Blind Signatures from Cryptographic Group Actions

Dung Hoang Duong[1], Xuan Thanh Khuc[1], Youming Qiao[2], Willy Susilo[1], and Chuanqi Zhang[2]

[1] Institute of Cybersecurity and Cryptology, School of Computing and Information Technology, University of Wollongong, Australia.
{hduong,xtkhuc,wsusilo}@uow.edu.au

[2] Centre for Quantum Software and Information, School of Computer Science, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia.
{Youming.Qiao,Chuanqi.Zhang}@uts.edu.au

Abstract. We provide a generic construction of blind signatures from cryptographic group actions following the framework of the blind signature CSI-Otter introduced by Katsumata et al. (CRYPTO'23) in the context of isogeny (commutative group action). We adapt and modify that framework to make it work even for non-commutative group actions. As a result, we obtain a blind signature from abstract group actions which are proven to be secure in the random oracle model. We also propose an instantiation based on a variant of linear code equivalence, interpreted as a symmetric group action.

Keywords: blind signature · group actions · square root

1 Introduction

Blind signature, introduced by Chaum [26] in 1982, is an interactive protocol between a signer, who holds a secret key, and a user, who holds a message, to jointly create a signature on a message in such a way that the message is oblivious to the signer at the signing time. Blind signatures have found many applications such as in e-cash [26, 28], in e-voting [27, 49], and in blockchains [23, 41, 64], and much more; see [37] and references therein for a rich list of applications and references.

One approach to construct a blind signature is to design a Schnorr-like sigma protocol [29] (or identification scheme) which has *module* structures [39] enabling the randomization of the interaction. The Schnorr blind signature was generalized by Pointcheval and Stern [57] and Abe and Okamoto [2]. The security proof of Abe and Okamoto contained a bug that has recently been fixed by Kastner, Loss and Xu [44] who provided a generic proof for Abe-Okamoto style blind signature. At CRYPTO'23, Katsumata et al. [46] proposed the first isogeny-based blind signature in the context of *cryptographic group actions*, called CSI-Otter, inspired by the Abe-Okamoto's construction in which Katsumata et al. utilized

the *quadratic twist* of an elliptic curve in a clever way to endow isogenies with richer structure than abstract group actions, but still weaker than module structures, that enables the blindness. The security proof is hence followed from the framework by Kastner et al. [44].

Cryptographic group actions were first introduced by Brassard and Yung [21] in the context of *one-way* group actions. It was then considered independently by Couveignes [31] in the context of *hard homogeneous spaces* and by Rostovsev and Stolbunov [59] in the context of isogeneous elliptic curves. This line of research was largely ignored until the proposal of CSIDH by Castryck et al. [24] in which the authors considered supersingular elliptic curves defined over a large prime field, rather than to ordinary elliptic curves as in the previous work of Couveignes [31] and Rostovsev-Stolbunov [59], on which most of efficient isogeny-based constructions are based, such as CSI-FiSh signature [14], threshold signature [36], ring signatures [13], group signature [12] and blind signature CSI-Otter [46].

In the context of non-commutative group actions, there have been several proposals that submitted to NIST's recent call for additional post-quantum signatures[3], including MEDS [30], LESS [16], and ALTEQ [62], whose underlying groups are either general linear group $\mathrm{GL}(n, q)$ (for the cases of MEDS and ALTEQ) or monomial matrix group $\mathrm{Mon}(n, q)$ (for the case of LESS). There have been several analogous cryptographic constructions to the case of isogenies in this context, such as (inefficient) threshold signature [8] and ring signatures [7, 17]. However, due to the non-commutativity of the underlying groups, the cryptographic constructions in this setting are still limited. For example, public key encryptions based on non-commutative group actions are only recently shown with *quantum ciphertexts* [42].

Even though non-commutative group action constructions are less efficient than the isogeny counterparts in terms of key/signature sizes, those schemes however enjoy the efficiency in terms of implementation. Furthermore, actions by "highly non-commutative" groups, such as symmetric and general linear groups, enjoy the property that most known quantum algorithmic techniques do not work for hidden subgroup problems for such groups [38]. These make non-commutative group actions an appealing candidate for post-quantum cryptography, so it is desirable to develop more advanced cryptographic schemes to increase cryptographic functionalities based on them. In particular, it brings to us the following question:

Can we construct a blind signature from non-commutative group actions?

1.1 Our Contribution

In this paper, we provide an affirmative answer to the above question. Our contribution in this paper is two-fold and can be summarized as follows.

[3] https://csrc.nist.gov/news/2023/additional-pqc-digital-signature-candidates

– We provide a framework to construct a Schnorr-type blind signature from abstract group actions. Our framework follows closely with the construction of CSI-Otter [46] with modifications to adjust for the case of generic groups. In particular, because we do not have *twists* as in the case of elliptic curves, we need to *double* the public key, compared to that of CSI-Otter, in such a way that the *additional* public key element plays a twist role in our context; see Section 1.2 for more detail. Another contribution in this fold is a zero-knowledge proof for well-formed public key. In contrast to the case of isogenies, one can easily verify that the public key is valid, i.e., the public key is indeed a valid supersingular elliptic curve. In our setting, we need a zero-knowledge protocol allowing one to validate the public key.
– We provide an efficient instantiation from non-commutative group actions[4]. In order to provide an instantiation for non-commutative group actions, we require several conditions for the underlying group. If the group is non-commutative, in order to ensure the soundness of our protocol, we need a group with an efficient square-root algorithm. This is because in our protocol, given two accepted transcripts with the same commitment, an extractor can obtain only g^2, where g is the secret key. Hence we need an efficient algorithm to compute g given g^2 in the group. In order to enable an efficient instantiation inherited from existing efficient schemes, we proposed a variant of LESS that instead of having monomial matrix $\mathrm{Mon}(n, q)$ action as in LESS, we have a permutation group action which fulfils our purpose; see details in Section 1.2.

1.2 Technique Overview

In this section, we present in detail our contributions. We will first describe the core in the construction of CSI-Otter [46] on which our framework is based. We then present a variant of linear code equivalence problem in LESS from which we provide an instantiation for a blind signature following our framework.

Construction Framework. In CSI-Otter [46], the authors consider the CSIDH group action $* : G \times \mathcal{E} \to \mathcal{E}$ where G is an ideal class group and $\mathcal{E}$ is a set of elliptic curves. It can be assumed that the structure of G is known and we can express G as $G = \langle \mathfrak{g} \rangle \cong \mathbb{Z}_N$ for some positive integer $N \in \mathbb{N}$ and generator $\mathfrak{g} \in G$ [14]. In isogeny settings, an elliptic curve $E_0 \in \mathcal{E}$ is fixed and the public key is of the form $A = [\mathfrak{g}^a] * E_0$ for a random $a \leftarrow\!\!\$\ \mathbb{Z}_N$, and the first-sender message (i.e., commitment) is computed similarly as $Y = [\mathfrak{g}^y] * E_0$ for a random $y \leftarrow\!\!\$\ \mathbb{Z}_N$. In order to enable a Schnorr-type blind signature, the normal procedure for the user in blinding the message M would be: (i) randomize the commitment, which can be done by computing $[\mathfrak{g}^z] * Y$ for $z \leftarrow\!\!\$\ \mathbb{Z}_N$; (ii) randomize the public key A and public parameter E_0, which can be done by computing $[\mathfrak{g}^b] * A$ and $[\mathfrak{g}^d] * E_0$ for $d, b \leftarrow\!\!\$\ \mathbb{Z}_N$; (iii) associate $[\mathfrak{g}^z] * Y$, $[\mathfrak{g}^b] * A$ and $[\mathfrak{g}^d] * E_0$ into one element,

[4] We note that our framework can be instantiated for generic groups, both commutative and non-commutative.

say X; (iv) compute the hash value $c = H(X\|M)$; and lastly (v) use z, d, b to randomize c to obtain a randomized challenge c' and send to the signer. In the discrete logarithm setting [29], all steps (i)–(v) can be done easily, especially step (iii) since $[\mathfrak{g}^z] * Y$, $[\mathfrak{g}^b] * A$ and $[\mathfrak{g}^d] * E_0$ are all group elements on which we can do operation to create a group element X. However, it is not the case for isogeny setting since $[\mathfrak{g}^z] * Y$, $[\mathfrak{g}^b] * A$ and $[\mathfrak{g}^d] * E_0$ are all elliptic curves and we do not have operations on elliptic curves.

In order to overcome that problem, Katsumata et al. [46] has cleverly used *quadratic twist* in elliptic curves. Briefly speaking, given $A = [\mathfrak{g}^a] * E_0$ for an unknown $a \in \mathbb{Z}$, every one can easily compute its quadratic twist $[\mathfrak{g}^{-a}] * E_0$, which was denoted by A^{-1} in [46]. Now step (i)–(iii) above can be done together in [46] as follows: choose $(d, z) \leftarrow\!\!\$\ \{-1, 1\} \times \mathbb{Z}_N$ and set $X := [\mathfrak{g}^z] * Y^d$. For the proof to work, following the proof by Kastner-Loss-Xu [44] for Abe-Okamoto blind signature [2], Katsumata et al. [46] modified the above idea to use the OR composition of the underlying sigma protocol. Specifically, CSI-Otter uses two public key $A_0 = [\mathfrak{g}^a] * E_0, A_1 = [\mathfrak{g}^b] * E_0$ where $a, b \leftarrow\!\!\$\ \mathbb{Z}_N$, and the secret key is one of the a_0 or a_1; see [46] for the details.

Since quadratic twists exist only in the isogeny setting, in order to enable such a CSI-Otter-like construction for abstract group actions, what we do is to *double* the public key. To be more precise, consider a group G acting on a set S by $* : G \times S \to S$ and fix an element $E_0 \in S$.[5] Our public key will consist of $A_b^{(c)} = g_b^c * E_0$ for $b \in \{0, 1\}, c \in \{-1, 1\}$ and the secret key is either $g_0 \in G$ or $g_1 \in G$. Here $A_b^{(-1)}$ will play the role for quadratic twists of $A_b^{(1)}$ as in the case of CSI-Otter. We also construct a protocol for an OR relation (cf. Figure 3) as in CSI-Otter as the underlying sigma protocol for the blind signature. Our blind signature follows the same route as in CSI-Otter but with modifications; see Section 4 for the detail. We highlight below two note-worthy differences between our scheme with CSI-Otter:

- Firstly, in our scheme, one of the responses is of the form $r = hg_\delta^{-1} \in G$ where g_δ with $\delta \in \{0, 1\}$ is the secret key. For the verification, we need to compute the action on $A_\delta^{(-1)} = g_\delta^{-1} * E_0$ and expect the outcome to be one of the commitment $Y := h^{-1} * E_0$.
  In this case, if we use $r^{-1} = g_\delta h^{-1}$ then we need g and h^{-1} commute. However, requiring h^{-1} to commute with g would break the HVZK property of the underlying sigma protocol.
  Therefore, instead of having one response hg_δ^{-1}, we send two responses hg_δ^{-1} and $h^{-1}g_\delta$. This turns out to be useful in restoring HVZK.
- Secondly, in order to ensure the soundness of the underlying sigma protocol in our scheme, the extractor can obtain, from two given accepted transcripts with the same commitment, the square g_δ^2 of the secret key g_δ. Hence, we need an efficient square-root algorithm in the group G to compute g_δ from g_δ^2.

[5] We use the same notation as in CSI-Otter to make readers easily follow the flows of the construction.

Another contribution is a zero-knowledge protocol to validate the public key. In contrast to the case of isogeny-based cryptography in which everyone can easily validate the public key – a valid supersingular elliptic curve, it is not the case for abstract group actions. Our protocol is presented in Fig. 1.

Instantiation. As mentioned above, to instantiate our blind signatures with non-commutative group actions, we require the corresponding group G to satisfy the following:

- Given g^2, there exists an efficient algorithm to compute g.
- Reusing $g \in G$ twice still gives a secure protocol (see Definition 4 for a more precise requirement).

To identify a non-commutative group action satisfying the above seems a tricky business.

For MEDS and ALTEQ, the underlying group is $\mathrm{GL}(n, q)$. For $\mathrm{GL}(n, q)$, the matrix square-root problem was addressed in [10], but the algorithm there requires going to an extension field. It is an interesting problem to devise a matrix square root algorithm without going to the extension field. This seems to be the only bottleneck for using the group actions underlying MEDS and ALTEQ, because the IGAP for these actions seem still hard.

For LESS, the underlying group is the monomial group $\mathrm{Mon}(n, q)$, consisting of matrices in $\mathrm{GL}(n, q)$ with each row and each column having exactly one non-zero entry. While square root can be efficiently done in this group, reusing group elements there is risky, and for certain parameters, using the same group element twice can result in an insecure protocol [22].

To address the above issues, we interpret the monomial code equivalence problem as a group action of the symmetric group. This is made possible by a canonical form algorithm (Section 5.3). In contrast to LESS, where the group action is by the monomial group, our group actions avoid the attacks on reusing group elements as in [22] (Section 5.1). We further show that by selecting a family of permutations that satisfy the square-root requirement (Section 5.2).

1.3 Related Work

Most of the existing post-quantum blind signatures are constructed from lattices. The first post-quantum blind signature was proposed by Rückert [60] following the design paradigm by Pointcheval and Stern [57]. However, Hauck et al. [40] dicovered a flaw in Rückert's security argument which results in many blind signatures [5, 20, 51, 53] following the design and security arguments of Rückert [60] being insecure. Hauck et al. [40] also introduced a new blind signature from linear hash functions [39] but it is impractical. Lyubashevsky et al. [52], del Pino-Katsumata [56], and Agrawal et al. [3] respectively proposed efficient two round lattice-based blind signatures, which was further improved by Beullens et al. [15] with a two round blind signature from standard lattice problems with signature size around 22 KB.

In another context, Petzoldt et al. [54] constructed a blind signature from multivariate quadratic equations. However, it has been recently broken by Beullens [11]. Blazy et al. [19] proposed a blind signature from codes but it had a flaw in the security proof, which was later fixed [18].

Recently, Katsumata et al. [46] proposed the first construction of a blind signature from isogenies with the signature size of around 8 KB for the basic scheme and 4 KB for the optimized version. In this paper, we generalize that result to abstract group actions.

Concurrent work. Recently, Kuchta, LeGrow and Persichetti proposed a construction of blind signatures from matrix code equivalence [50]. In [50], the framework also follows that in [46], with a focus on matrix code equivalence. To resolve issues caused by non-commutativity, the authors of [50] make use of the actions of both A and its inverse transpose A^{-T} for an invertible matrix A, and require A to be (anti)symmetric. The security of the scheme relies on the hardness assumption of the Modified Inverse Matrix Code Equivalence Problem (MIMCE), a computational version of Inverse Matrix Code Equivalence Problem (IMCE). IMCE was recently attacked in [22], and this attack was not discussed in [50].

