u}_{\ell,j}$ . Let  $\text{bin}(j) = [j_1 | \dots | j_\ell]^\top \in \{0,1\}^\ell$  be the binary representation of  $j$ .
2. Consider the path from  $\mathbf{u}_{\ell,j}$  to the root  $\mathbf{u}$ , the witness  $w$  then consists of  $\text{bin}(j)$  as well as all the sibling nodes of the path. Let  $w = (\text{bin}(j), (\mathbf{w}_\ell, \dots, \mathbf{w}_1)) \in \mathbb{Z}_2^\ell \times (\mathbb{Z}_2^n)^\ell$ . We give an example in Figure 5.

**TVerify<sub>B</sub>**( $\mathbf{u}, \mathbf{d}, w$ ): Let  $w$  be of the following form:

$$w = ([j_1 | \dots | j_\ell]^\top, (\mathbf{w}_\ell, \dots, \mathbf{w}_1)).$$

This algorithm then computes  $\mathbf{v}_\ell, \dots, \mathbf{v}_0$ . Let  $\mathbf{v}_\ell = \mathbf{d}$  and

$$\forall i \in \{\ell-1, \dots, 1, 0\} : \mathbf{v}_i = \begin{cases} h_{\mathbf{B}}(\mathbf{v}_{i+1}, \mathbf{w}_{i+1}), & \text{if } j_{i+1} = 0; \\ h_{\mathbf{B}}(\mathbf{w}_{i+1}, \mathbf{v}_{i+1}), & \text{if } j_{i+1} = 1. \end{cases} \quad (16)$$

Output 1 if  $\mathbf{v}_0 = \mathbf{u}$  or 0 otherwise.

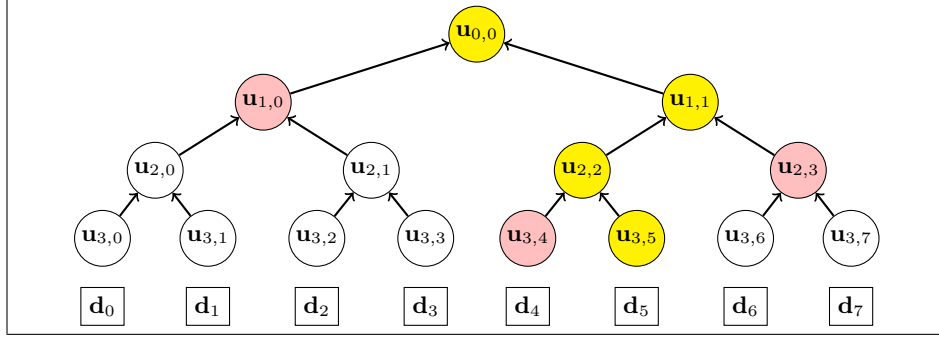
**TUpdate<sub>B</sub>**( $\text{bin}(j), \mathbf{d}^*$ ): Let  $\mathbf{d}_j$  be the existing leaf value of the leaf node  $\text{bin}(j)$ . It executes the algorithm **TWitGen<sub>B</sub>**( $R, \mathbf{d}_j$ ), obtaining  $w = (\text{bin}(j), (\mathbf{w}_\ell, \dots, \mathbf{w}_1))$ . It then sets  $\mathbf{v}_\ell = \mathbf{d}^*$  and recursively computes  $\mathbf{v}_{\ell-1}, \dots, \mathbf{v}_0$  as in (16). Finally, for  $i \in [0, \ell]$ , it sets  $\mathbf{u}_{i, \lfloor \frac{j}{2^{\ell-i}} \rfloor} = \mathbf{v}_i$ .

**Lemma 3** ([74]). *Assume that the 2-RNSD<sub>n,2n,c</sub> problem is hard, then the given accumulator scheme is correct and secure, i.e., it is infeasible to prove that a value  $\mathbf{d}^*$  was accumulated in a value  $\mathbf{u}$  if it was not (see, e.g., [58,23] for formal definition).*

## B Detailed Descriptions of Supporting ZK Protocols

### B.1 New Zero-Knowledge Proof of Set Membership

This section presents our new ZK protocol for proving set membership. Let  $n$  be an integer and  $m = 2 \cdot 2^c \cdot n/c$ . Given a random matrix  $\mathbf{B} = [\mathbf{B}_0 | \mathbf{B}_1] \in \mathbb{Z}_2^{n \times m}$



**Fig. 5:** A Merkle tree with  $2^3 = 8$  leaves, which accumulates the data blocks  $\mathbf{d}_0, \dots, \mathbf{d}_7$  into the value  $\mathbf{u} = \mathbf{u}_{0,0}$  at the root. Let  $j = 5$ . Then the path from  $\mathbf{u}_{3,5} = \mathbf{d}_5$  to the root consists of the yellow nodes, whose siblings are the pink nodes. Hence,  $\text{bin}(j)$  as well as the pink nodes form a witness to the fact that  $\mathbf{d}_5$  is accumulated in  $\mathbf{u}$ . If one needs to update  $\mathbf{d}_5$  to  $\mathbf{d}^*$ , it suffices to update the yellow nodes.

and the accumulated value  $\mathbf{u}$ , the target of  $\mathcal{P}$  is to convince  $\mathcal{V}$  that it possesses a value  $\mathbf{d} \in \{0, 1\}^n$  and a witness  $w \in \{0, 1\}^\ell \times (\{0, 1\}^n)^\ell$ . Define the associated relation as follows:

$$\mathbf{R}_{\text{acc}} = \left\{ \left( (\mathbf{B}, \mathbf{u}); (\mathbf{d}, w) \right) \in \left( \mathbb{Z}_2^{n \times m} \times \mathbb{Z}_2^n \right) \times \left( \{0, 1\}^n \times \{0, 1\}^\ell \times (\{0, 1\}^n)^\ell \right) : \right. \\ \left. \text{TVerify}_{\mathbf{B}}(\mathbf{u}, \mathbf{d}, w) = 1 \right\}.$$

Before constructing our ZK protocol, let us first recall some notions from [74].