In contrast, our framework is applicable for general (non-commutative) group actions, with security based on assumptions such as in Definition 16. One requirement for instantiating our scheme is the reuse of secret keys, which leads us to propose the use of symmetric group action to equivalence classes of linear codes under the action of general linear group and diagonal group. This viewpoint helps to thwart the attack of reusing keys as in [22]; see Section 5 for the details.

1.4 Discussion about ROS-related Attacks

A blind signature is required to satisfy two security properties: *blindness* and *one-more unforgeability*. Briefly speaking, blindness means that the signer is unable to know the message that he/she signs (message is blinded), and one-more unforgeability means that the malicious user cannot output $l+1$ or more valid signatures after finishing l signing sessions with the signer; see Section 2.4 for the formal definition. In particular, if we allow a malicious user to concurrently open l signing sessions, i.e., open l signing queries in parallel, we use the term *l-concurrent unforgeability*; otherwise we use the term *sequential unforgeability*.

In [61], Schnorr introduced the Random inhomogeneous in an Overdetermined Solvable system of linear equations (a.k.a. ROS_l) problem in dimension l and showed that an algorithm for ROS_l can be used to break the l-concurrent unforgeability of the Schnorr signature. Recently, Benhamouda et al. [9] proposed a polynomial time algorithm for ROS_l with $l = \mathrm{poly}(\lambda)$ where λ is the security parameter, which hence breaks the Schnorr blind signature: for $l = 128$, it only takes time roundly 2^{32} hash computations to break unforgeability. An implication of Benhamouda et al. [9] is that a Schnorr-type blind signature is

not concurrently unforgeable for l larger than polylogarithmic in the security parameter. Because the lack of algebraic structures in isogeny setting, it is unclear yet how to apply the attack by Benhamouda et al. [9] to CSI-Otter as left open in [46].

Recently, Katsumata et al. [47] and Do et al. [33] independently solve that open problem by proposing a polynomial time attack against the l-concurrent unforgeability of Schnorr-type blind signatures. In [47], Katsumata et al. proposed a parallel ROS problem and proved that it is solvable in polynomial time for appropriate parameters. As a consequence, they are able to break the l-concurrent unforgeability of CSI-Otter for $l = \mathrm{poly}(\lambda)$: for $l = 4$, it takes only in time roundly 2^{34} hash computations. in [33], Do et al. proposed a generic attack that does not require any algebraic structures and can be applicable to all blind signatures built from a sigma protocol with small challenge space. In particular, if the underlying sigma protocol's challenge space is $\mathcal{C}$ and the protocol needs to repeat k times to attain the required security level, then Do et al.'s attack [33] can break the l-concurrent unforgeability (for $l \geq k$) of the corresponding blind signature in time $O(k \cdot |\mathcal{C}|)$. So for CSI-Otter with $k = 128$ and $|\mathcal{C}| = 2$, Do et al.'s attack can break the concurrent unforgeability after 128 concurrent signing sessions for the basic attack and with only 8 sessions in the optimized attack. Do et al. [33] also introduced some countermeasures, such as double the k (from 128 to 256) or using other techniques from [25, 48], but also mentioned that those will result in inefficient schemes compared to existing lattice ones. Katsumata et al. [47] suggested considering other techniques [1, 45, 63] used in classical settings, which is an interesting open problem.

Our proposed blind signature in this paper follows the same framework of CSI-Otter, and hence it is vulnerable to Do et al.'s attack [33]. We suspect that the attack by Katsumata et al. [47] may be applicable. We will leave it as a future work to explicitly investigate the attack by Katsumata et al. [47] against our scheme. Our paper shows the feasibility of a CSI-Otter-like blind signature construction for abstract group actions, and a fix for CSI-Otter, as suggested by Katsumata et al. [47], probably also yield a fix for our scheme against those attacks, which will leave as a future investigation.

Acknowledgment

We would like to thank Lucjan Hanzlik for his insightful comments and helpful suggestions.

2 Preliminaries

2.1 Notations

For a prime power q, let $\mathbb{F}_q$ be the field consisting of q elements. Denote by $\mathbb{F}_q^n$ the linear space of length-n row vectors over $\mathbb{F}_q$. Denote by $\mathrm{Mat}(m \times n, q)$ the linear space of $m \times n$ matrices over $\mathbb{F}_q$, and $\mathrm{Mat}(n, q) := \mathrm{Mat}(n \times n, q)$.

We use $\mathrm{GL}(n, q)$ to denote the group of $n \times n$ invertible matrices over $\mathbb{F}_q$, and $\mathrm{D}(n, q)$ to denote the group of $n \times n$ invertible diagonal matrices over $\mathbb{F}_q$. The symmetric group on $\{1, \ldots, n\}$ is denoted by S_n. By encoding each $\sigma \in \mathrm{S}_n$ as a $n \times n$ permutation matrix over $\mathbb{F}_q$, we can embed S_n as a subgroup of $\mathrm{GL}(n, q)$. A matrix in $\mathrm{Mat}(n, q)$ is said to be monomial, if it is the product between a diagonal and a permutation matrix. The group of monomial matrices is denoted by $\mathrm{Mon}(n, q)$.

For a positive integer k, we denote $[k]$ to be the set $\{1, \ldots, k\}$. For a vector $\overrightarrow{h}$, denote by h_i the i-th entry of $\overrightarrow{h}$. We will also denote a vector by bold character, e.g., $\mathbf{h}$. For a finite set S, we write $x \leftarrow\!\!\$\ S$ to denote x is sampled randomly from S. We use $\odot$ to denote the component-wise multiplication of vectors in $\mathbb{R}$. In particular, for $c \in \mathbb{R}$ and vectors $\mathbf{a} = (a_1, \cdots, a_k), \mathbf{b} = (b_1, \ldots, b_k)$, we write $c \odot \mathbf{a}$ for $(ca_1, \ldots, ca_k)$ and $\mathbf{a} \odot \mathbf{b} = (a_1 b_1, \ldots, a_k b_k)$. We also extend this component-wise notation for exponentiation, e.g., we write $\mathbf{a}^c$ for $(a_1^c, \ldots, a_k^c)$, $\mathbf{a}^{\mathbf{b}}$ for $(a_1^{b_1}, \ldots, a_k^{b_k})$, and for group action, e.g., we write the action of vector $\mathbf{a}$ on $\mathbf{s} \in S^k$ as $\mathbf{a} * \mathbf{s}$ for $(a_1 * s_1, \ldots, a_k * s_k)$ (here $*$ indicates the action operation - see Section 2.3 for group action definition).

2.2 Sigma Protocols

Definition 1 (Sigma Protocol). *A sigma protocol for an NP relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is a public-coin three-move interactive protocol between a prover $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2)$ and a verifier $\mathcal{V}$ as the following.*

- *The prover on input a statement X and a witness W such that $(\mathsf{X}, \mathsf{W}) \in R$, runs $(\mathsf{com}, \mathsf{state}) \leftarrow \mathcal{P}_1(\mathsf{X}, \mathsf{W})$ and sents a commitment com to the verifier.*
- *The verifier samples a random challenge $\mathsf{ch} \leftarrow\!\!\$\ \mathcal{C}$ from the challenge space $\mathcal{C}$ and sends ch to the prover.*
- *Upon receiving the challenge ch, the prover $\mathcal{P}_2$ generates a response rsp and sents rsp to the verifier.*
- *The verifier runs $\mathcal{V}(\mathsf{X}, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ and outputs 1 to indicate acceptance, and 0 otherwise.*

A sigma protocol must satisfy correctness, honest-verifier zero-knowledge (HVZK), and special soundness defined as the following.

Correctness. It is required that if the prover $\mathcal{P}$ and the verifier $\mathcal{V}$ follow the sigma protocol honestly, then the verifier would output 1 with probability 1.

Honest Verifier Zero-Knowledge (HVZK). There exists a PPT simulator Sim that given a statement X, a challenge $\mathsf{ch} \in \mathcal{C}$, outputs a valid transcipt $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ that is indistinguishable from a real transcript.

Special Soundness. There exists a determinsitic polynomial time extractor Ext that given two accepted transcripts $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ and $(\mathsf{com}, \mathsf{ch}', \mathsf{rsp}')$ with the same commitment com and different challenges $\mathsf{ch} \neq \mathsf{ch}'$, outputs W such that $(\mathsf{X}, \mathsf{W}) \in R$.

We also provide a definition for a hard instance generator for the NP relation R as follows.

Definition 2 (Hard Instance Generator). *An NP relation R is associated with an instance generator (IG) if IG, given as input the security parameter 1^n, outputs a statement-witness pair $(\mathsf{X}, \mathsf{W}) \in R$. Moreover, we say that the instance generator is hard if the following holds for any PPT adversary $\mathcal{A}$:*

$$\Pr[(\mathsf{X}, \mathsf{W}) \leftarrow \mathsf{IG}(1^n), \mathsf{W}' \leftarrow \mathcal{A}(\mathsf{X}) : (\mathsf{X}, \mathsf{W}') \in R] = \mathrm{negl}(n).$$

2.3 Cryptographic Group Actions

Let G be a group and S a set. An action of G on S is a map $* : G \times S \to S$ satisfying the following properties: (i) $\mathrm{id} * s = s$ for all $s \in S$ and the identity element $\mathrm{id} \in G$; and (ii) $g * (h * s) = gh * s$ for all $g, h \in G$ and $s \in S$. A group action is said to be [4]:

- *transitive* if for all $s, t \in S$, there exists $g \in G$ such that $g * s = t$;
- *faithful* if there does not exist $g \in G \setminus \{\mathrm{id}\}$ such that $g * s = s$ for all $s \in S$, i.e., if $g * s = s$ for all $s \in S$ then $g = \mathrm{id}$;
- *free* if whenever there exists $s \in S$ such that $g * s = s$ then $g = id$; and
- *regular* if it is free and transitive.

Given a group action $*$ of G on S, the orbit of an element $s \in S$ is defined as $\mathsf{Orb}(s) := \{g * s : \forall g \in G\}$. Note that if the group action is transitive then $\mathsf{Orb}(s) = G$. The stabilizer of s is defined by $\mathsf{Stab}(s) := \{g \in G : g * s = s)\}$ which is a subgroup of G. The Orbit-Stabilizer theorem says that, if G is finite then $|G| = |\mathsf{Stab}(s)| \cdot |\mathsf{Orb}(s)|$.

In this paper we shall mostly consider finite groups acting on finite sets. To use group actions in algorithms, we assume that group and set elements have natural encodings, as well as group operations, group actions, and random samplings of group and set elements can be efficiently computed; see [4, 21, 43] for more details and certain variations. In particular, we assume that uniform random samplings from the group G and the set S are efficient.

A group action is *one-way*, if for a random s, the function $f_s : G \to S$ defined by $f_s(g) := g * s$ is one-way. The one-way assumption is formulated as the Group Action Inverse Problem (GAIP) defined in the following.

Definition 3 (GAIP). *Given a group action $* : G \times S \to S$, uniformly random $s \in S$, and uniformly random $t \in \mathsf{Orb}(s)$, find $g \in G$ such that $g * s = t$.*

Here we restrict to the case of transitive group actions, as in the isogeny-based setting [24, 31, 35], or we can restrict the element g to be in the orbit Orb(s) of s as in the case of non-commutative group actions [16, 30, 62].

For the purpose of our paper, we define what we call the Inverse Group Action Problem (IGAP), as follows.

Definition 4 (IGAP). *Given a group action $* : G \times S \to S$, uniformly random $s \in S$, and a pair $(g * s, g^{-1} * s)$, find g.*

IGAP was called Inverse Linear Code Equivalence (ILCE) problem [7] in the context of linear code equivalence underlying LESS, and Inverse Matrix Code Equivalence (IMCE) problem [30] in the context of matrix code equivalence underlying MEDS. In a recent work, Budroni et al. [22] introduced an efficient algorithm for ILCE, but it is unclear yet the impact for IMCE or IGAP for the case of alternating trilinear forms underlying ALTEQ [62].

In this paper, we provide a generic framework of a blind signature for abstract group actions. For the framework to work, an instantiated group G needs to satisfy the following assumptions.

Assumption 1 (Square-Root Assumption)**.** Given g^2, there exists an efficient algorithm to output g.

Asssumption 1 is necessary for proving the soundness of the underlying protocol. In fact, in such a proof, given two accepted transcripts with the same commitment, our extractor can only output g^2, where g is a secret key. Hence we need an efficient algorithm to compute g from g^2.

Note that g^2 may have several square-roots, namely there exist g and g' such that $g^2 = g'^2$. There are two possible workarounds for this. The first one is to require the square-root algorithm to output all square-roots. The second one is to restrict to those group elements with unique square-roots, and this is indeed achievable for our instantiation (see Remark 2).

It is noted that commutative groups trivially satisfy Assumption 1. For the case of non-commutative groups, some discussions on some groups supporting MEDS, ALTEQ, and LESS can be found at the end of Section 1.2.

2.4 Blind Signatures

We follow [2, 44, 45, 46] to define a three-move blind signature.

Definition 5 (Blind Signature). *A three-move blind signature BS with efficient decidable public key space $\mathcal{PK}$ consists of the following PPT algorithms.*

- $\mathsf{BS.KGen}(1^n) \to (\mathsf{pk}, \mathsf{sk})$*: On input the security parameter 1^n, the key generation algorithm outputs a pair of public and secret keys $(\mathsf{pk}, \mathsf{sk})$.*
- $\mathsf{BS.S} = (\mathsf{BS.S}_1, \mathsf{BS.S}_2)$*: The signer consists of two phases:*
  - $\mathsf{BS.S}_1(\mathsf{sk}) \to (\mathsf{state}_S, \rho_{S,1})$*: On input the secret key, outputs an internal signer state state_S and the first-sender message $\rho_{S,1}$.*
  - $\mathsf{BS.S}_2(\mathsf{state}_S, \rho_U) \to \rho_{S,2}$*: On input the signer state state_S and a user message ρ_U, outputs a second-sender message $\rho_{S,2}$.*
- $\mathsf{BS.U} = (\mathsf{BS.U}_1, \mathsf{BS.U}_2)$*: The user consists of two phases:*
  - $\mathsf{BS.U}_1(\mathsf{pk}, M, \rho_{S,1}) \to (\mathsf{state}_U, \rho_U)$*: On input the public key pk, a message M and the first-sender message $\rho_{S,1}$, outputs an internal user state state_U and a user message ρ_U.*
  - $\mathsf{BS.U}_2(\mathsf{state}_U, \rho_{S,2})$*: On input a user state state_U and a second-signer message $\rho_{S,2}$, outputs a signature σ*
- $\mathsf{BS.Verify}(\mathsf{pk}, M, \sigma)$*: On input the public key pk, a message M and a signature σ, it outputs 1 to indicate the signature is valid, and 0 otherwise.*

We require that a blind signature be complete, blind against the malicious signer, and satisfy one-more unforgeability, as defined below.

Definition 6 (Correctness). *A three-move blind signature scheme BS is correct if for all public and secret key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{BS.KGen}(1^n)$, we have*

$$\Pr\left[\mathsf{BS.Verify}(\mathsf{pk}, M, \sigma) = 1 \;\middle|\; \begin{array}{c} (\mathsf{state}_S, \rho_{S,1}) \leftarrow \mathsf{BS.S}_1(\mathsf{sk}) \\ (\mathsf{state}_U, \rho_U) \leftarrow \mathsf{BS.U}_1(\mathsf{pk}, M, \rho_{S,1}) \\ \rho_{S,2} \leftarrow \mathsf{BS.S}_2(\mathsf{state}_S, \rho_U) \\ \sigma \leftarrow \mathsf{BS.U}_2(\mathsf{state}_U, \rho_{S,2}) \end{array}\right] = 1.$$

Definition 7 (Blindness under Chosen Keys). *For a blind signature BS, define the blindness game $\mathsf{Blind}_{\mathsf{BS}}$ with an adversary $\mathcal{A}$ (playing the signer) as follows.*

Setup. *The challenger samples a bit $\mathsf{coin} \leftarrow\!\!\$\ \{0,1\}$ and runs $\mathcal{A}$ on input 1^n.*

Online Phase. *$\mathcal{A}$ outputs two message M_0^* and M_1^*, a public key $\mathsf{pk} \in \mathcal{PK}$, the game checks if pk is valid and if so, it assigns $(M_0, M_1) = (M^*_{\mathsf{coin}}, M^*_{1-\mathsf{coin}})$. If pk is not valid, the game aborts and outputs 0. The adversary $\mathcal{A}$ is given access to oracles $\mathsf{U}_1, \mathsf{U}_2$ which behave as follows.*

- **Oracle U_1.** *On input $b \in \{0,1\}$ and a first-signer message $\rho_{S,1,b}$, if the session b is not yet open, the oracle marks session b as open and generates a state and a challenge as $(\mathsf{state}_{U,b}, \rho_{U,b}) \leftarrow\!\!\$\ \mathsf{BS.U}_1(\mathsf{pk}, M_b, \rho_{S,1})$. It returns $\rho_{U,b}$ to $\mathcal{A}$.*
- **Oracle U_2.** *On input $b \in \{0,1\}$ and a second-signer message $\rho_{S,2,b}$, if the session b is opened, the oracle creates a signature $\sigma_b \leftarrow \mathsf{BS.U}_2(\mathsf{state}_{U,b}, \rho_{S,2,b})$. It marks session b as closed. Oracle U_2 does not output anything.*

Output Determination. *When both sessions are closed and for $b \in \{0,1\}$ we have that $\mathsf{BS.Verify}(\mathsf{pk}, M_b, \sigma_b) = 1$, the oracle returns the two signatures $(\sigma_{\mathsf{coin}}, \sigma_{1-\mathsf{coin}})$ to $\mathcal{A}$, where note that σ_{coin} (resp. $\sigma_{1-\mathsf{coin}}$) is a valid signature for M_0^* (resp. M_1^*) regardless of the choice of coin. $\mathcal{A}$ outputs a guess coin^* for coin. We say that $\mathcal{A}$ wins if $\mathsf{coin}^* = \mathsf{coin}$.*

We say that BS is blind under chosen keys if the probability that $\mathcal{A}$ wins is negligible.

Definition 8 (One-More Unforgeability). *For a blind signature BS and $l \in \mathbb{N}$, we define l-one-more unforgeability via the following game between a challenger and an adversary $\mathcal{A}$:*

Setup. *The challenger samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{BS.KGen}(1^n)$ and runs $\mathcal{A}$ on input pk. It initializes $l_{\mathsf{closed}} = 0$ and $\mathsf{opened}_{\mathsf{sid}} = \mathsf{false}$ for all $\mathsf{sid} \in \mathbb{N}$.*

Online Phase. *$\mathcal{A}$ is given access to two oracle S_1 and S_2 as follows.*

- **Oracle S_1:** *The oracle samples a fresh session identifier sid. It sets $\mathsf{opened}_{\mathsf{sid}} \leftarrow \mathsf{true}$ and generates $(\mathsf{state}_{S,\mathsf{sid}}, \rho_{S,1}) \leftarrow \mathsf{BS.S}_1(\mathsf{sk})$. It then returns sid and the first-sender message $\rho_{S,1}$ to $\mathcal{A}$.*

Oracle $\mathbf{S}_2$: *On input a user message ρ_U and a session identifier sid, if $l_{\mathsf{closed}} \geq l$ or $\mathsf{opened}_{\mathsf{sid}} = \mathsf{false}$, then it returns $\perp$. Otherwise, it increments l_{closed} and $\mathsf{opened}_{\mathsf{sid}} = \mathsf{false}$. It then computes the second-signer message $\rho_{S,2} \leftarrow \mathsf{BS.S}_2(\mathsf{state}_{S,\mathsf{sid}}, \rho_U)$ and returns $\rho_{S,2}$ to $\mathcal{A}$.*

Output Determination. *When $\mathcal{A}$ outputs distinct tuples of message-signature pairs $(M_1, \sigma_1), \ldots, (M_k, \sigma_k)$, we say that $\mathcal{A}$ wins if $k \geq l_{\mathsf{closed}} + 1$ and for all $i \in [k]$, $\mathsf{BS.Verify}(\mathsf{pk}, M_i, \sigma_i) = 1$.*

We say that the blind signature BS is l-one-more unforgeable if the probability that $\mathcal{A}$ wins is negligible.

2.5 Proof Techniques for Blind Signatures

In this section, we briefly present the key idea in proving one-more unforgeability for Schnorr-type signatures from [46], which extracts from the recent work of Kastner, Loss and Xu [44] for the proof of the Abe-Okamoto blind signature [2]. We refer the readers for [46] and references therein for the detailed information. In what follows, we present only key definitions and theorems needed.

Preparation. We assume that the adversary $\mathcal{A}$ against the l-one-more unforgeability game makes only $l+1$ distinct hash queries to the random oracle. In addition, we further assume that the underlying sigma protocol is for the NP OR-relation, i.e., the prover convinces the verifier that he knows one of the two witnesses either W_1 for statement X_1 or W_2 for statement X_2. We also assume that the user-message ρ_U queried to the signing algorithm $\mathsf{BS.S}_2$ satisfies that $\rho_U \in \mathcal{C}$, where $\mathcal{C}$ is the challenge space of the underlying sigma protocol.

Definition 9 (Instances). *Assume that the public key of a Schnorr-type blind signature has exactly two corresponding secret keys $\mathsf{sk}_0 = (0, W_0)$ and $\mathsf{sk}_1 = (1, W_1)$. A $\mathbf{0}$-side (resp. $\mathbf{1}$-side) instance consists of sk_0 (resp. sk_1) and the randomness used by the honest signer algorithm when the secret key is fixed to be sk_0 (resp. sk_1).*

Let $\overrightarrow{h}$ be the vector of responses returned by the random oracle. Note that by the above assumption, $|\overrightarrow{h}| = l+1$. Let rand be the randomness used by the one-more unforgeability adversary. A wrapper $\mathcal{W}$ is a deterministic algorithm that takes as input $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ where $\mathbf{I}$ is an instance. $\mathcal{W}$ will invoke the signer and the adversary on input $\mathbf{I}$ and rand, and use $\overrightarrow{h}$ to answer for hash queries by the adversary. We define $\mathcal{W}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ to output $\perp$ if the adversary aborts or fails to win the one-more unforgeability game. Otherwise, $\mathcal{W}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ outputs whatever the adversary outputs. Denote by $\mathsf{Succ} = \{(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) | \mathcal{W}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \neq \perp\}$ the set of all successful tuples.

Definition 10 (Successful Forking [45]). *Two successful tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are said to fork from each other at index $i \in [l+1]$ if $\overrightarrow{h}_{[i-1]} = \overrightarrow{h}'_{[i-1]}$ but $h_i \neq h'_i$. We denote the set of hash vector pairs (h_i, h'_i) such that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$ fork at index i as $F_i(\mathbf{I}, \mathsf{rand})$.*

Definition 11 (Transcripts [45]). *Consider the wrapper $\mathcal{W}$ running on input $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. The **query transcript**, denoted by $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, is the vector of user message ρ_U queries made to the signing algorithm $\mathsf{BS}.\mathsf{S}_2$ (simulated by $\mathcal{W}$ by the adversary. The **full transcript**, denoted by $\mathsf{trans}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, is the transcript produced between the signer and the adversary.*

Definition 12 (Partners [45]). *Two successful tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are called **partners** at index $i \in [l+1]$ if the following hold:*

- *$(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ fork at index i; and*
- *$\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$.*

We denote by $\mathsf{prt}_i(\mathbf{I}, \mathsf{rand})$ the set of $(\overrightarrow{h}, \overrightarrow{h}')$ such that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are partners at index i.

Definition 13 (Triangles [45]). *A **triangle** at index $i \in [l+1]$ with respect to $\mathbf{I}$, rand is a tuple of three successful tuples in the following set:*

$$\triangle_i(\mathbf{I}, \mathsf{rand}) = \left\{ \begin{array}{l} (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'') \end{array} \middle| \begin{array}{l} (\overrightarrow{h}, \overrightarrow{h}') \in \mathsf{prt}_i(\mathbf{I}, \mathsf{rand}) \\ (\overrightarrow{h}, \overrightarrow{h}'') \in F_i(\mathbf{I}, \mathsf{rand}) \\ (\overrightarrow{h}', \overrightarrow{h}'') \in F_i(\mathbf{I}, \mathsf{rand}) \end{array} \right\}$$

For a triangle $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')) \in \triangle_i(\mathbf{I}, \mathsf{rand})$, we call the pair $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'))$ the base, and $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}''))$ and $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}''))$ the sides.

Definition 14 (Mapping Instances via Transcript [45, 46]). *For a successful tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$, we define $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I})$ as a function that maps a $\mathbf{0}$-side instance $\mathbf{I}$ (resp. $\mathbf{1}$-side instance $\mathbf{I}$) to a $\mathbf{1}$-side instance $\mathbf{I}'$ (resp. $\mathbf{0}$-side instance $\mathbf{I}'$).*

Finally, we define the witness extractor used by the reduction. It was first defined by Kastner, Loss and Xu [45] and later generalized by Katsumata et al. [46]. For the purpose of the paper, we present a variant in the following definition.

Definition 15 (Witness Extraction). *Fix $\mathbf{I}, \mathsf{rand}$ and let $\overrightarrow{h}, \overrightarrow{h}' \in F_i(\mathbf{I}, \mathsf{rand})$ for some $i \in [l+1]$. Moreover, denote by σ, σ' the signatures corresponding to h_i and h_i' respectively. Deterministic algorithms $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ are called **witness extractors** for a function f of $(\mathsf{Ext}_0(\sigma, \sigma'), \mathsf{Ext}_1(\sigma, \sigma')) \in \{(f(\mathsf{sk}_0), \perp), (\perp, f(\mathsf{sk}_1)), (f(\mathsf{sk}_0), f(\mathsf{sk}_1))\}$[6]. For $b \in \{0, 1\}$, we say that the b-side witness can be extracted from $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ at index $i \in [l+1]$ for a function f if $\mathsf{Ext}_b(\sigma, \sigma')$ outputs $f(\mathsf{sk}_b)$.*

[6] In our paper, f is either the identity function (for commutative group actions) or a square function (for non-commutative group actions).

Remark 1. In this paper, we consider only functions f whose inversion is efficiently computable, i.e., given $f(x)$, it is easy to compute x. In fact, in our instantiations and constructions, we consider square function ($f(x) = x^2$) and group G such that computing x from x^2 is easy. Hence, our witness extractors, in fact, can extract the witness sk_0 or sk_1.

We are now ready to describe the idea by Kastner, Loss and Xu [45] for the one-more unforgeability proof of Schnorr-type blind signatures, which was adapted by Katsumata et al. [46] in the case of CSI-Otter.

First of all, if the map $\Phi_{\mathsf{rand},\overrightarrow{h}}$ is a bijection that preserves transcripts for any rand and $\overrightarrow{h}$, then it maps a partner tuple with b-side instance to another partner tuple with $(1-b)$-side instance for the same rand and $\overrightarrow{h}$ ([45, Corollary 1 and Lemma 3]). This implies that the extracted witness from a partner tuple is independent of the reduction's secret key. Hence, what Kastner, Loss and Xu [45] suggested is to use the sides of triangle, rather than the base, to extract a witness with the observation that if a b-site witness can be extracted from the base of a triangle then it can also be extracted from at least one of two sides of the triangle. The reduction then starts with having a b-side witness that hits a corner of the base of a triangle in the first run, then hits the top of the triangle such that it creates side with a $(1-b)$-side witness with probability about $\frac{1}{2}$.

The results by Kastner, Loss and Xu [45] are summarized in the following Lemmas. Lemma 1 shows that the blind signature is perfectly *witness indistinguishable*, while Lemma 2 states that if a witness can be extracted from a base of a triangle, then the same witness can be extracted from at least one of its sides.

Lemma 1 ([45, Lemma 2]). *Fix rand and $\overrightarrow{h}$. For all tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$, $\Phi_{\mathsf{rand},\overrightarrow{h}}$ is a self-inverse bijection and $\mathsf{trans}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \mathsf{trans}(\Phi_{\mathsf{rand},\overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$.*

Lemma 2 ([45, Corollary 3]). *Fix $\mathbf{I}, \mathsf{rand}$ and let $(\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_i(\mathbf{I}, \mathsf{rand})$, for some $i \in [l+1]$. If the $\mathbf{0}$-side ($\mathbf{1}$-side) witness can be extracted from the base $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ of the triangle at index i, then one can also extract the $\mathbf{0}$-side ($\mathbf{1}$-side) witness from at least one of the sides $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ or $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ at index i.*

For the proof of the one-more unforgeability, what we need to do is to make sure our constructions (e.g., $\Phi_{\mathsf{rand},\overrightarrow{h}}$, instances) satisfy those Lemmas, which is stated as follows.