- For  $d \in \{0, 1\}$  and  $\mathbf{x} \in \{0, 1\}^{\frac{m}{2}}$ , denote  $\text{Ext}(d, \mathbf{x}) = (\bar{d} \cdot \mathbf{x} \parallel d \cdot \mathbf{x}) \in \{0, 1\}^m$ .
- For  $a \in \{0, 1\}$ , for  $\mathbf{b} \in \{0, 1\}^n$ , for  $\mathbf{p} = (\mathbf{p}_0 \parallel \mathbf{p}_1) \in \{0, 1\}^m$  with  $\mathbf{p}_i \in \{0, 1\}^{m/2}$ , define a function  $\Psi : (\{0, 1\} \times \{0, 1\}^n) \times \{0, 1\}^m \rightarrow \{0, 1\}^m$  as  $\Psi((a, \mathbf{b}), \mathbf{p}) = (F_{\text{re}}(\mathbf{b}, \mathbf{p}_a) \parallel F_{\text{re}}(\mathbf{b}, \mathbf{p}_{\bar{a}}))$ . Namely, it rearranges the blocks of  $\mathbf{p}$  according to  $a$  and then applies the function  $F_{\text{re}}$  to the rearranged blocks. Due to “homomorphism” of the function  $F_{\text{re}}$  (see (6)), for all  $a \in \{0, 1\}$ ,  $\mathbf{b} \in \{0, 1\}^n$ , and all  $\mathbf{p}, \mathbf{p}' \in \{0, 1\}^m$ , we obtain the following

$$\Psi((a, \mathbf{b}), \mathbf{p}) \oplus \Psi((a, \mathbf{b}), \mathbf{p}') = \Psi((a, \mathbf{b}), \mathbf{p} \oplus \mathbf{p}'). \quad (17)$$

- For any  $d, a \in \{0, 1\}$  and  $\mathbf{v}, \mathbf{w}, \mathbf{b}, \mathbf{c} \in \{0, 1\}^n$ , it follows from (7) that the following equivalences hold:

$$\begin{cases} \mathbf{p} = \text{Ext}(d, \text{RE}(\mathbf{v})) & \iff \Psi((a, \mathbf{b}), \mathbf{p}) = \text{Ext}(d \oplus a, \text{RE}(\mathbf{v} \oplus \mathbf{b})) \\ \mathbf{q} = \text{Ext}(\bar{d}, \text{RE}(\mathbf{w})) & \iff \Psi((a, \mathbf{c}), \mathbf{q}) = \text{Ext}(\bar{d} \oplus a, \text{RE}(\mathbf{w} \oplus \mathbf{c})). \end{cases} \quad (18)$$

Let us now take a closer look at  $\mathbf{R}_{\text{acc}}$ . Recall that  $w = (\text{bin}(j), \mathbf{w}_\ell, \dots, \mathbf{w}_1)$  and the  $\text{TVerify}$  algorithm computes  $\mathbf{v}_\ell = \mathbf{d}, \mathbf{v}_{\ell-1}, \dots, \mathbf{v}_0 = \mathbf{u}$ , where  $\mathbf{v}_i$  for

$i \in \{\ell - 1, \dots, 1, 0\}$  is computed as

$$\begin{aligned} \mathbf{v}_i &= \begin{cases} \mathbf{B}_0 \cdot \text{RE}(\mathbf{v}_{i+1}) \oplus \mathbf{B}_1 \cdot \text{RE}(\mathbf{w}_{i+1}) & \text{if } j_{i+1} = 0; \\ \mathbf{B}_0 \cdot \text{RE}(\mathbf{w}_{i+1}) \oplus \mathbf{B}_1 \cdot \text{RE}(\mathbf{v}_{i+1}) & \text{if } j_{i+1} = 1. \end{cases} \\ &= \mathbf{B} \cdot \text{Ext}(j_{i+1}, \text{RE}(\mathbf{v}_{i+1})) \oplus \mathbf{B} \cdot \text{Ext}(\bar{j}_{i+1}, \text{RE}(\mathbf{w}_{i+1})). \end{aligned}$$

For the sake of simplicity, define the following notions for all  $i \in [1, \ell]$ .

$$\begin{aligned} \mathbf{p}_i &= \text{Ext}(j_i, \text{RE}(\mathbf{v}_i)) \in \{0, 1\}^m, \quad \mathbf{q}_i = \text{Ext}(\bar{j}_i, \text{RE}(\mathbf{w}_i)) \in \{0, 1\}^m, \\ \mathbf{y}_i &= \text{Encode}(\mathbf{v}_i) \in \{0, 1\}^{2n}. \end{aligned} \quad (19)$$

Recall that we have  $\mathbf{v}_i = \mathbf{I}_n^* \cdot \mathbf{y}_i$  for  $i \in [1, \ell]$ . It is then observed that it is sufficient for  $\mathcal{P}$  to prove knowledge of  $\text{bin}(j) = [j_1 | \dots | j_\ell]^\top \in \{0, 1\}^\ell$ ,  $\mathbf{p}_1, \dots, \mathbf{p}_\ell$ ,  $\mathbf{q}_1, \dots, \mathbf{q}_\ell$ ,  $\mathbf{y}_1, \dots, \mathbf{y}_{\ell-1}$  of form (19) such that

$$\begin{cases} \mathbf{B} \cdot \mathbf{p}_1 \oplus \mathbf{B} \cdot \mathbf{q}_1 = \mathbf{u}; & \mathbf{B} \cdot \mathbf{p}_2 \oplus \mathbf{B} \cdot \mathbf{q}_2 \oplus \mathbf{I}_n^* \cdot \mathbf{y}_1 = \mathbf{0}; \\ & \dots \dots \dots \\ \mathbf{B} \cdot \mathbf{p}_{\ell-1} \oplus \mathbf{B} \cdot \mathbf{q}_{\ell-1} \oplus \mathbf{I}_n^* \cdot \mathbf{y}_{\ell-2} = \mathbf{0}; & \mathbf{B} \cdot \mathbf{p}_\ell \oplus \mathbf{B} \cdot \mathbf{q}_\ell \oplus \mathbf{I}_n^* \cdot \mathbf{y}_{\ell-1} = \mathbf{0}. \end{cases} \quad (20)$$

Through some basic algebra, we can transform the equations in (20) to a unified equation of the form  $\mathbf{M}_{\text{acc}} \cdot \mathbf{w}_{\text{acc}} = \mathbf{v}_{\text{acc}}$ , where  $\mathbf{M}_{\text{acc}} \in \mathbb{Z}_2^{\ell n \times D_{\text{acc}}}$  and  $\mathbf{v}_{\text{acc}} \in \mathbb{Z}_2^{\ell n}$  are public and  $\mathbf{w}_{\text{acc}} \in \{0, 1\}^{D_{\text{acc}}}$  is secret with  $D_{\text{acc}} = 2\ell m + 2(\ell - 1)n$  and

$$\mathbf{w}_{\text{acc}} = (\mathbf{p}_1 \parallel \dots \parallel \mathbf{p}_\ell \parallel \mathbf{q}_1 \parallel \dots \parallel \mathbf{q}_\ell \parallel \mathbf{y}_1 \parallel \dots \parallel \mathbf{y}_{\ell-1}). \quad (21)$$

Now let us specify the sets  $\text{valid}_{\text{acc}}, \mathcal{R}_{\text{acc}}, \mathcal{Z}_{\text{acc}}, \mathcal{S}_{\text{acc}}$  as well as three functions  $F_{\text{acc}}, F'_{\text{acc}}, F''_{\text{acc}}$  so that  $\text{valid}_{\text{acc}}$  contains our secret vector  $\mathbf{w}_{\text{acc}}$  and all four requirements specified in Section 4.1 are fulfilled. Let  $\text{valid}_{\text{acc}}$  contain all vectors of the form  $\hat{\mathbf{w}}_{\text{acc}} = (\hat{\mathbf{p}}_1 \parallel \dots \parallel \hat{\mathbf{p}}_\ell \parallel \hat{\mathbf{q}}_1 \parallel \dots \parallel \hat{\mathbf{q}}_\ell \parallel \hat{\mathbf{y}}_1 \parallel \dots \parallel \hat{\mathbf{y}}_{\ell-1}) \in \{0, 1\}^{D_{\text{acc}}}$  satisfying the following constraints:

- For  $i \in [1, \ell]$ , there exist  $\hat{j}_i \in \{0, 1\}$ ,  $\hat{\mathbf{v}}_i, \hat{\mathbf{w}}_i \in \{0, 1\}^n$  such that

$$\begin{aligned} \hat{\mathbf{p}}_i &= \text{Ext}(\hat{j}_i, \text{RE}(\hat{\mathbf{v}}_i)) \in \{0, 1\}^m; \quad \hat{\mathbf{q}}_i = \text{Ext}(\bar{\hat{j}}_i, \text{RE}(\hat{\mathbf{w}}_i)) \in \{0, 1\}^m, \\ \hat{\mathbf{y}}_i &= \text{Encode}(\hat{\mathbf{v}}_i) \in \{0, 1\}^{2n}. \end{aligned} \quad (22)$$

Regarding  $\mathcal{R}_{\text{acc}}$  and  $\mathcal{Z}_{\text{acc}}$ , define the following sets where  $\mathcal{R}_{\text{bin}}, \mathcal{Z}_{\text{bin}}$  are as in (5).

$$\begin{aligned} \mathcal{R}_{\text{acc}} &= (\{0, 1\}^{2\ell m} \parallel \underbrace{\mathcal{R}_{\text{bin}} \parallel \dots \parallel \mathcal{R}_{\text{bin}}}_{\ell-1 \text{ copies}}) \subset \mathbb{Z}_2^{D_{\text{acc}}}, \\ \mathcal{Z}_{\text{acc}} &= (\{0, 1\}^{2\ell m} \parallel \underbrace{\mathcal{Z}_{\text{bin}} \parallel \dots \parallel \mathcal{Z}_{\text{bin}}}_{\ell-1 \text{ copies}}) \subset \mathbb{Z}_2^{D_{\text{acc}}}. \end{aligned}$$

Let  $\mathcal{S}_{\text{acc}} = (\{0, 1\}^\ell \times (\{0, 1\}^n)^\ell \times (\{0, 1\}^n)^\ell)$ . Then for each  $\phi \in \mathcal{S}_{\text{acc}}$  and  $\hat{\mathbf{w}}_{\text{acc}} \in \{0, 1\}^{D_{\text{acc}}}$  of forms

$$\phi = (g_1, \dots, g_\ell, \mathbf{b}_1, \dots, \mathbf{b}_\ell, \mathbf{c}_1, \dots, \mathbf{c}_\ell), \quad (23)$$

$$\hat{\mathbf{w}}_{\text{acc}} = (\hat{\mathbf{p}}_1 \parallel \dots \parallel \hat{\mathbf{p}}_\ell \parallel \hat{\mathbf{q}}_1 \parallel \dots \parallel \hat{\mathbf{q}}_\ell \parallel \hat{\mathbf{y}}_1 \parallel \dots \parallel \hat{\mathbf{y}}_{\ell-1}), \quad (24)$$

where  $\widehat{\mathbf{p}}_i, \widehat{\mathbf{q}}_i \in \mathbb{Z}_2^m$  for  $i \in [1, \ell]$  and  $\widehat{\mathbf{y}}_i \in \mathbb{Z}_2^{2^n}$  for  $i \in [1, \ell - 1]$ , define  $F_{\text{acc}} : \mathcal{S}_{\text{acc}} \times \{0, 1\}^{D_{\text{acc}}} \rightarrow \{0, 1\}^{D_{\text{acc}}}$  as

$$F_{\text{acc}}(\phi, \widehat{\mathbf{w}}_{\text{acc}}) = (\mathbf{p}_1^* \parallel \dots \parallel \mathbf{p}_\ell^* \parallel \mathbf{q}_1^* \parallel \dots \parallel \mathbf{q}_\ell^* \parallel \mathbf{y}_1^* \parallel \dots \parallel \mathbf{y}_{\ell-1}^*),$$

where for  $i \in [1, \ell]$ ,

$$\mathbf{p}_i^* = \Psi(g_i, \mathbf{b}_i), \widehat{\mathbf{p}}_i), \quad \mathbf{q}_i^* = \Psi(g_i, \mathbf{c}_i), \widehat{\mathbf{q}}_i), \quad \mathbf{y}_i^* = F_{\text{bin}}(\mathbf{b}_i, \widehat{\mathbf{y}}_i). \quad (25)$$

Similarly,  $F'_{\text{acc}} : \mathcal{S}_{\text{acc}} \times \mathcal{R}_{\text{acc}} \rightarrow \mathcal{R}_{\text{acc}}$ ,  $F''_{\text{acc}} : \mathcal{S}_{\text{acc}} \times \mathcal{Z}_{\text{acc}} \rightarrow \mathcal{Z}_{\text{acc}}$  are defined as  $F'_{\text{acc}}(\phi, \widehat{\mathbf{w}}_{\text{acc}}) = F_{\text{acc}}(\phi, \widehat{\mathbf{w}}_{\text{acc}})$ ,  $F''_{\text{acc}}(\phi, \widehat{\mathbf{w}}_{\text{acc}}) = F_{\text{acc}}(\phi, \widehat{\mathbf{w}}_{\text{acc}})$ , respectively. Note that  $F_{\text{acc}}(\phi, \cdot)$  is indeed a permutation, thus  $F'_{\text{acc}}, F''_{\text{acc}}$  are well defined. We now show that the four requirements in Section 4.1 are satisfied.