Theorem 1 ([45, Theorem 1], [46, Theorem 3.12]). *Let the blind Schnorr-type signature BS be as defined in the preparation at the beginning of this Section. Assume that the public key consists of two instances of the NP relation generated by the hard instance generator IG and the underlying sigma protocol has challenge space $\mathcal{C}$. If Lemma 1 and Lemma 2 hold, then for all $l \in \mathbb{N}$, if there exist an adversary $\mathcal{A}$ that makes Q hash queries to the random oracle and breaks the*

l-one-more unforgeability of BS with advantage $\epsilon_{\mathcal{A}} \geq \frac{C_1}{|\mathcal{C}|}\binom{Q}{l+1}$*, then there exists an efficient algorithm* $\mathcal{B}$ *that breaks the hard instance generator with advantage* $\epsilon_{\mathcal{B}} \geq C_2 \cdot \frac{\epsilon_{\mathcal{A}}^2}{\binom{Q}{l+1}^2 \cdot (l+1)^3}$ *for some universal positive constants* C_1 *and* C_2*.*

3 Base Sigma Protocols

3.1 Sigma Protocol for Validating Public Key

In contrast to the isogeny setting in CSI-Otter [46] in which checking that a public key (even generated by malicious adversary) is a valid elliptic curve can be done efficiently, it is not the case for generic cryptographic group actions. In this section, we present a base sigma protocol for validating the public key, i.e., in such a protocol, we prove that the public key is well-formed. To be more precise, we provide a sigma protocol for the following relation:

$$R = \{(\mathsf{X} = (A^{(1)}, A^{(-1)}), \mathsf{W} = g) | A^{(b)} = g^b * E, \ \forall b \in \{-1, 1\}\}. \tag{1}$$

Here we consider a group G acting on the set S and fix an element $E \in S$. The protocol is defined as in Fig. 1. It is a variant of the GMW-type protocol for generic group actions (see for example [17]).

$\mathcal{P}(\mathsf{X} = (A^{(1)}, A^{(-1)}), \mathsf{W} = g)$		$\mathcal{V}(\mathsf{X} = (A^{(1)}, A^{(-1)})$
$h \leftarrow\$ G$		
$Y^{(1)} = h * E, Y^{(-1)} = h^{-1} * E$		
$A := h * A^{(-1)}$	$\xrightarrow{(Y^{(1)}, Y^{(-1)}, A)}$	
	$\xleftarrow{c}$	$c \leftarrow\$ \{0, 1\}$
$c = 0 : r = h$		
$c = 1 : r = hg^{-1}$		
	$\xrightarrow{r}$	if $c = 0$ check whether
		$r^b * E = Y^{(b)} \ \forall \ b \in \{-1, 1\}$
		$\wedge \, r * A^{(-1)} = A$
		if $c = 1$ check whether
		$r * A^{(1)} = Y^{(1)} \wedge r * E = A$

Fig. 1: Sigma Protocol for Validating Public Key

Correctness. It is a routine to check the correctness of the protocol. Assume that the prover $\mathcal{P}$ and the verifier $\mathcal{V}$ honestly follow the protocol. For $c = 0$ then $r = h$, and hence $r * E = h * E = Y^{(1)}$, $r^{-1} * E = h^{-1} * E = Y^{(-1)}$, and $r * A^{(-1)} = h * A^{(-1)} = A$ as desired. For the case $c = 1$ then $r = hg^{-1}$, and hence $r * A^{(1)} = hg^{-1} * A^{(1)} = h * E = Y^{(1)}$, and $r * E = hg^{-1} * E = h * A^{(-1)} = A$ as desired.

Special Soundness. Let's consider two accepted transcripts $(Y^{(1)}, Y^{(-1)}, A, c, r)$ and $(Y^{(1)}, Y^{(-1)}, A, c', r')$ with $c \neq c'$. Without loss of generality, we can assume that $c = 0$ and $c' = 1$. Then, since both transcripts are accepted, we have

$$r * E = Y^{(1)} = r' * A^{(1)}$$
$$r * A^{(-1)} = A = r' * E$$

which imply that ${r'}^{-1} r * E = A^{(1)}$ and $r^{-1} r' * E = A^{(-1)}$. Hence if we set $g := {r'}^{-1} r$ then $g * E = A^{(1)}$ and $g^{-1} * E = A^{(-1)}$. It follows that g is a witness for relation R in Equation (1).

Honest Verifier Zero-knowledge (HVZK). Given a statement $\mathsf{X} = (A^{(1)}, A^{(-1)})$ and a challenge $c \in \{0, 1\}$. The simulator Sim works as follows:

- If $c = 0$, sample $r \in G$, and define $Y^{(b)} := r^b * E$ for $b \in \{-1, 1\}$ and $A := r * A^{(-1)}$.
- If $c = 1$, sample $r \in G$, and define $Y^{(1)} := r * A^{(1)}$, $A := r * E$ and samples $Y^{(-1)}$ from the orbit of E.

It is now clear that the simulated transcript $((Y^{(1)}, Y^{(-1)}), A, c, r)$ is accepted by the verifier. Furthermore, for the case of $c = 0$, it is straightforward to see that the simulated and real transcripts are indistinguishable; for the case of $c = 1$, $Y^{(-1)}$ requires special attention. In the simulated transcript, $Y^{(-1)}$ is sampled randomly from the orbit of E; while in the real transcript, $Y^{(-1)} = h^{-1} * E$ is subject to other commitments $Y^{(1)} = h * E$ and $A = hg^{-1} * E$ as defined in Fig. 1. Also, the statements $A^{(1)} = g * E$ and $A^{(-1)} = g^{-1} * E$ are public. The relationship between these elements is illustrated in Fig. 2. Therefore, we need to assume the hardness of the following problem of group action strong decisional Diffie-Hellman (GASDDH), in analogy to the strong decisional Diffie-Hellman assumption in [55, Assumption 2].

Definition 16 (GASDDH). *Given a group action $\alpha : G \times S \to S$ and $s \in S$, distinguish the distributions between $(s, \alpha(g, s), \alpha(g^{-1}, s), \alpha(h, s), \alpha(hg^{-1}, s), \alpha(h^{-1}, s))$ and $(s, \alpha(g, s), \alpha(g^{-1}, s), \alpha(h, s), \alpha(hg^{-1}, s), \alpha(f, s))$ for uniformly random $g, h, f \in G$.*

If no PPT adversary can distinguish the two distributions above, then the simulated and real transcripts for $c = 1$ are clearly indistinguishable as well. In particular, this problem is at least as hard as the following problem of group action inverse decisional Diffie-Hellman (GAIDDH), in analogy to the inverse decisional Diffie-Hellman assumption first studied in [55].

Definition 17 (GAIDDH). *Given a group action $\alpha : G \times S \to S$ and $s \in S$, distinguish the distributions between $(s,\ \alpha(h,s),\ \alpha(h^{-1},s))$ and $(s,\ \alpha(h,s),\ \alpha(f,s))$ for uniformly random $h, f \in G$.*

We claim that the problem of GAIDDH reduces to the problem of GASDDH. Indeed, suppose we have a PPT algorithm to solve GASDDH, then given a group action $\alpha : G \times S \to S$, $s \in S$, and two distributions $(s,\ \alpha(h,s),\ \alpha(h^{-1},s))$ and $(s,\ \alpha(h,s),\ \alpha(f,s))$ for some uniformly random $h, f \in G$, we can treat it as an instance applicable to the algorithm for GASDDH by taking $g = \mathrm{id} = g^{-1}$, where id is the identity element in G. It follows that GAIDDH can also be solved efficiently in this case.

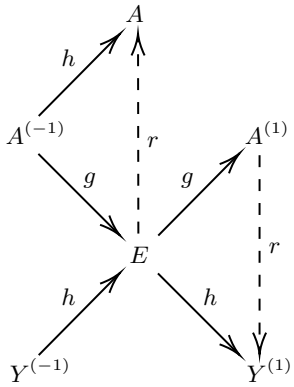

Fig. 2: The relationship between the elements defined in Fig. 1. The group actions on the arrows act on the starting points, producing the endpoints. The dashed arrows only work in the case of $c = 1$, i.e., for $r = hg^{-1}$.

3.2 Base Sigma Protocol for an OR relation

We present the generic version of the sigma protocol in [46] to prove that the prover knows at least one of the two secrets corresponding to the public statement $\mathsf{X} = (A_0^{(1)}, A_0^{(-1)}, A_1^{(1)}, A_1^{(-1)})$, where $A_b^{(c)} = g_b^c * E$ with $b \in \{0,1\}$ and $c \in \{-1,1\}$. The sigma protocol is described in Fig. 3 in which we follow [46] to remove 0 from the challenge space, i.e., our challenge space is now $\{-1,1\}^n$ where n is the security parameter.

Correctness. It is easy to verify the correctness of the protocol. We need to prove that if the Prover $\mathcal{P}$ and Verifier $\mathcal{V}$ follow the protocol honestly then the verifier $\mathcal{V}$ will accept, i.e., we need to verify that $\mathbf{r}_b * A_b^{(\mathbf{c}_b)} = \mathbf{Y}_b\ \forall b \in \{0,1\}$ ($\mathbf{c} = \mathbf{c}_0 \odot \mathbf{c}_1$ is obvious). It is clear for the case $b = 1 - \delta$. For $b = \delta$ we have $\mathbf{r}_\delta = \mathbf{h}_\delta \odot g_\delta^{-\mathbf{c}_\delta}$ and hence

$$\mathbf{r}_\delta * A_\delta^{(\mathbf{c}_\delta)} = \mathbf{h}_\delta \odot g_\delta^{-\mathbf{c}_\delta} * (g_\delta^{\mathbf{c}_\delta} * E) = \mathbf{h}_\delta * E = \mathbf{Y}_\delta$$

as desired.

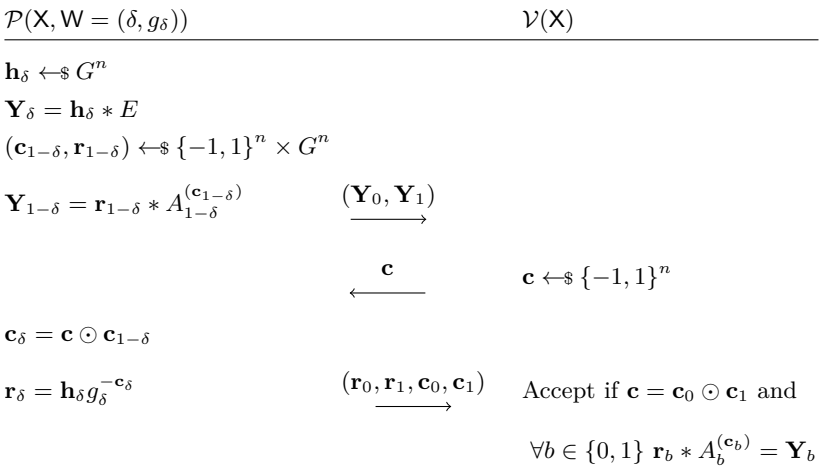

Fig. 3: Base OR Sigma Protocol underlying our Blind Signature

Honest Verifier Zero-knowledge (HVZK). Given a challenge $\mathbf{c} \in \{-1,1\}^n$, a simulator Sim first samples $\mathbf{c}_0, \mathbf{c}_1 \leftarrow\$ \{-1,1\}^n$ subject to $\mathbf{c} = \mathbf{c}_0 \odot \mathbf{c}_1$ and $(\mathbf{r}_0, \mathbf{r}_1) \leftarrow\$ G^{2n}$. Then Sim defines $\mathbf{Y}_b = \mathbf{r}_b * A_b^{(\mathbf{c}_b)}$ for $b \in \{0,1\}$ and outputs the simulated transcript $((\mathbf{Y}_0, \mathbf{Y}_1), \mathbf{c}, (\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1))$ which is indistinguishable from the true transcripts.

Special Soundness. Now, let us consider two accepted transcripts $((\mathbf{Y}_0, \mathbf{Y}_1), \mathbf{c}, (\mathbf{r}_0, \mathbf{r}_1, \mathbf{c}_0, \mathbf{c}_1))$ and $((\mathbf{Y}_0, \mathbf{Y}_1), \mathbf{c}', (\mathbf{r}'_0, \mathbf{r}'_1, \mathbf{c}'_0, \mathbf{c}'_1))$ with $\mathbf{c} \neq \mathbf{c}'$. Without loss of generality, we assume $\mathbf{c}_0 \neq \mathbf{c}'_0$, and so there exists an index $i \in [n]$ such that $c_{0,i} \neq c'_{0,i}$. Then we can obtain $g_0^{c_{0,i} - c'_{0,i}} = r_{0,i}^{-1} r'_{0,i}$. In fact, since two transcripts are valid, we have $r_{0,i} * A_0^{(c_{0,i})} = Y_{0,i} = r'_{0,i} * A_0^{(c'_{0,i})}$ or $r_{0,i} g_0^{c_{0,i}} * E = r'_{0,i} g_0^{c'_{0,i}} * E$ which implies that $g_0^{c_{0,i} - c'_{0,i}} = r_{0,i}^{-1} r'_{0,i}$. Since $c_{0,i}, c'_{0,i} \in \{-1,1\}$, we obtain g_0^2 from $r_{0,i}$ and $r'_{0,i}$. Since we assume that our group G allows an efficient square-root algorithm, it follows that we can obtain g_0 from g_0^2.

4 Our Blind Signature

4.1 Description of our Blind Signature

In this section, we present the description of our blind signature for generic group actions following the framework of Katsumata et al. [46]. We consider a group G acting on a set S and fix an element $E \in S$. We also consider the group G in which computing square-roots is efficient. Let $H : \{0,1\}^* \to \{-1,1\}^n$ be a hash function modelled as a random oracle in the security proof. The blind signature BS consists of the following algorithms, which are summarized in Fig. 4.

$\mathsf{BS.KGen}(1^n)$: On input the security parameter 1^n, it samples a bit $\delta \in \{0,1\}$, $(g_0, g_1) \in G^2$ and computes $A_b^{(c)} = g_b^c * E$ for $b \in \{0,1\}$ and $c \in \{-1,1\}$. It outputs a public key $\mathsf{pk} = (A_0^{(1)}, A_0^{(-1)}, A_1^{(1)}, A_1^{(-1)})$ and a secret key $\mathsf{sk} = (\delta, g_\delta)$.

$\mathsf{BS.S}_1(\mathsf{sk})$: On input the secret key $\mathsf{sk} = (\delta, g_\delta)$, the signer first samples $\mathbf{h}_\delta \leftarrow\!\!\$\ G^n$, and sets $\mathbf{Y}_\delta^{(c)} = \mathbf{h}_\delta^c * E$ for $c \in \{-1,1\}$. Then it samples $(\mathbf{c}^*_{1-\delta}, \mathbf{r}^{*(1)}_{1-\delta}, \mathbf{r}^{*(-1)}_{1-\delta}) \leftarrow\!\!\$\ \{-1,1\}^n \times G^n \times G^n$ and computes $\mathbf{Y}^{(c)}_{1-\delta} = \mathbf{r}^{*(c)}_{1-\delta} * A_{1-\delta}^{(c \odot \mathbf{c}^*_{1-\delta})}$ for $c \in \{-1,1\}$. It outputs the signer state $\mathsf{state}_S = (\mathbf{h}_\delta, \mathbf{c}^*_{1-\delta}, \mathbf{r}^{*(1)}_{1-\delta}, \mathbf{r}^{*(-1)}_{1-\delta})$ and the first-sender message $\rho_{S,1} = (\mathbf{Y}_0^{(1)}, \mathbf{Y}_0^{(-1)}, \mathbf{Y}_1^{(1)}, \mathbf{Y}_1^{(-1)})$.