First, “homomorphism” is satisfied due to that of  $\Psi$  and  $F_{\text{bin}}$  (see (17) and (3), respectively). Next, observe that if  $\widehat{\mathbf{w}}_{\text{acc}} \in \text{valid}_{\text{acc}}$ , i.e., (22) is satisfied, then due to equivalences observed in (18) and (4), formulas in (25) are equivalent to the following.

$$\begin{aligned} \mathbf{p}_i^* &= \text{Ext}(\widehat{j}_i \oplus g_i, \text{RE}(\widehat{\mathbf{v}}_i \oplus \mathbf{b}_i)), & \mathbf{q}_i^* &= \text{Ext}(\widehat{j}_i \oplus g_i, \text{RE}(\widehat{\mathbf{w}}_i \oplus \mathbf{c}_i)), \\ & & \mathbf{y}_i^* &= \text{Encode}(\widehat{\mathbf{v}}_i \oplus \mathbf{b}_i). \end{aligned} \quad (26)$$

For all  $\phi \in \mathcal{S}_{\text{acc}}$ , all  $\widehat{\mathbf{w}}_{\text{acc}} \in \{0, 1\}^{D_{\text{acc}}}$ , we then conclude that  $\widehat{\mathbf{w}}_{\text{acc}} \in \text{valid}_{\text{acc}}$  if and only if  $F_{\text{acc}}(\phi, \widehat{\mathbf{w}}_{\text{acc}}) \in \text{valid}_{\text{acc}}$ . Thus, “closure of  $F_{\text{acc}}$  with respect to  $\text{valid}_{\text{acc}}$ ” is satisfied. In addition, formulas in (26) also imply that for all  $\widehat{\mathbf{w}}_{\text{acc}} \in \text{valid}_{\text{acc}}$ , distributions  $\{\mathbf{t} = F_{\text{acc}}(\phi, \widehat{\mathbf{w}}_{\text{acc}}) \mid \phi \xleftarrow{\$} \mathcal{S}_{\text{acc}}\}$  and  $\{\mathbf{t} \mid \mathbf{t} \xleftarrow{\$} \text{valid}_{\text{acc}}\}$  are identical. Finally, since we use random vectors in the set  $\{0, 1\}^{2^{\ell m}}$  to mask  $(\widehat{\mathbf{p}}_1 \parallel \dots \parallel \widehat{\mathbf{p}}_\ell \parallel \widehat{\mathbf{q}}_1 \parallel \dots \parallel \widehat{\mathbf{q}}_\ell)$  and  $\mathcal{R}_{\text{bin}}$  to mask  $\widehat{\mathbf{y}}_i$ , it follows that the distributions  $\{\mathbf{z} = \widehat{\mathbf{w}}_{\text{acc}} \oplus \mathbf{r} \mid \mathbf{r} \xleftarrow{\$} \mathcal{R}_{\text{acc}}\}$  and  $\{\mathbf{z} \mid \mathbf{z} \xleftarrow{\$} \mathcal{Z}_{\text{acc}}\}$  are the same.

**The interactive protocol.** Given the above preparations, the interactive protocol works as follows.

- The public input consists of matrix  $\mathbf{M}_{\text{acc}} \in \mathbb{Z}_2^{\ell n \times D_{\text{acc}}}$  and vector  $\mathbf{v}_{\text{acc}} \in \mathbb{Z}_2^{\ell n}$  that are built from  $\mathbf{u}, \mathbf{B}$ .
- The prover’s witness is a vector  $\mathbf{w}_{\text{acc}} \in \text{valid}_{\text{acc}}$ , which is obtained from the witness  $(\mathbf{d}, w)$ .

Both parties run the protocol as in Figure 4. The protocol employs a statistically hiding and computationally binding commitment scheme COM from [74], as described in Appendix A.1. The following theorem follows directly from Theorem 2.

**Theorem 3.** *Let COM be a statistically hiding and computationally binding string commitment scheme with commitment  $n$  bits and randomness  $r$  bits. Then the protocol presented above is a  $\Sigma$ -protocol for the relation  $\mathbf{R}_{\text{acc}}$ , with perfect completeness and average communication cost  $\mathcal{O}(\lambda \log \lambda)$ .*

*Proof.* For honest-verifier ZK, we simply run the simulator of Theorem 2. Regarding special soundness, we invoke the PPT extractor of Theorem 2 to obtain a vector  $\mathbf{w}' \in \text{valid}_{\text{acc}}$  such that  $\mathbf{M}_{\text{acc}} \cdot \mathbf{w}' = \mathbf{v}_{\text{acc}}$ . Now, by “backtracking” the transformation steps, we are able to extract  $(\mathbf{d}', w')$  from  $\mathbf{w}'$  for the relation  $\mathcal{R}_{\text{acc}}$ . The perfect completeness and average communication cost directly follow from those of the abstract relation  $\mathcal{R}_{\text{abstract}}$ . Therefore, the average communication cost is

$$\begin{aligned} \zeta &= 3n + 2r + \frac{\log(|\text{valid}_{\text{acc}}|) + 2\log(|\mathcal{R}_{\text{acc}}|) + \log(|\mathcal{Z}_{\text{acc}}|) + 2\log(|\mathcal{S}_{\text{acc}}|)}{3} \\ &= \mathcal{O}(\lambda \log \lambda), \end{aligned}$$

where  $\log(|\mathcal{R}_{\text{acc}}|) = \log(|\mathcal{Z}_{\text{acc}}|) = 2\ell m + (\ell - 1)n$ ,  $\log|\mathcal{S}_{\text{acc}}| = \ell + 2\ell n$  and  $\log|\text{valid}_{\text{acc}}| = \ell + 2\ell n$ .

## B.2 Details of Combining the Three Layers of the Main Zero-Knowledge Protocol for Algorithm Sign

We now specify the sets  $\text{valid}_{\text{fdabs}}$ ,  $\mathcal{R}_{\text{fdabs}}$ ,  $\mathcal{Z}_{\text{fdabs}}$ ,  $\mathcal{S}_{\text{fdabs}}$ , and  $F_{\text{fdabs}}$ ,  $F'_{\text{fdabs}}$ ,  $F''_{\text{fdabs}}$  such that the four constraints in Section 4.1 are obeyed. Let  $\text{valid}_{\text{fdabs}}$  contain all vectors of the following form:

$$\widehat{\mathbf{w}}_{\text{fdabs}} = (\widehat{\mathbf{w}}_{\text{acc}} \parallel \widehat{\mathbf{y}}_{\ell} \parallel \widehat{\mathbf{z}}_0 \parallel \widehat{\mathbf{z}}_1 \parallel \widehat{X}_1 \parallel \dots \parallel \widehat{X}_{K-1} \parallel \widehat{X}_K), \quad (27)$$

such that

- $\widehat{\mathbf{w}}_{\text{acc}} \in \text{valid}_{\text{acc}}$ , and there exists  $\widehat{\mathbf{v}}_{\ell} \in \{0, 1\}^n$  such that  $\widehat{\mathbf{y}}_{\ell} = \text{Encode}(\widehat{\mathbf{v}}_{\ell})$ .
- There exist  $\widehat{\mathbf{x}} = [x_1 | \dots | x_L]^{\top} \in \mathbb{Z}_2^L$  and  $\widehat{\mathbf{r}} \in \mathbb{Z}_2^k$  such that  $\widehat{\mathbf{z}}_0 = \text{RE}(\widehat{\mathbf{x}}) \in \mathbb{Z}_2^{m_0}$ ,  $\widehat{\mathbf{z}}_1 = \text{RE}(\widehat{\mathbf{r}}) \in \mathbb{Z}_2^{m_1}$ .
- There exist  $x_{L+1}, \dots, x_{L+K-1} \in \mathbb{Z}_2$  such that  $\widehat{X}_i = \text{ENC}(x_{g(i)}, x_{h(i)}, x_{L+i})$  for  $i \in [1, K-1]$  and  $\widehat{X}_K = \text{ext}(x_{g(K)}, x_{h(K)})$ .

It is clear that our vector  $\mathbf{w}_{\text{fdabs}} \in \text{valid}_{\text{fdabs}}$ . Let  $\mathcal{R}_{\text{fdabs}}$ ,  $\mathcal{Z}_{\text{fdabs}}$ ,  $\mathcal{S}_{\text{fdabs}}$  be as follows.

$$\begin{aligned} \mathcal{R}_{\text{fdabs}} &= \left( \mathcal{R}_{\text{acc}} \parallel \mathcal{R}_{\text{bin}} \parallel \mathcal{R}_{\text{re},L} \parallel \mathcal{R}_{\text{re},k} \parallel \underbrace{\mathcal{R}_{\text{nand}} \parallel \dots \parallel \mathcal{R}_{\text{nand}}}_{K-1 \text{ copies}} \parallel \mathcal{R}_{\text{mult}} \right), \\ \mathcal{Z}_{\text{fdabs}} &= \left( \mathcal{Z}_{\text{acc}} \parallel \mathcal{Z}_{\text{bin}} \parallel \mathcal{Z}_{\text{re},L} \parallel \mathcal{Z}_{\text{re},k} \parallel \underbrace{\mathcal{Z}_{\text{nand}} \parallel \dots \parallel \mathcal{Z}_{\text{nand}}}_{K-1 \text{ copies}} \parallel \mathcal{Z}_{\text{mult}} \right), \\ \mathcal{S}_{\text{fdabs}} &= \left( \mathcal{S}_{\text{acc}} \times (\{0, 1\})^{L+K-1} \times \{0, 1\}^k \right). \end{aligned}$$

For  $\phi = (\phi_{\text{acc}}, d_1, \dots, d_L, d_{L+1}, \dots, d_{L+K-1}, \mathbf{e}) \in \mathcal{S}_{\text{fdabs}}$  and  $\widehat{\mathbf{w}}_{\text{fdabs}} \in \mathbb{Z}_2^{D_{\text{fdabs}}}$  of form (27), define the function  $F_{\text{fdabs}} : \mathcal{S}_{\text{fdabs}} \times \{0, 1\}^{D_{\text{fdabs}}} \rightarrow \{0, 1\}^{D_{\text{fdabs}}}$  as

$$\begin{aligned} F_{\text{fdabs}}(\phi, \widehat{\mathbf{w}}_{\text{fdabs}}) &= (F_{\text{acc}}(\phi_{\text{acc}}, \widehat{\mathbf{w}}_{\text{acc}}) \parallel F_{\text{bin}}(\mathbf{b}_{\ell}, \widehat{\mathbf{y}}_{\ell}) \parallel F_{\text{re}}(\mathbf{d}, \widehat{\mathbf{z}}_0) \parallel F_{\text{re}}(\mathbf{e}, \widehat{\mathbf{z}}_1) \parallel \\ &\quad F_{\text{nand}}(\mathbf{f}_1, \widehat{X}_1) \parallel \dots \parallel F_{\text{nand}}(\mathbf{f}_{K-1}, \widehat{X}_{K-1}) \parallel F_{\text{mult}}(\mathbf{f}_K, \widehat{X}_K)), \quad (28) \end{aligned}$$

where  $\mathbf{b}_\ell$  is an element of  $\phi_{\text{acc}}$  (see (23) for details),  $\mathbf{d} = [d_1 | \dots | d_L]^\top$ ,  $\mathbf{f}_i = [d_{g(i)} | d_{h(i)} | d_{L+i}]^\top$  for  $i \in [1, K-1]$  and  $\mathbf{f}_K = [d_{g(K)} | d_{h(K)}]^\top$ . Now we define  $F'_{\text{fdabs}} : \mathcal{S}_{\text{fdabs}} \times \mathcal{R}_{\text{fdabs}} \rightarrow \mathcal{R}_{\text{fdabs}}$  and  $F''_{\text{fdabs}} : \mathcal{S}_{\text{fdabs}} \times \mathcal{Z}_{\text{fdabs}} \rightarrow \mathcal{Z}_{\text{fdabs}}$  similarly by changing all the sub-functions to  $F'_{\text{acc}}, F'_{\text{bin}}, F'_{\text{re}}, F'_{\text{nand}}, F'_{\text{mult}}$  and  $F''_{\text{acc}}, F''_{\text{bin}}, F''_{\text{re}}, F''_{\text{nand}}, F''_{\text{mult}}$ , respectively. Since all the sub-statements with respect to sets  $\text{valid}_{\text{sub}}, \mathcal{R}_{\text{sub}}, \mathcal{Z}_{\text{sub}}, \mathcal{S}_{\text{sub}}$  and functions  $F_{\text{sub}}, F'_{\text{sub}}, F''_{\text{sub}}$  for  $\text{sub} \in \{\text{acc}, \text{bin}, \text{re}, \text{nand}, \text{mult}\}$  all satisfy the four conditions in Section 4.1, it is not hard to verify that  $\text{valid}_{\text{fdabs}}, \mathcal{R}_{\text{fdabs}}, \mathcal{Z}_{\text{fdabs}}, \mathcal{S}_{\text{fdabs}}$ , and  $F_{\text{fdabs}}, F'_{\text{fdabs}}, F''_{\text{fdabs}}$  satisfy the four constraints as well.