$\mathsf{BS.U}_1(\mathsf{pk}, M, \rho_{S,1})$: On input the public key $\mathsf{pk} = (A_0^{(1)}, A_0^{(-1)}, A_1^{(1)}, A_1^{(-1)})$, a message M, and the first-sender message $\rho_{S,1} = (\mathbf{Y}_0^{(1)}, \mathbf{Y}_0^{(-1)}, \mathbf{Y}_1^{(1)}, \mathbf{Y}_1^{(-1)})$, it samples, for $b \in \{0,1\}$, $(\mathbf{d}_b, \mathbf{z}_b) \leftarrow\!\!\$\ \{-1,1\}^n \times G^n$ and sets $\mathbf{Z}_b = \mathbf{z}_b * \mathbf{Y}_b^{(\mathbf{d}_b)}$. Then it computes $\mathbf{c} = H(\mathbf{Z}_0 \| \mathbf{Z}_1 \| M)$ and sets $\mathbf{c}^* = \mathbf{c} \odot \mathbf{d}_0 \odot \mathbf{d}_1 \in \{-1,1\}^n$. It outputs the internal user state $\mathsf{state}_U = (\mathbf{d}_b, \mathbf{z}_b)_{b \in \{0,1\}}$ and a user message $\rho_U = \mathbf{c}^*$.

$\mathsf{BS.S}_2(\mathsf{state}_S, \rho_U)$: On input the internal state $\rho_S = (\mathbf{h}_\delta, \mathbf{c}^*_{1-\delta}, \mathbf{r}^{*(1)}_{1-\delta}, \mathbf{r}^{*(-1)}_{1-\delta})$ and a user message $\rho_U = \mathbf{c}^*$, it computes $\mathbf{c}^*_\delta = \mathbf{c}^* \odot \mathbf{c}^*_{1-\delta} \in \{-1,1\}^n$ and $\mathbf{r}^{*(c)}_\delta = \mathbf{h}_\delta^c g_\delta^{-c \odot \mathbf{c}^*_\delta}$, for $c \in \{-1,1\}$. It outputs the second-sender message $\rho_{S,2} = (\mathbf{c}^*_b, \mathbf{r}^{*(1)}_b, \mathbf{r}^{*(-1)}_b)_{b \in \{0,1\}}$.

$\mathsf{BS.U}_2(\mathsf{state}_U, \rho_{S,2})$: On input the internal user state $\mathsf{state}_U = (\mathbf{d}_b, \mathbf{z}_b)_{b \in \{0,1\}}$ and $\rho_{S,2} = (\mathbf{c}^*_b, \mathbf{r}^{*(1)}_b, \mathbf{r}^{*(-1)}_b)_{b \in \{0,1\}}$, it sets $\mathbf{c}_b = \mathbf{c}^*_b \odot \mathbf{d}_b, \mathbf{r}_b^{(\mathbf{d}_b)} = \mathbf{z}_b(\mathbf{r}_b^{*(\mathbf{d}_b)})$ for $b \in \{0,1\}$. Then it checks if

$$\mathbf{c}_0 \odot \mathbf{c}_1 = H(\mathbf{r}_0^{(\mathbf{d}_0)} * \mathbf{A}_0^{(\mathbf{c}_0)} \| \mathbf{r}_1^{(\mathbf{d}_1)} * A_1^{(\mathbf{c}_1)} \| M). \tag{2}$$

If it holds then it outputs a signature $\sigma = (\mathbf{c}_b, \mathbf{r}_b^{(1)}, \mathbf{r}_b^{(-1)})_{b \in \{0,1\}}$.

$\mathsf{BS.Verify}(\mathsf{pk}, M, \sigma)$: On input the public key $\mathsf{pk} = (A_0^{(1)}, A_0^{(-1)}, A_1^{(1)}, A_1^{(-1)})$, a message M and a signature $\sigma = (\mathbf{c}_b, \mathbf{r}_b^{(1)}, \mathbf{r}_b^{(-1)})_{b \in \{0,1\}}$, it outputs 1 if the equation (2) holds, and 0 otherwise.

4.2 Correctness and Blindness

Correctness. We need to verify the equation (2), i.e.,

$$\mathbf{c}_0 \odot \mathbf{c}_1 = H(\mathbf{r}_0^{(1)} * \mathbf{A}_0^{(\mathbf{c}_0)} \| \mathbf{r}_0^{(-1)} * \mathbf{A}_0^{(-\mathbf{c}_0)} \| \mathbf{r}_1^{(1)} * A_1^{(\mathbf{c}_1)} \| \mathbf{r}_1^{(-1)} * A_1^{(-\mathbf{c}_1)} \| M).$$

holds if both the signer and user follow the protocol honestly. First of all, it is easy to check that $\mathbf{Y}_b^{(\mathbf{c})} = (\mathbf{r}_b^{*(\mathbf{c})}) * A_b^{(\mathbf{c} \odot \mathbf{c}^*_b)}$ for $b \in \{0,1\}$ and $\mathbf{c}, \mathbf{c}^*_b \in \{-1,1\}^n$. This is obvious from the protocol (lines 206-207) for the case $b = 1 - \delta$. For $b = \delta$, we have that

$$\mathbf{Y}_\delta^{(\mathbf{c})} = \mathbf{h}_\delta^{\mathbf{c}} * E = \mathbf{h}_\delta^{\mathbf{c}} g_\delta^{-\mathbf{c} \odot \mathbf{c}^*_b} * A_\delta^{(\mathbf{c} \odot \mathbf{c}^*_b)} = (\mathbf{r}_\delta^{*(\mathbf{c})}) * A_\delta^{(\mathbf{c} \odot \mathbf{c}^*_b)}$$

$\mathsf{BS.KGen}(1^n)$

101 : $\delta \leftarrow\!\$\ \{0,1\}$

102 : $(g_0, g_1) \leftarrow\!\$\ G^2$

103 : $(A_0^{(1)}, A_0^{(-1)}) \leftarrow (g_0 * E, g_0^{-1} * E)$

104 : $(A_1^{(1)}, A_1^{(-1)}) \leftarrow (g_1 * E, g_1^{-1} * E)$

105 : **return** $\mathsf{pk} = (A_0^{(1)}, A_0^{(-1)}, A_1^{(1)}, A_1^{(-1)})$

106 : $\mathsf{sk} = (\delta, g_\delta)$

$\mathsf{BS.S}_1(\mathsf{sk})$

201 : **parse** $(\delta, g_\delta) \leftarrow \mathsf{sk}$

202 : $\mathbf{h}_\delta \leftarrow\!\$\ G^n$

203 : $\mathbf{Y}_\delta^{(1)} \leftarrow \mathbf{h}_\delta * E, \mathbf{Y}_\delta^{(-1)} \leftarrow \mathbf{h}_\delta^{-1} * E$

204 : $(\mathbf{c}^*_{1-\delta}, \mathbf{r}^{*(1)}_{1-\delta}, \mathbf{r}^{*(-1)}_{1-\delta}) \leftarrow\!\$\ \{-1,1\}^n \times G^n \times G^n$

205 : $\mathbf{Y}_{1-\delta}^{(1)} \leftarrow \mathbf{r}^{*(1)}_{1-\delta} * A_{1-\delta}^{(\mathbf{c}^*_{1-\delta})}$

206 : $\mathbf{Y}_{1-\delta}^{(-1)} \leftarrow \mathbf{r}^{*(-1)}_{1-\delta} * A_{1-\delta}^{(-\mathbf{c}^*_{1-\delta})}$

207 : $\mathsf{state}_S \leftarrow (\mathbf{h}_\delta, \mathbf{c}^*_{1-\delta}, \mathbf{r}^{*(1)}_{1-\delta}, \mathbf{r}^{*(-1)}_{1-\delta})$

208 : $\rho_{S,1} = (\mathbf{Y}_0^{(1)}, \mathbf{Y}_0^{(-1)}, \mathbf{Y}_1^{(1)}, \mathbf{Y}_1^{(-1)})$

209 : **return** $(\mathsf{state}_S, \rho_{S,1})$

$\mathsf{BS.U}_1(\mathsf{pk}, M, \rho_{S,1})$

301 : **parse** $(\mathbf{Y}_0^{(1)}, \mathbf{Y}_0^{(-1)}, \mathbf{Y}_1^{(1)}, \mathbf{Y}_1^{(-1)}) \leftarrow \rho_{S,1}$

302 : **for** $b \in \{0,1\}$

303 : $\quad (\mathbf{d}_b, \mathbf{z}_b) \leftarrow\!\$\ \{-1,1\}^n \times G^n$

304 : $\quad \mathbf{Z}_b^{(1)} \leftarrow \mathbf{z}_b * \mathbf{Y}_b^{(\mathbf{d}_b)}$

305 : $\quad \mathbf{Z}_b^{(-1)} \leftarrow \mathbf{z}_b^{-1} * \mathbf{Y}_b^{(-\mathbf{d}_b)}$

306 : $\mathbf{c} \leftarrow H(\mathbf{Z}_0^{(1)} \| \mathbf{Z}_0^{(-1)} \| \mathbf{Z}_1^{(1)} \| \mathbf{Z}_1^{(-1)} \| M)$

307 : $\mathbf{c}^* \leftarrow \mathbf{c} \odot \mathbf{d}_0 \odot \mathbf{d}_1 \in \{-1,1\}^n$

308 : $\mathsf{state}_U \leftarrow (\mathbf{d}_b, \mathbf{z}_b)_{b\in\{0,1\}}$

309 : **return** $(\mathsf{state}_U, \rho_U = \mathbf{c}^*)$

$\mathsf{BS.S}_2(\mathsf{state}_S, \rho_U)$

401 : **parse** $(\mathbf{h}_\delta, \mathbf{c}^*_{1-\delta}, \mathbf{r}^{*(1)}_{1-\delta}, \mathbf{r}^{*(-1)}_{1-\delta}) \leftarrow \mathsf{state}_S$

402 : **parse** $\mathbf{c}^* \leftarrow \rho_U$

403 : $\mathbf{c}^*_\delta \leftarrow \mathbf{c}^* \odot \mathbf{c}^*_{1-\delta} \in \{-1,1\}^n$

404 : $\mathbf{r}^{*(1)}_\delta \leftarrow \mathbf{h}_\delta g_\delta^{-\mathbf{c}^*_\delta}$

405 : $\mathbf{r}^{*(-1)}_\delta \leftarrow \mathbf{h}_\delta^{-1} g_\delta^{\mathbf{c}^*_\delta}$

406 : **return** $\rho_{S,2} = (\mathbf{c}^*_b, \mathbf{r}^{*(1)}_b, \mathbf{r}^{*(-1)}_b)_{b\in\{0,1\}}$

$\mathsf{BS.U}_2(\mathsf{state}_U, \rho_{S,2})$

501 : **parse** $(\mathbf{d}_b, \mathbf{z}_b)_{b\in\{0,1\}} \leftarrow \mathsf{state}_U$

502 : **parse** $(\mathbf{c}^*_b, \mathbf{r}^{*(1)}_b, \mathbf{r}^{*(-1)}_b)_{b\in\{0,1\}} \leftarrow \rho_{S,2}$

503 : **for** $b \in \{0,1\}$

504 : $\quad \mathbf{c}_b \leftarrow \mathbf{c}^*_b \odot \mathbf{d}_b$

505 : $\quad \mathbf{r}_b^{(1)} \leftarrow \mathbf{z}_b(\mathbf{r}_b^{*(\mathbf{d}_b)})$

506 : $\quad \mathbf{r}_b^{(-1)} \leftarrow \mathbf{z}_b^{-1}(\mathbf{r}_b^{*(-\mathbf{d}_b)})$

507 : $\mathbf{c}' \leftarrow H(\mathbf{r}_0^{(1)} * A_0^{(\mathbf{c}_0)} \| \mathbf{r}_0^{(-1)} * A_0^{(-\mathbf{c}_0)} \|$

508 : $\qquad \mathbf{r}_1^{(1)} * A_1^{(\mathbf{c}_1)} \| \mathbf{r}_1^{(-1)} * A_1^{(-\mathbf{c}_1)} \| M)$

509 : **if** $\mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}'$

510 : $\quad$ **return** $\sigma = (\mathbf{c}_b, \mathbf{r}_b^{(1)}, \mathbf{r}_b^{(-1)})_{b\in\{0,1\}}$

511 : **return** $\sigma = \perp$

$\mathsf{BS.Verify}(\mathsf{pk}, M, \sigma)$

601 : **parse** $(\mathbf{c}_b, \mathbf{r}_b^{(1)}, \mathbf{r}_b^{(-1)})_{b\in\{0,1\}} \leftarrow \sigma$

602 : $\mathbf{c}' \leftarrow H(\mathbf{r}_0^{(1)} * A_0^{(\mathbf{c}_0)} \| \mathbf{r}_0^{(-1)} * A_0^{(-\mathbf{c}_0)} \|$

603 : $\qquad \mathbf{r}_1^{(1)} * A_1^{(\mathbf{c}_1)} \| \mathbf{r}_1^{(-1)} * A_1^{(-\mathbf{c}_1)} \| M)$

604 : **if** $\mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}'$

605 : $\quad$ **return** 1

606 : **return** 0

Fig. 4: Blind Signature from Cryptographic Group Actions

where the second equation follows from the key generation procedure ($\mathsf{BS.KGen}(1^n)$), the third equation follows from lines 404-405 in Fig. 4.