Now we have successfully reduced the considered statement to an instance of  $R_{\text{abstract}}$ .

**The interactive protocol.** Given the above preparations, our interactive protocol works as follows.

- The public input consists of matrix  $\mathbf{M}_{\text{fdabs}} \in \mathbb{Z}_2^{D_0 \times D_{\text{fdabs}}}$  and vector  $\mathbf{v}_{\text{fdabs}} \in \mathbb{Z}_2^{D_0}$  that are built from  $P, \mathbf{u}_\tau, \mathbf{B}, \mathbf{C}_0, \mathbf{C}_1$ .
- The prover’s witness is a vector  $\mathbf{w}_{\text{fdabs}} \in \text{valid}_{\text{fdabs}}$ , which is obtained from the witness  $\xi$ .

Both parties run the protocol as in Figure 4. The protocol employs a statistically hiding and computationally binding commitment scheme COM from [74], as described in Appendix A.1. The following theorem follows directly from Theorem 2.

**Theorem 4.** *Let COM be a statistically hiding and computationally binding string commitment scheme with commitment  $n$  bits and randomness  $r$  bits. Then the protocol presented above is a  $\Sigma$ -protocol for the relation  $R_{\text{fdabs}}$  associated with our FDABS scheme, with perfect completeness and average communication cost  $\zeta = \mathcal{O}(\lambda \log \lambda + L + K)$ .*

*Proof.* For honest-verifier ZK, we simply run the simulator of Theorem 2. Regarding special soundness, we invoke the PPT extractor of Theorem 2 to obtain a vector  $\mathbf{w}' \in \text{valid}_{\text{fdabs}}$  such that  $\mathbf{M}_{\text{fdabs}} \cdot \mathbf{w}' = \mathbf{v}_{\text{fdabs}}$ . Now, by “backtracking” the transformation steps, we are able to extract  $\xi' = (\mathbf{d}', \mathbf{x}', \mathbf{r}', \text{bin}(j'), \mathbf{w}'_\ell, \dots, \mathbf{w}'_1)$  from  $\mathbf{w}'$  for the relation  $R_{\text{fdabs}}$ . The perfect completeness and average communication cost directly follow from the counterparts of the abstract relation  $R_{\text{abstract}}$ . Therefore, the average communication cost is

$$\begin{aligned} \zeta &= 3n + 2r + \frac{\log(|\text{valid}_{\text{fdabs}}|) + 2 \log(|\mathcal{R}_{\text{fdabs}}|) + \log(|\mathcal{Z}_{\text{fdabs}}|) + 2 \log(|\mathcal{S}_{\text{fdabs}}|)}{3} \\ &= \mathcal{O}(\lambda \log \lambda + L + K), \end{aligned}$$

where

$$\begin{aligned} \log(|\mathcal{R}_{\text{fdabs}}|) &= \log(|\mathcal{Z}_{\text{fdabs}}|) = 2\ell m + \ell n + (2^c - 1) \cdot \frac{L}{c} + (2^c - 1) \cdot \frac{k}{c} + 3K, \\ \log|\mathcal{S}_{\text{fdabs}}| &= \log|\text{valid}_{\text{fdabs}}| = \ell + 2\ell n + L + K - 1 + k. \end{aligned}$$

## C Proof of Theorem 1

The proof of Theorem 1 is established by Lemma 4 and Lemma 6.

**Lemma 4.** *If the underlying NIZK protocol is statistically zero-knowledge, then our construction is statistically private.*

*Proof.* Let  $\mathcal{A}$  be any (computationally unbounded) privacy adversary against our FDABS scheme with advantage  $\epsilon$ . We will prove  $\epsilon = \text{negl}(\lambda)$  using the statistical ZK property of the underlying NIZK system, which in turn relies on the statistical ZK of the interactive Stern-like protocol in Section 4.3. Specifically, we construct indistinguishable games  $G_b, G_2$  such that  $G_b$  is the experiment  $\text{Expt}_{\mathcal{A}}^{\text{privacy-}b}(1^\lambda)$ . Denote  $W_i$  as the event that  $G_i$  outputs 1 for  $i \in \{0, 1, 2\}$ .

**Game  $G_b$ :** This game is the original privacy experiment  $\text{Expt}_{\mathcal{A}}^{\text{privacy-}b}(1^\lambda)$  for  $b = 0$  and  $b = 1$ . The challenger runs  $\text{Setup}_{\text{init}}(1^\lambda)$  to obtain  $\text{pp}$  and then gives  $\text{pp}$  to adversary  $\mathcal{A}$ . Upon receiving  $\text{pp}$ ,  $\mathcal{A}$  is allowed to compute the authority's key pair  $(\text{mpk}, \text{msk})$  as  $\mathcal{A}$  wishes. Furthermore,  $\mathcal{A}$  has full control of the registration table  $\text{reg}$  and system information  $\text{info}$  during the lifetime of the scheme. However,  $\mathcal{A}$  is required to guarantee the well-formedness of the key pair  $(\text{mpk}, \text{msk})$  and  $\text{reg}$ ,  $\text{info}$  at any time.  $\mathcal{A}$  can introduce honest attributes to the system by interacting with the oracle  $\text{SndToHX}$ . In the challenge phase,  $\mathcal{A}$  outputs two honest attributes  $\mathbf{x}_0, \mathbf{x}_1$ , a message  $M$  together with a policy  $P$  and epoch  $\tau$ . If both attributes satisfy the policy and are active at time  $\tau$ , the challenger computes a challenge signature  $\Sigma = \Pi \leftarrow \text{Sign}(\text{sk}_{\mathbf{x}_b}, M, P, \text{info}_\tau)$  and sends it to  $\mathcal{A}$ . The adversary is still allowed to access the oracle  $\text{SndToHX}$  as well as updating  $\text{reg}$ ,  $\text{info}$ . Queries to  $\mathcal{H}_{\text{FS}}$  and  $\mathcal{H}_G$  are processed by choosing random strings at their respective domains. Finally,  $\mathcal{A}$  outputs a guess  $b'$  and halts. This experiment then outputs whatever  $\mathcal{A}$  outputs. By definition,  $\Pr[\text{Expt}_{\mathcal{A}}^{\text{privacy-}b}(1^\lambda) = 1] = \Pr[W_b]$ .