Now for $b \in \{0,1\}$ and $\mathbf{c}_b \in \{-1,1\}^n$, by substituting $\mathbf{c}_b = \mathbf{c}_b^* \odot \mathbf{d}_b$ and $\mathbf{r}_b^{(1)} = \mathbf{z}_b(\mathbf{r}_b^{*(\mathbf{d}_b)})$, $\mathbf{r}_b^{(-1)} = \mathbf{z}_b^{-1}(\mathbf{r}_b^{*(-\mathbf{d}_b)})$ we have

$$\begin{aligned}
\mathbf{r}_b^{(1)} * A_b^{(\mathbf{c}_b)} &= \mathbf{z}_b(\mathbf{r}_b^{*(\mathbf{d}_b)}) * A_b^{(\mathbf{d}_b \odot \mathbf{c}_b^*)} = \mathbf{z}_b * (\mathbf{r}_b^{*(\mathbf{d}_b)} * A_b^{(\mathbf{d}_b \odot \mathbf{c}_b^*)}) \\
&= \mathbf{z}_b * \mathbf{Y}_b^{(\mathbf{d}_b)} = \mathbf{Z}_b^{(1)}. \\
\mathbf{r}_b^{(-1)} * A_b^{(-\mathbf{c}_b)} &= \mathbf{z}_b^{-1}(\mathbf{r}_b^{*(-\mathbf{d}_b)}) * A_b^{(-\mathbf{d}_b \odot \mathbf{c}_b^*)} = \mathbf{z}_b^{-1} * (\mathbf{r}_b^{*(-\mathbf{d}_b)} * A_b^{(-\mathbf{d}_b \odot \mathbf{c}_b^*)}) \\
&= \mathbf{z}_b^{-1} * \mathbf{Y}_b^{(-\mathbf{d}_b)} = \mathbf{Z}_b^{(-1)}.
\end{aligned}$$

Finally, $\mathbf{c} = \mathbf{c}^* \odot \mathbf{d}_0 \odot \mathbf{d}_1 = \mathbf{c}_0^* \odot \mathbf{c}_1^* \odot \mathbf{d}_0 \odot \mathbf{d}_1 = \mathbf{c}_0 \odot \mathbf{c}_1$, where $\mathbf{c} = H(\mathbf{Z}_0^{(1)} \| \mathbf{Z}_0^{(-1)} \| \mathbf{Z}_1^{(1)} \| \mathbf{Z}_1^{(-1)} \| M)$, we have that $\mathbf{c}_0 \odot \mathbf{c}_1 = H(\mathbf{r}_0^{(1)} * \mathbf{A}_0^{(\mathbf{c}_0)} \| \mathbf{r}_0^{(-1)} * \mathbf{A}_0^{(-\mathbf{c}_0)} \| \mathbf{r}_1^{(1)} * A_1^{(\mathbf{c}_1)} \| \mathbf{r}_1^{(-1)} * A_1^{(-\mathbf{c}_1)} \| M)$ as desired.

Theorem 2 (Blindness). *The blind signature scheme in Fig. 4 is blind, with overwhelming probability, under chosen keys.*

Proof. The proof is similar to that of CSI-Otter [46]: we will show that for any valid public key pk, and first and second signer message $\rho_{S,1}, \rho_{S,2}$, and valid signature σ, there exists a unique and pairwise distinct user state state_U, with overwhelming probability, that could have generated σ. First, the validity of the public key can be efficiently verified using the protocol in Fig. 1.

Fix now sk (and hence pk), $\rho_{S,1} = (\mathbf{Y}_0^{(c)}, \mathbf{Y}_1^{(c)})_{c \in \{-1,1\}}$, $\rho_{S,2} = (\mathbf{c}_b^*, \mathbf{r}_b^{*(1)}, \mathbf{r}_b^{*(-1)})_{b \in \{0,1\}}$, a valid signature $\sigma = (\mathbf{c}_b, \mathbf{r}_b^{(1)}, \mathbf{r}_b^{(-1)})_{b \in \{0,1\}}$. We define the user state $\mathsf{state}_U = (\mathbf{d}_b, \mathbf{z}_b)_{b \in \{0,1\}}$ as $\mathbf{d}_b = \mathbf{c}_b \odot \mathbf{c}_b^*$ and $\mathbf{z}_b = \mathbf{r}_b^{(\mathbf{d}_b)} \mathbf{r}_b^{*(-\mathbf{d}_b)}$ for $b \in \{0,1\}$. Similar to the proof of correctness, we have for $b \in \{0,1\}$ and $c \in \{-1,1\}$:

$$\mathbf{Z}_b^{(c)} := \mathbf{z}_b^c * (\mathbf{Y}_b)^{(c \odot \mathbf{d}_b)} = \mathbf{r}_b^{(c)} \mathbf{r}_b^{*(-c \odot \mathbf{d}_b)} * (\mathbf{r}_b^{*(c \odot \mathbf{d}_b)} * A_b^{(c \odot \mathbf{d}_b \odot \mathbf{c}_b^*)}) = \mathbf{r}_b^{(c)} * A_b^{(c \odot \mathbf{c}_b)}.$$

In addition, since σ is a valid signature, we have

$$\mathbf{c}_0 \odot \mathbf{c}_1 = H((\mathbf{r}_b^{(c)} * A_b^{(c \odot \mathbf{c}_b)})_{b \in \{0,1\}, c \in \{1,-1\}} \| M) = H((\mathbf{Z}_b^{(c)})_{b \in \{0,1\}, c \in \{1,-1\}} \| M).$$

Therefore, the defined state_U is exactly the user state in generating the signature σ. Moreover, for any choice of $\rho_{S,2}$ and any $\sigma \neq \sigma'$, it is clear that the corresponding user states state_U and state'_U are distinct. This completes the proof. □

4.3 Proof for One-More Unforgeability

We will now follow the sufficient conditions described in Section 2.5 for the proof. We first need to define instances, the map $\Phi_{\mathsf{rand}, \vec{h}}$, and the witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$. Then we will show that with our definitions, Lemma 1 and Lemma 2 hold.

In what follows, we denote by $\overrightarrow{X}$ the vector $(X^{(1)}, \ldots, X^{(l)})$ and endow $\overrightarrow{X}$ with the same operations defined for $X^{(k)}$ by operating component wise. Recall that rand is the adversary's randomness, and $\overrightarrow{h} = (\mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(l)})$ is the random's oracle response vector conditioned on the adversary making only l random oracle queries. Furthermore, once $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ is fixed, the query transcript $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ is defined, which is the vector of user message ρ_U queries made to the signing algorithm $\mathsf{BS.S}_2$, denoted by $\mathbf{c}^*$.

Preparation: Instances. We now first define **0**-instance $\mathbf{I}_0$ and **1**-instance $\mathbf{I}_1$. We assume that the adversary makes l signing queries in total. Recall that instance $\mathbf{I}_0$ (resp. $\mathbf{I}_1$) will consist of the secret key $\mathsf{sk}_0 = g_0$ (resp. $\mathsf{sk}_1 = g_1$) and the randomness used by the honest signer algorithm when the secret key is fixed to $\mathsf{sk}_0 = g_0$ (resp. $\mathsf{sk}_1 = g_1$).

A **0**-side instance $\mathbf{I}_0 = (0, g_0, A_1^{(1)}, A_1^{(-1)}, \mathbf{h}_0, \mathbf{c}_1^*, \mathbf{r}_1^{*(1)}, \mathbf{r}_1^{*(-1)})$ is defined as follows.

- $(0, g_0)$: The secret key sk_0 when $\delta = 0$.
- $A_1^{(1)}, A_1^{(-1)}$: The part of the public key $\mathsf{pk} = (A_0^{(c)}, A_1^{(c)})_{c\in\{-1,1\}}$ whose secret key is unknown.
- $(\mathbf{h}_0)^{(k)}$: The randomness of the commitment $(\mathbf{Y}_0)^{(k)}$ in the k-th ($k \in [l]$) first-sender message when $\delta = 0$ such that $(\mathbf{Y}_0^{(c)})^{(k)} = (\mathbf{h}_0^c)^{(k)} * E$ for $c \in \{-1, 1\}$.
- $(\mathbf{c}_1^*)^{(k)}$: The simulated challenge in the k-th ($k \in [l]$) first-sender message when $\delta = 0$.
- $((\mathbf{r}_1^{*(1)})^{(k)}, (\mathbf{r}_1^{*(-1)})^{(k)})$: The randomnesses in generating the commitment $(\mathbf{Y}_1^{(c)})^{(k)}$ in the k-th ($k \in [l]$) first-sender message when $\delta = 0$ such that $(\mathbf{Y}_1^{(c)})^{(k)} = (\mathbf{r}_1^{*(c)})^{(k)} * A_1^{(c\cdot(\mathbf{c}_1^*)^{(k)})}$ for $c \in \{-1, 1\}$.

A **1**-side instance $\mathbf{I}_1 = (1, g_1, A_0^{(1)}, A_0^{(-1)}, \mathbf{h}_1, \mathbf{c}_0^*, \mathbf{r}_0^{*(1)}, \mathbf{r}_0^{*(-1)})$ is defined as follows.

- $(1, g_1)$: The secret key sk_1 when $\delta = 1$.
- $A_0^{(1)}, A_0^{(-1)}$: The part of the public key $\mathsf{pk} = (A_0^{(c)}, A_1^{(c)})_{c\in\{-1,1\}}$ whose secret key is unknown.
- $(\mathbf{h}_1)^{(k)}$: The randomness of the commitment $(\mathbf{Y}_1^{(c)})^{(k)}$ in the k-th ($k \in [l]$) first-sender message when $\delta = 1$ such that $(\mathbf{Y}_1^{(c)})^{(k)} = (\mathbf{h}_1^c)^{(k)} * E$ for $c \in \{-1, 1\}$.
- $(\mathbf{c}_0^*)^{(k)}$: The simulated challenge in the k-th ($k \in [l]$) first-sender message when $\delta = 1$.
- $((\mathbf{r}_0^{*(1)})^{(k)}, (\mathbf{r}_0^{*(-1)})^{(k)})$: The randomness in generating the commitment $(\mathbf{Y}_0^{(c)})^{(k)}$ in the k-th ($k \in [l]$) first-sender message when $\delta = 1$ such that $(\mathbf{Y}_0^{(c)})^{(k)} = (\mathbf{r}_0^{*(c)})^{(k)} * A_0^{(c\cdot(\mathbf{c}_0^*)^{(k)})}$.

Preparation: Map $\Phi_{\mathsf{rand},\overrightarrow{h}}$. We define the map $\Phi_{\mathsf{rand},\overrightarrow{h}}$ that maps a **0**-side instance $\mathbf{I}_0$ into a **1**-side instance $\mathbf{I}_1$ and vice versa as follows.

– A **0**-side instance $\mathbf{I}_0 = (0, g_0, A_1^{(1)}, A_1^{(-1)}, \mathbf{h}_0, \mathbf{c}_1^*, \mathbf{r}_1^{*(1)}, \mathbf{r}_1^{*(-1)})$ into a **1**-side instance $\mathbf{I}_1$ such that

$$\mathbf{I}_1 = (1, g_1, A_0^{(1)} = g_0 * E, A_0^{(-1)} = g_0^{-1} * E, \\ \mathbf{h}_1 = \mathbf{r}_1^{*(1)} g_1^{\mathbf{c}_1^*}, \mathbf{c}_0^* = \mathbf{c}^* \odot \mathbf{c}_1^*, \mathbf{r}_0^{*(c)} = \mathbf{h}_0^c g_0^{-c \odot \mathbf{c}_0^*})$$

where $c \in \{-1, 1\}$, g_1 is such that $g_1 * E = A_1^{(1)}, g_1^{-1} * E = A_1^{(-1)}$, and $\mathbf{c}^* = \overrightarrow{e}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h})$.

– A **1**-side instance $\mathbf{I}_1 = (1, g_1, A_0^{(1)}, A_0^{(-1)}, \mathbf{h}_1, \mathbf{c}_0^*, \mathbf{r}_0^{*(1)}, \mathbf{r}_0^{*(-1)})$ into a **0**-side instance $\mathbf{I}_0$ such that

$$\mathbf{I}_0 = (0, g_0, A_1^{(1)} = g_1 * E, A_1^{(-1)} = g_1^{-1} * E, \\ \mathbf{h}_0 = \mathbf{r}_0^{*(1)} g_0^{\mathbf{c}_0^*}, \mathbf{c}_1^* = \mathbf{c}^* \odot \mathbf{c}_0^*, \mathbf{r}_1^{*(c)} = \mathbf{h}_1^c g_1^{-c \odot \mathbf{c}_1^*})$$

where $c \in \{-1, 1\}$, g_0 is such that $g_0 * E = A_0^{(1)}, g_0^{-1} * E = A_0^{(-1)}$, and $\mathbf{c}^* = \overrightarrow{e}(\mathbf{I}_1, \mathsf{rand}, \overrightarrow{h})$.

Lemma 3. *Lemma 1 holds for our definition of the map* $\Phi_{\mathsf{rand}, \overrightarrow{h}}$.

Proof. Since the proof for **0**-side instance $\mathbf{I}_0$ and that of **1**-side instance $\mathbf{I}_1$ are similar, we present only for the case of **0**-side instance $\mathbf{I}_0$. For any $\mathsf{rand}, \overrightarrow{h}$, consider the query transcript $\overrightarrow{e}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}) = \mathbf{c}^*$. Since the underlying sigma protocol is HVZK, and hence perfectly witness indistinguishable (see Section 3.2, for each $k \in [l]$ and $\mathbf{c}^{(k)}$, there is a set of randomness, defined by $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0)$, that the signer with secret key $(1, g_1)$ (i.e., a **1**-side witness) could have used to produce the same view to the adversary. Therefore we have $\mathsf{trans}(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}) = \mathsf{trans}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0), \mathsf{rand}, \overrightarrow{h})$. In addition, it is easy to check from the above definition of $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ that $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}_0)) = \mathbf{I}_0$. Hence $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ is a self-bijection. This concludes the proof. □

Preparation: Witness Extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$. Fix $\mathbf{I}, \mathsf{rand}$ and let $(\overrightarrow{h}, \overrightarrow{h}') \in F_i(\mathbf{I}, \mathsf{rand})$ for some index $i \in [l+1]$. Denote by $\sigma = (\mathbf{c}_b, \mathbf{r}_b)_{b \in \{0,1\}}$ and $\sigma' = (\mathbf{c}'_b, \mathbf{r}'_b)_{b \in \{0,1\}}$ the signatures corresponding to $\mathbf{c}^{(i)}$ and $\mathbf{c}'^{(i)}$, where $\mathbf{c}^{(i)}$ (resp. $\mathbf{c}'^{(i)}$) is the i-th entry of $\overrightarrow{h}$ (resp. $\overrightarrow{h}'$). It follows from the protocol that $\mathbf{c}^{(i)} = \mathbf{c}_0 \odot \mathbf{c}_1$ and $\mathbf{c}'^{(i)} = \mathbf{c}'_0 \odot \mathbf{c}'_1$. We define the witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ as in Fig. 5.