**Game  $G_2$ :** In this game, the following modification is introduced. Instead of generating the NIZK proof  $\Pi$  faithfully, the challenger simulates it without using the witness tuple of form (1). This is done by running our simulator  $\text{SIM}_{\text{fdabs}}$  as described in Figure 2. Note that now the challenge signature  $\Pi$  does not depend on the challenger's bit  $b$  and is statistically close to the one in Game  $G_b$ . Hence  $|\Pr[W_b] - \Pr[W_2]| \leq \text{negl}(\lambda)$  holds for any  $\mathcal{A}$ .

It then follows that

$$\begin{aligned} \epsilon &= |\Pr[\text{Expt}_{\mathcal{A}}^{\text{privacy-}0}(1^\lambda) = 1] - \Pr[\text{Expt}_{\mathcal{A}}^{\text{privacy-}1}(1^\lambda) = 1]| = |\Pr[W_0] - \Pr[W_1]| \\ &\leq |\Pr[W_0] - \Pr[W_2]| + |\Pr[W_1 - W_2]| \leq \text{negl}(\lambda). \end{aligned}$$

We introduce the following lemma that will be utilized in the proof of Lemma 6.

**Lemma 5.** *Let  $\mathbf{C}_1 \in \mathbb{Z}_2^{n \times m_1}$ , where  $m_1 = 2^c \cdot k/c$  and  $k \geq n + 2\lambda + \mathcal{O}(1)$  as specified in Section 3.2. If  $\mathbf{r}$  is uniformly chosen from  $\{0, 1\}^k$ , then with probability  $1 - 2^{-2\lambda}$ , there exists a different  $\mathbf{r}' \in \{0, 1\}^k$  such that  $\mathbf{C}_1 \cdot \text{RE}(\mathbf{r}) = \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}') \pmod{2}$ .*

*Proof.* There are at most  $2^n - 1$  elements  $\mathbf{r} \in \{0, 1\}^k$  for which there is no other  $\mathbf{r}'$  such that  $\mathbf{C}_1 \cdot \text{RE}(\mathbf{r}) = \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}') \pmod 2$ . Therefore, for a randomly chosen  $\mathbf{r}$ , the probability that it does have a corresponding  $\mathbf{r}'$  such that  $\mathbf{C}_1 \cdot \text{RE}(\mathbf{r}) = \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}') \pmod 2$  is at least  $(2^k - 2^n + 1)/2^k \geq 1 - 2^{-2\lambda}$ .  $\square$

**Lemma 6.** *If the underlying NIZK protocol is online extractable and statistically zero-knowledge,  $2\text{-RNSD}_{n,2n,c}$  and  $2\text{-RNSD}_{n,L+k,c}$  problems are hard, then our construction is unforgeable.*

*Proof.* Assume the underlying NIZK protocol is online extractable. If a PPT adversary  $\mathcal{A}$  breaks the unforgeability of our FDABS scheme with advantage  $\epsilon$ , then we construct a PPT algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  as a subroutine to solve an instance of the  $2\text{-RNSD}_{n,2n,c}$  or  $2\text{-RNSD}_{n,L+k,c}$  problems. We also show that the success probability of  $\mathcal{B}$  is polynomially related to  $\epsilon$ . Since these two problems are assumed to be hard, it then follows  $\epsilon$  is negligible. This will complete the proof.

Given a matrix  $\mathbf{B} \in \mathbb{Z}_2^{n \times m}$  for the  $2\text{-RNSD}_{n,2n,c}$  problem and matrices  $\mathbf{C}_0 \in \mathbb{Z}_2^{n \times m_0}, \mathbf{C}_1 \in \mathbb{Z}_2^{n \times m_1}$  for the  $2\text{-RNSD}_{n,L+k,c}$  problem,  $\mathcal{B}$  simulates the experiment  $\text{Expt}_{\mathcal{A}}^{\text{unforge}}(1^\lambda)$  as follows. It first generates the remaining public parameter as specified in the algorithm  $\text{Setup}_{\text{init}}$ , produces the key pair  $(\text{mpk}, \text{msk})$  for the attribute-issuing authority, initializes the registration table  $\text{reg}$ , system information  $\text{info}$  and several lists  $\text{HL}, \text{BL}, \text{SL}$ . Next, it invokes  $\mathcal{A}$  by sending over  $\text{pp}, \text{mpk}$  and making  $\text{reg}$  and  $\text{info}$  visible to  $\mathcal{A}$ . When  $\mathcal{A}$  makes queries to oracles  $\text{AddHX}, \text{RevealX}, \text{Sign}, \text{Update}$ ,  $\mathcal{B}$  handles them honestly and faithfully, since  $\mathcal{B}$  has all the required information. In our construction,  $\mathcal{A}$  will also query the hash function  $\mathcal{H}_{\text{FS}}$  and  $\mathcal{H}_G$ . For the former, this is managed by replying randomly chosen elements  $r$  from  $\{1, 2, 3\}^\kappa$ . For the latter, this is managed by programming it as a polynomial  $p_G$  over  $\text{GF}(2^T)$  of degree at least  $2q_G - 1$ , where  $q_G$  is an upper bound of all queries to  $\mathcal{H}_G$  and  $T$  is super-logarithmic in  $\lambda$ .

Eventually,  $\mathcal{A}$  outputs a tuple  $(M^*, P^*, \tau^*, \Sigma^*)$ . Suppose the output satisfies all the conditions required in  $\text{Expt}_{\mathcal{A}}^{\text{unforge}}(1^\lambda)$  to return 1. Then  $(M^*, P^*, \tau^*, \Sigma^*)$  is not obtained from oracle queries and  $\text{Verify}(M^*, P^*, \text{info}_{\tau^*}, \Sigma^*) = 1$ .  $\mathcal{B}$  then exploits the forgery as follows.