Lemma 4. *Under the Assumption 1, the witness extractors* $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ *defined in Fig. 5 satisfy the definition in Definition 15.*

Proof. The proof is similar to that of [46, Lemma 4.3]. By the definition of $F_i(\mathbf{I}, \mathsf{rand})$, we have $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ and $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$. Since $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$, two signatures σ, σ' are valid, i.e.,

$$\mathbf{c}^{(i)} = \mathbf{c}_0 \odot \mathbf{c}_1 = H((\mathbf{r}_b^{(c)} * A_b^{(c \odot \mathbf{c}_b)})_{b \in \{0,1\}, c \in \{1,-1\}} \| M)$$

$\mathsf{Ext}_0(\sigma, \sigma')$	$\mathsf{Ext}_1(\sigma, \sigma')$
1 : **if** $\exists t \in [n]$ s.t. $c_{0,t} \neq c'_{0,t}$	1 : **if** $\exists t \in [n]$ s.t. $c_{1,t} \neq c'_{1,t}$
2 : **return** g_0^2 as either $r_{0,t}^{-1} r'_{0,t}$	2 : **return** g_1^2 as either $r_{1,t}^{-1} r'_{1,t}$
3 : or $(r'_{0,t})^{-1} r_{0,t}$	3 : or $(r'_{1,t})^{-1} r_{1,t}$
4 : **return** $\perp$	4 : **return** $\perp$

Fig. 5: Witness Extractors

and

$$\mathbf{c}'^{(i)} = \mathbf{c}'_0 \odot \mathbf{c}'_1 = H((\mathbf{r}'^{(c)}_b * A_b^{(c \odot \mathbf{c}_b)})_{b \in \{0,1\}, c \in \{1,-1\}} \| M').$$

Since $\overrightarrow{h}$ and $\overrightarrow{h}'$ agree up to the i-th index and the challenge and adversary's randomness are fixed, the input to the hash functions are the same, i.e.,

$$\mathbf{r}_b^{(c)} * A_b^{(\mathbf{c}_b)} = \mathbf{r}'^{(c)}_b * A_b^{(\mathbf{c}'_b)} \text{ for } b \in \{0,1\}, c \in \{1,-1\} \wedge M = M'.$$

Since $\mathbf{c}^{(i)} \neq \mathbf{c}'^{(i)}$, we must have either $\mathbf{c}_0 \neq \mathbf{c}'_0$ or $\mathbf{c}_1 \neq \mathbf{c}'_1$. By the soundness of the underlying sigma protocol in Section 3.2, one of the witness extractors Ext_0 or Ext_1 always outputs either g_0^2 or g_1^2 from which a valid secret key (g_0 or g_1) can be easily computed (by Assumption 1). The proof follows. □

Lemma 5. *Lemma 2 holds for our definition of the witness extractors* ($\mathsf{Ext}_0, \mathsf{Ext}_1$).

Proof. We consider the **0**-side case; the **1**-side case is done similarly. We prove by contradiction (following that of [46, Lemma 4.5]). Assume that the **0**-side witness can be extracted from the base $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}')$ at index i, but cannot be extracted from either of the sides $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}')$, $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}'')$ or $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}'')$. By Lemma 4, the assumption holds if and only if $\mathbf{c}_0 = \mathbf{c}''_0$ and $\mathbf{c}'_0 = \mathbf{c}''_0$, which implies that $\mathbf{c}_0 = \mathbf{c}'_0$. This again follows from Lemma 4 that the **0**-side witness cannot be extracted from $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}_0, \mathsf{rand}, \overrightarrow{h}')$, which is a contradiction. This completes the proof. □

Our main theorem is stated as follows.

Theorem 3. *Under Assumption 1 and the hardness of IGAP problem, the blind signature defined in Fig. 4 satisfies the one-more unforgeability property. Concretely, for $l \in \mathbb{N}$, if there exist an adversary $\mathcal{A}$ that makes Q hash queries to the random oracle and breaks the l-one-more unforgeability of BS with advantage $\epsilon_{\mathcal{A}} \geq \frac{C_1}{|\mathcal{C}|}\binom{Q}{l+1}$, then there exists an efficient algorithm $\mathcal{B}$ that breaks the IGAP problem with advantage $\epsilon_{\mathcal{B}} \geq C_2 \cdot \frac{\epsilon_{\mathcal{A}}^2}{\binom{Q}{l+1}^2 \cdot (l+1)^3}$ for some universal positive constants C_1 and C_2.*

Proof. We define the hard instance generator IG to output a IGAP problem instance. Now the proof follows from Lemma 3, Lemma 5 and Theorem 1. □

5 Instantiation based on Monomial Code Equivalence

In this section, we present a concrete blind signature protocol based on monomial code equivalence following the framework in Fig. 4.

5.1 Monomial Code Equivalence

A linear code over $\mathbb{F}_q$ is a subspace of $\mathbb{F}_q^n$. An m-dimensional code in $\mathbb{F}_q^n$ is represented by $C \in \mathrm{Mat}(m \times n, q)$, whose rows form a basis of the code. The monomial code equivalence problem is the following.

Problem 1 (Monomial code equivalence)**.** For $n \in \mathbb{N}$, let $m \in [n]$. Let $C, C' \in \mathrm{Mat}(m \times n, q)$ be of rank m. Decide if there exist $A \in \mathrm{GL}(m, q)$, $D \in \mathrm{D}(n, q)$, and $P \in \mathrm{S}_n$, such that $ACDP = C'$. If yes, compute such A, D, and P.

Equivalently, we can formulate monomial code equivalence as asking if there exist $A \in \mathrm{GL}(m, q)$ and $M \in \mathrm{Mon}(n, q)$, such that $ACM = C'$. By writing $M \in \mathrm{Mon}(n, q)$ as DP where $D \in \mathrm{D}(n, q)$ and $P \in \mathrm{S}_n$, we can define a symmetric group action as below.

Monomial code equivalence as a symmetric group action. Let $C, C' \in \mathrm{Mat}(m \times n, q)$. In Definition 1, three matrices, $A \in \mathrm{GL}(m, q)$, $D \in \mathrm{D}(n, q)$, and $P \in \mathrm{S}_n$, are used to define equivalence between C and C'. As a result, there is more than one way to interpret the group action behind monomial code equivalence.

The first, most straightforward, way is to consider the action of the group $\mathrm{GL}(m, q) \times (\mathrm{D}(n, q) \rtimes \mathrm{S}_n) = \mathrm{GL}(m, q) \times \mathrm{Mon}(n, q)$[7] on the set $\mathrm{Mat}(m \times n, q)$.

The second approach is to consider the monomial group $\mathrm{Mon}(n, q)$ acting on the set of m-dimensional codes in $\mathbb{F}_q^n$. This is the natural action from the viewpoint of coding theory, as seen in [16, 22, 32].

We take the third approach by formulating it as the symmetric group S_n acting on a set S. This set S is the set of equivalence classes of m-dimensional codes in $\mathbb{F}_q^n$ under scalar multiplications on the coordinates. Note that this is actually in line with the second approach, where m-dimensional codes in $\mathbb{F}_q^n$ are actually the set of equivalence classes of the set of invertible matrices in $\mathrm{Mat}(m \times n, q)$ under left multiplying $A \in \mathrm{GL}(m, q)$.

Let us examine this set S in more detail. Recall that $m \in [n]$. For $C_1, C_2 \in \mathrm{Mat}(m \times n, q)$, we define an equivalence relation $\sim$ as $C_1 \sim C_2$ if and only if there exists some $A \in \mathrm{GL}(m, q)$ and $D \in \mathrm{D}(n, q)$ such that $C_1 = AC_2D$. Note that this equivalence relation partitions $\mathrm{Mat}(m \times n, q)$ into orbits of $\mathrm{Mat}(m \times n, q)$ under the action of $\mathrm{GL}(m, q) \times \mathrm{D}(n, q)$. Denote by $[C]_\sim := \{ACD : A \in \mathrm{GL}(m, q), D \in \mathrm{D}(n, q)\}$ the equivalence class determined by $\sim$ and corresponding to $C \in \mathrm{Mat}(m \times n, q)$. Let $\mathrm{Mat}(m \times n, q)/\sim = \{[C]_\sim : C \in \mathrm{Mat}(m \times n, q)\}$ be the set of equivalence classes under $\sim$. This is the set S to be acted on.

We wish to define an action of S_n on $\mathrm{Mat}(m \times n, q)/\sim$. For $P \in \mathrm{S}_n$, since $[C]_\sim := \{ACD : A \in \mathrm{GL}(m, q), D \in \mathrm{D}(n, q)\}$ is a set of matrices, a natural

[7] Here $\rtimes$ denotes semidirect product of groups.

map is to send $[C]_\sim$ to $[C]_\sim P := \{ACDP : A \in \mathrm{GL}(m,q), D \in \mathrm{D}(n,q)\}$. For S_n to act on $\mathrm{Mat}(m \times n, q)/\sim$, we need to show that $[C]_\sim P$ is an element in $\mathrm{Mat}(m \times n, q)/\sim$ by the following proposition.

Proposition 1. *Let $[C]_\sim \in \mathrm{Mat}(m \times n, q)/\sim$, $P \in \mathrm{S}_n$, and $[C]_\sim P$ be as above. Then $[C]_\sim P = [CP]_\sim$.*

Proof. Recall that $C \in \mathrm{Mat}(m \times n, q)$. Let $A \in \mathrm{GL}(m,q)$, and $D \in \mathrm{D}(n,q)$. Note that for any $P \in \mathrm{S}_n$, we have $\mathrm{D}(n,q) = P^{-1}\,\mathrm{D}(n,q)P = \{P^{-1}DP : D \in \mathrm{D}(n,q)\}$. This is because one can verify that $P^{-1}DP$ is a diagonal matrix with ith diagonal entry being the $P(i)$th entry of D. Therefore, $[C]_\sim P := \{ACDP : A \in \mathrm{GL}(m,q), D \in \mathrm{D}(n,q)\} = \{ACPP^{-1}DP : A \in \mathrm{GL}(m,q), D \in \mathrm{D}(n,q)\} = \{ACPD' : A \in \mathrm{GL}(m,q), D' \in \mathrm{D}(n,q)\} = [CP]_\sim$. □

This ensures that the map $\alpha : \mathrm{S}_n \times \mathrm{Mat}(m \times n, q)/\sim \to \mathrm{Mat}(m \times n, q)/\sim$ by $P \in \mathrm{S}_n$ sending $[C]_\sim$ to $[C]_\sim P = [CP]_\sim$ is a well-defined group action. Now, we can translate Fig. 4 into the setting of group action α. By Proposition 1, starting from the public key that consists of four equivalence classes, $A_0^{(1)} = [Eg_0]_\sim$, $A_0^{(-1)} = [Eg_0^{-1}]_\sim$, $A_1^{(1)} = [Eg_1]_\sim$, and $A_1^{(-1)} = [Eg_1^{-1}]_\sim$, it is straightforward to verify $\mathbf{Z}_0 \sim \mathbf{r}_0 * A_0^{(\mathbf{c}_0)}$ and $\mathbf{Z}_1 \sim \mathbf{r}_1 * A_1^{(\mathbf{c}_1)}$[8] for all $\mathbf{h}_\delta \leftarrow\!\$\ G^n$. Now we can state the computational version of our security problem as follows:

Problem 2 (Diagonal-masked Inverse linear code equivalence (DmILCE))**.** For $n \in \mathbb{N}$, let $m \in [n]$. Let $\{C_0, C_1, C_2\} \subseteq \mathrm{Mat}(m \times n, q)$. Decide if there exist $A_0, A_1 \in \mathrm{GL}(m,q)$, $D_0, D_1 \in \mathrm{D}(n,q)$, and $P \in \mathrm{S}_n$, such that $C_1 = A_0 C_0 D_0 P$ and $C_2 = A_1 C_1 D_1 P^{-1}$. If yes, compute such a permutation P.

Comparison with LESS. The formulation of our instantiation mainly consists of two parts: one is about establishing the setting of group G, where we utilize the symmetric group because the computation of the square-roots (in certain family) is efficient; the other is about hiding the elements of the acted set S within an equivalence class, because the single symmetric group actions are not secure enough. In a comparable setting, some recent papers [22, 32] have addressed the following relaxed monomial code equivalence problem with multiple samples.

Problem 3 (Inverse linear code equivalence (ILCE))**.** For $n \in \mathbb{N}$, let $m \in [n]$. Let $\{C_0, C_1, C_2\} \subseteq \mathrm{Mat}(m \times n, q)$. Decide if there exist $A \in \mathrm{GL}(m,q)$ and $M \in \mathrm{Mon}(n,q)$, such that $C_1 = AC_0M$ and $C_2 = A^{-1}C_0M^{-1}$. If yes, compute such a monomial matrix M.

From the group action viewpoint, this problem corresponds to reusing monomial group actions (compared to our symmetric group actions) and there is only a single general linear group $\mathrm{GL}(m,q)$ acting on the left of the generator matrices (compared to our equivalence classes under a composite group $\mathrm{GL}(m,q) \times \mathrm{D}(n,q)$) used to hide secret information during communication. This

[8] For $(s_1, \ldots, s_n), (s'_1, \ldots, s'_n) \in S^n$, we say $(s_1, \ldots, s_n) \sim (s'_1, \ldots, s'_n)$, if $s_i \sim s'_i$ for each $i \in [n]$.

leaves a potential breakthrough point that can be leveraged to tackle Problem 3: Upon converting the known generator matrices of linear codes into their systematic forms, the effect of the left group actions can be somehow eliminated by introducing the parity-check matrices that serve for the construction of a homogeneous linear system involving the monomial matrix as the only variable matrix. Therefore, since the number of equations increases with the number of samples very quickly while the number of variables always remains the same, [22] is able to give a heuristic algorithm to find such a monomial matrix by first converting ILCE problem to the general monomial code equivalence problem with two samples.

However, this cannot apply to our case directly. The reason is that in an equivalence class, the way to scale the columns of the generator matrices can vary, so treating these scalars as variables and combining them with the common permutation as another variable matrix would make a quadratic equation system instead of a linear system. Besides, the algorithm in [22] for solving 2-sample monomial code equivalence relies on the structure of a homogeneous linear system. Specifically, it checks whether an entry of the variable monomial matrix can be non-zero or not, one by one; in each case, the algorithm sets the entry to 1 and then conducts the check, despite the fact that a monomial matrix could have any value on the non-zero position in general. This is because the solutions to the homogeneous linear system should be multiples of a monomial matrix, allowing us to normalize any entry to 1 while staying within the solution set, and avoiding the need to account for the values of scalar variables. Again, since the scalars for the commitment and keys keep updating in our protocol, their values must be considered and cannot simply be set to 1. These two essential gaps differentiate our use of monomial code equivalence from the one broken in LESS.

5.2 Square-root Computation Algorithm

To achieve the soundness of the underlying sigma protocol for our blind signature, we need an algorithm to efficiently compute a certain square-root of $g_\delta^2 \in \mathrm{S}_n$.

Proposition 2. *There is a polynomial-time square-root algorithm that inputs $\sigma \in \mathrm{S}_n$ and outputs a square-root of σ if it exists.*

Proof. We first give a constructive proof to show that any $\sigma \in \mathrm{S}_n$ can be efficiently decomposed into a finite product of disjoint cycles. Let $p_1 \in [n]$ be the smallest element such that $\sigma(p_1) = p_2 \neq p_1$. Now we can denote $\sigma(p_2) = p_3 \neq p_2$, as p_2 has been occupied. It follows that we must end up with some p_k such that $\sigma(p_k) = p_1$, which yields a k-cycle $(p_1, \ldots, p_k)$. Then we let $q_1 \in [n]$ be the smallest element such that $\sigma(q_1) = q_2 \notin \{p_1, \ldots, p_k, q_1\}$ and repeat the procedure to get another cycle. Since n is finite, this will result in a finite product of disjoint cycles eventually.