Let  $\Sigma^* = (\{\text{CMT}_i^*\}_{i=1}^\kappa, \{\overline{\text{RSP}}_{i,j}\}_{i \in [1,\kappa], j \in \{1,2,3\} \setminus \{\text{ch}_i^*\}}, \text{CH}^*, \{\text{RSP}_{i,\text{ch}_i}^*\}_{i=1}^\kappa)$ . Since  $\Sigma^*$  is valid, we run our extractor  $\mathcal{E}_{\text{fdabs}}$  described in Figure 3, obtaining a witness  $\xi^* = (\mathbf{d}^*, \mathbf{x}^*, \mathbf{r}^*, \text{bin}(j^*), \mathbf{w}_\ell^*, \dots, \mathbf{w}_1^*)$  of the form (1) such that

- (i)  $\mathbf{d}^* = \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}^*) \oplus \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}^*)$  and  $\text{wt}(\mathbf{d}^*) = 1 \pmod 2$ ;
- (ii)  $\text{TVerify}_{\mathbf{B}}(\mathbf{u}_{\tau^*}, \mathbf{d}^*, \text{bin}(j^*), (\mathbf{w}_\ell^*, \dots, \mathbf{w}_1^*)) = 1$ ;
- (iii)  $P^*(\mathbf{x}^*) = 1$ .

We consider the following three cases.

Case 1: The attribute corresponding to the  $j^*$ -th leaf is inactive at epoch  $\tau^*$ . (Either this leaf has not been assigned an attribute at all or the assigned attribute has been revoked no later than  $\tau^*$ .) In other words, the real leaf value  $\mathbf{d}_{j^*}$  associated with leaf  $\text{bin}(j^*)$  is  $\mathbf{0}^n$ . Therefore, there are two different paths

starting from leaf  $\text{bin}(j^*)$  to the root  $\mathbf{u}_{\tau^*}$ . Thus, one can find a non-zero vector  $\mathbf{z} \in 2\text{-Regular}(2n, c)$  such that  $\mathbf{B} \cdot \mathbf{z} = \mathbf{0}^n \pmod{2}$ . This solves the  $2\text{-RNSD}_{n,2n,c}$  problem.

Case 2: The attribute corresponding to the  $j^*$  leaf is active at epoch  $\tau^*$  and  $\mathbf{d}_{j^*} \neq \mathbf{d}^*$ . This, however, results in two different paths from the leaf  $\text{bin}(j^*)$  to the root as Case 1 and would solve the  $2\text{-RNSD}(n, 2n, c)$  problem.

Case 3: The attribute corresponding to the  $j^*$  leaf is active at epoch  $\tau^*$  and  $\mathbf{d}_{j^*} = \mathbf{d}^*$ . Let the corresponding attribute and opening of  $\mathbf{d}_{j^*}$  be  $\mathbf{x}_{j^*}$  and  $\mathbf{r}_{j^*}$ , respectively. We distinguish two subcases.

- If  $\mathbf{x}_{j^*} \in \text{BL}$ , i.e.,  $\mathcal{A}$  learns signing key  $\text{sk}_{\mathbf{x}_{j^*}}$ , we claim  $P^*(\mathbf{x}_{j^*}) = 0$ . To prove the claim, first note that for the tuple  $(M^*, P^*, \tau^*, \Sigma^*)$  output by  $\mathcal{A}$ , either  $P^*(\mathbf{x}) = 0$  or  $\text{lsActive}(\mathbf{x}, \text{info}_{\tau^*}) = 0$  for all  $\mathbf{x} \in \text{BL}$ . However,  $\mathbf{x}_{j^*}$  is active at  $\tau^*$ , i.e.,  $\text{lsActive}(\mathbf{x}_{j^*}, \text{info}_{\tau^*}) = 1$ , implies  $P^*(\mathbf{x}_{j^*}) = 0$ . Having both  $P^*(\mathbf{x}_{j^*}) = 0$  and  $P^*(\mathbf{x}^*) = 1$ , one concludes that  $\mathbf{x}_{j^*} \neq \mathbf{x}^*$ . Let  $\mathbf{z} = (\text{RE}(\mathbf{x}^*) \oplus \text{RE}(\mathbf{x}_{j^*}) \parallel \text{RE}(\mathbf{r}^*) \oplus \text{RE}(\mathbf{r}_{j^*})) \in 2\text{-Regular}(L+k, c)$ . Then  $\mathbf{z}$  is non-zero and  $[\mathbf{C}_0 \mid \mathbf{C}_1] \cdot \mathbf{z} = \mathbf{0}^n \pmod{2}$ . This solves the  $2\text{-RNSD}_{n,L+k,c}$  problem.
- If  $\mathbf{x}_{j^*}^* \notin \text{BL}$ , we claim that  $\mathbf{r}_{j^*} \neq \mathbf{r}^*$  with probability at least  $1/2$ . This again solves the  $2\text{-RNSD}_{n,L+k,c}$  problem. To argue  $\mathbf{r}_{j^*} \neq \mathbf{r}^*$  with probability at least  $1/2$ , let us summarize what  $\mathcal{A}$  learns about  $\mathbf{r}_{j^*}$ . In the extreme case,  $\mathcal{A}$  learns  $\mathbf{d}_{j^*}$  and  $\mathbf{x}_{j^*}$  and hence  $\mathbf{C}_1 \cdot \text{RE}(\mathbf{r}_{j^*})$ , which is  $\mathbf{d}_{j^*} \oplus \mathbf{C}_0 \cdot \text{RE}(\mathbf{x}_{j^*})$ . However, by Lemma 5, with overwhelming probability there exists at least another vector  $\mathbf{r}^* \neq \mathbf{r}_{j^*}$  for which  $\mathbf{C}_1 \cdot \text{RE}(\mathbf{r}^*) = \mathbf{C}_1 \cdot \text{RE}(\mathbf{r}_{j^*})$ . In addition, the statistical zero knowledge property of the NIZK protocol reveals at most negligible information about which witness among  $\mathbf{r}_{j^*}$  and  $\mathbf{r}^*$  is used to sign messages. Therefore, with probability at least  $1/2$ ,  $\mathbf{r}_{j^*} \neq \mathbf{r}^*$ . This proves the claim.  $\square$