Suppose there exists one square-root of σ, say $g^2 = \sigma$. Applying the disjoint-cycle decomposition to g, we can denote $g = g_1 \circ \cdots \circ g_\ell$ where g_i is a disjoint

cycle for all $i \in [\ell]$. Note that disjoint cycles always commute, which implies that σ can be represented by $g_1^2 \circ \cdots \circ g_\ell^2$. Thus, it suffices to find a square-root of an arbitrary cycle square g_i^2.

Let $g_i = (p_1, \ldots, p_k)$ be a k-cycle. We see that if k is odd, then $g_i^2 = (p_1, p_3, \ldots, p_{k-2}, p_k)$ is still a k-cycle. If k is even, then $g_i^2 = (p_1, p_3, \ldots, p_{k-1}) \circ (p_2, p_4, \ldots, p_k)$, which consists of two disjoint cycles. These are the only two types for each g_i^2, which can be distinguished by employing the efficient disjoint-cycle decomposition. Finally, we give a specific construction of the square-root in either case:

- If $g_i^2 = (j_1, \ldots, j_k)$, then
$$\sqrt{g_i^2} = \left(j_1, j_{\frac{k+3}{2}}, j_2, j_{\frac{k+5}{2}}, \ldots, j_{\frac{k-1}{2}}, j_{\frac{k+k}{2}}, j_{\frac{k+1}{2}}\right).$$
- If $g_i^2 = (j_1, \ldots, j_k) \circ (j'_1, \ldots, j'_k)$, then $\sqrt{g_i^2}$ has several possibilities, namely $(j_1, j'_1, \ldots, j_k, j'_k)$, $(j_1, j'_2, j_2, j'_3, \ldots, j_k, j'_1)$, $\ldots$, $(j_1, j'_k, j_2, j'_1, \ldots, j_k, j'_{k-1})$.

It can be easily verified that $g_i^2 = (\sqrt{g_i^2})^2$ for all $i \in [\ell]$, and thereby $\sigma = g^2 = (\sqrt{g_1^2} \circ \cdots \circ \sqrt{g_\ell^2})^2$. □

Remark 2. By the proof of Proposition 2, we note that if g is a full cycle in S_n with n odd, then g has a unique square-root. To avoid the complication of multiple square-roots, in the instantiation of the blind signature, the signer needs to set n to be odd, and sample a full-cycle permutation from S_n. The probability of sampling a full-cycle permutation is $1/n$, simply because there are $(n-1)!$ full-cycle permutations in S_n. The signer then needs to sample $\mathbf{r}_{1-\delta}^{*(1)}$ from the set of full-cycle permutations, in order not to lead δ.

We further note that the monomial code equivalence problem would not be easier if the underlying permutation is a full cycle. Indeed, we can reduce from general permutations to full cycles by a random reduction: if C_1 and C_2 are related by a permutation Q, we can apply a random permutation R to C_2 to get C_3, so with probability $1/n$, C_1 and C_3 are related by a full cycle.

5.3 Canonical Form Algorithm

When blinding the message concatenated with the set elements (e.g., $\mathbf{Z}_0, \mathbf{Z}_1 \in S^n$) under a hash function H as in Fig. 4, it is less practical to feed the entire equivalence class into H. By contrast, we can efficiently compute the *canonical forms* that can uniquely represent the underlying equivalence classes. Specifically, for an authentic signature and $b \in \{0, 1\}$, the signer and the user each hold $\mathbf{Z}_b$ and $\mathbf{r}_b * A_b^{(\mathbf{c}_b)}$ in the same equivalence class. To ensure the resulting hash values c and c' to be the same, the user (and other verifiers) need to perform a canonical form algorithm for these equivalence classes. Canonical forms are a well-studied topic for graphs and matrix tuples [6, 58]. Given a group G acting on a set S, a canonical form algorithm for this group action takes $s \in S$ and outputs $s^* \in \mathsf{Orb}(s) := \{g * s : \forall g \in G\}$. Furthermore, for another $s' \in \mathsf{Orb}(s)$, the

algorithm should output the same s^*. In our case, G and S can be interpreted as $\mathrm{GL}(m,q) \times \mathrm{D}(n,q)$ and $\mathrm{Mat}(m \times n, q)$, respectively. To this end, we establish the following proposition.

Proposition 3. *There is a polynomial-time canonical form algorithm for the action of* $\mathrm{GL}(m,q) \times \mathrm{D}(n,q)$ *on* $\mathrm{Mat}(m \times n, q)$.

Proof. Let $A, B \in \mathrm{Mat}(m \times n, q)$. Denote by $r(A), r(B)$ the reduced row echelon form of A, B, respectively. We claim that there exists $T \in \mathrm{GL}(m,q)$ and $D \in \mathrm{D}(n,q)$ such that $A = TBD$ if and only if there exists $D_1 \in \mathrm{D}(m,q)$ and $D_2 \in \mathrm{D}(n,q)$ such that $r(A) = D_1 r(B) D_2$.

The if direction is straightforward. By the definition of the reduced row echelon form, we can find $S_1, S_2 \in \mathrm{GL}(m,q)$ such that $r(A) = S_1 A$ and $r(B) = S_2 B$. Then $r(A) = D_1 r(B) D_2$ for some $D_1 \in \mathrm{D}(m,q)$ and $D_2 \in \mathrm{D}(n,q)$ implies that $A = S_1^{-1} D_1 S_2 B D_2$, where $S_1^{-1} D_1 S_2 \in \mathrm{GL}(m,q)$ and $D_2 \in \mathrm{D}(n,q)$.

For the only if direction, given $A = TBD$ for some $T \in \mathrm{GL}(m,q)$ and $D \in \mathrm{D}(n,q)$, we first consider the form of $r(B)D = S_2 BD$ for $S_2 \in \mathrm{GL}(m,q)$. Note that $D \in \mathrm{D}(n,q)$ is just doing the column scaling and not changing the pattern of non-zero entries in $r(B)$, it follows that $r(B)D$ is already in a row echelon form of $S_2 BD$. To further make it to be the reduced row echelon form $r(S_2 BD)$, the only thing is to scale the leading non-zero entry of each row to 1, which can be done by left-multiplying a diagonal matrix, say $D' \in \mathrm{D}(m,q)$. Thus, we have that $D' r(B) D = r(S_2 BD) = r(BD) = r(TBD) = r(A)$ for some $D' \in \mathrm{D}(m,q)$ and $D \in \mathrm{D}(n,q)$.

Since the reduced row echelon form can be uniquely and efficiently computed by Gaussian elimination in time $O(m^2 n)$, this shows a polynomial-time reduction from finding a canonical form for the action of $\mathrm{GL}(m,q) \times \mathrm{D}(n,q)$ on $\mathrm{Mat}(m \times n, q)$ to finding a canonical form for the action of $\mathrm{D}(m,q) \times \mathrm{D}(n,q)$ on $\mathrm{Mat}(m \times n, q)$. Such questions have been studied in e.g. [34]. Formally, we prove the following.

Proposition 4. *There is a polynomial-time canonical form algorithm for the left-right action of* $\mathrm{D}(m,q) \times \mathrm{D}(n,q)$ *on* $\mathrm{Mat}(m \times n, q)$.

The detailed algorithmic proof for Proposition 4 is provided in Appendix A. Since the reduced row echelon forms are canonical, we can apply the canonical form algorithm for the left-right action of $\mathrm{D}(m,q) \times \mathrm{D}(n,q)$ on the reduced row echelon form of any $A \in \mathrm{Mat}(m \times n, q)$ to obtain a canonical form for the left-right action of $\mathrm{GL}(m,q) \times \mathrm{D}(n,q)$ in a total running time of $O(m^2 n)$. This concludes the proof. □

6 Conclusion

In this paper, we present a framework, following [46], for designing a blind signature from abstract group actions. We prove that our scheme is secure in the random oracle model. We also provide an instantiation from a variant of linear

code equivalence, with an intensive treatment, that can avoid the attack by [22]. As discussed in Section 1.4, our scheme, as similar to CSI-Otter [46], is vulnerable to Do et al.'s attack [33] and possibly also to Katsumata et al.'s attack [47] against CSI-Otter. A consequence of this is, recommended by Do et al. [33], to use our blind signatures sequentially, not concurrently. An interesting question is to investigate whether Katsumata et al.'s attack [47] can be applicable to our framework for abstract group actions, and particularly to our instantiation. Another interesting question, raised by Katsumata et al. in [47] is enable a group action-based blind signature construction from those [1, 44, 63] that can thwart the ROS-related attacks.

We suspect that our framework can be extended to a partially blind signature, as in [46]. In fact, in the Appendix B, we show that we can design a 2-out-of-3 sigma protocol from abstract group actions, the base protocol for constructing the partially isogeny-based blind signature in [46]. We will leave it as a future work to show the possibility/impossibility of such a construction.

Bibliography

[1] Abe, M.: A secure three-move blind signature scheme for polynomially many signatures. In: Pfitzmann, B. (ed.) Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding. Lecture Notes in Computer Science, vol. 2045, pp. 136–151. Springer (2001). https://doi.org/10.1007/3-540-44987-6_9, https://doi.org/10.1007/3-540-44987-6_9

[2] Abe, M., Okamoto, T.: Provably secure partially blind signatures. In: Bellare, M. (ed.) Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1880, pp. 271–286. Springer (2000). https://doi.org/10.1007/3-540-44598-6_17, https://doi.org/10.1007/3-540-44598-6_17

[3] Agrawal, S., Kirshanova, E., Stehlé, D., Yadav, A.: Practical, round-optimal lattice-based blind signatures. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022. pp. 39–53. ACM (2022). https://doi.org/10.1145/3548606.3560650, https://doi.org/10.1145/3548606.3560650

[4] Alamati, N., Feo, L.D., Montgomery, H., Patranabis, S.: Cryptographic group actions and applications. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12492, pp. 411–439. Springer (2020). https://doi.org/10.1007/978-3-030-64834-3_14

[5] Alkadri, N.A., Bansarkhani, R.E., Buchmann, J.: BLAZE: practical lattice-based blind signatures for privacy-preserving applications. In: Bonneau, J., Heninger, N. (eds.) Financial Cryptography and Data Security - 24th International Conference, FC 2020, Kota Kinabalu, Malaysia, February 10-14, 2020 Revised Selected Papers. Lecture Notes in Computer Science, vol. 12059, pp. 484–502. Springer (2020). https://doi.org/10.1007/978-3-030-51280-4_26, https://doi.org/10.1007/978-3-030-51280-4_26

[6] Babai, L.: Canonical form for graphs in quasipolynomial time: preliminary report. In: Charikar, M., Cohen, E. (eds.) Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019. pp. 1237–1246. ACM (2019). https://doi.org/10.1145/3313276.3316356

[7] Barenghi, A., Biasse, J., Ngo, T., Persichetti, E., Santini, P.: Advanced signature functionalities from the code equivalence problem. Int. J. Comput. Math. Comput. Syst. Theory **7**(2), 112–128 (2022).

https://doi.org/10.1080/23799927.2022.2048206, https://doi.org/10.1080/23799927.2022.2048206

[8] Battagliola, M., Borin, G., Meneghetti, A., Persichetti, E.: Cutting the GRASS: threshold group action signature schemes. In: Oswald, E. (ed.) Topics in Cryptology - CT-RSA 2024 - Cryptographers' Track at the RSA Conference 2024, San Francisco, CA, USA, May 6-9, 2024, Proceedings. Lecture Notes in Computer Science, vol. 14643, pp. 460–489. Springer (2024). https://doi.org/10.1007/978-3-031-58868-6_18, https://doi.org/10.1007/978-3-031-58868-6_18

[9] Benhamouda, F., Lepoint, T., Loss, J., Orrù, M., Raykova, M.: On the (in)security of ROS. In: Canteaut, A., Standaert, F. (eds.) Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12696, pp. 33–53. Springer (2021). https://doi.org/10.1007/978-3-030-77870-5_2, https://doi.org/10.1007/978-3-030-77870-5_2

[10] Berthomieu, J., Faugere, J.C., Perret, L.: Polynomial-time algorithms for quadratic isomorphism of polynomials: The regular case. Journal of Complexity **31**(4), 590–616 (2015)

[11] Beullens, W.: Multivariate blind signatures revisited. IACR Cryptol. ePrint Arch. p. 720 (2024), https://eprint.iacr.org/2024/720

[12] Beullens, W., Dobson, S., Katsumata, S., Lai, Y., Pintore, F.: Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13276, pp. 95–126. Springer (2022). https://doi.org/10.1007/978-3-031-07085-3_4

[13] Beullens, W., Katsumata, S., Pintore, F.: Calamari and falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12492, pp. 464–492. Springer (2020). https://doi.org/10.1007/978-3-030-64834-3_16

[14] Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: Efficient isogeny based signatures through class group computations. In: Advances in Cryptology - ASIACRYPT 2019. Lecture Notes in Computer Science, vol. 11921, pp. 227–247. Springer (2019). https://doi.org/10.1007/978-3-030-34578-5_9

[15] Beullens, W., Lyubashevsky, V., Nguyen, N.K., Seiler, G.: Lattice-based blind signatures: Short, efficient, and round-optimal. In: Meng, W., Jensen, C.D., Cremers, C., Kirda, E. (eds.) Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023. pp. 16–29.

ACM (2023). https://doi.org/10.1145/3576915.3616613, https://doi.org/10.1145/3576915.3616613

[16] Biasse, J., Micheli, G., Persichetti, E., Santini, P.: LESS is more: Code-based signatures without syndromes. In: Nitaj, A., Youssef, A.M. (eds.) Progress in Cryptology - AFRICACRYPT 2020 - 12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20-22, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12174, pp. 45–65. Springer (2020). https://doi.org/10.1007/978-3-030-51938-4_3

[17] Bläser, M., Chen, Z., Duong, D.H., Joux, A., Nguyen, T.N., Plantard, T., Qiao, Y., Susilo, W., Tang, G.: On digital signatures based on group actions: QROM security and ring signatures. In: Saarinen, M.O., Smith-Tone, D. (eds.) Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Oxford, UK, June 12-14, 2024, Proceedings, Part I. Lecture Notes in Computer Science, vol. 14771, pp. 227–261. Springer (2024). https://doi.org/10.1007/978-3-031-62743-9_8

[18] Blazy, O., Gaborit, P., Mac, D.T.: A correction to a code-based blind signature scheme. In: Wachter-Zeh, A., Bartz, H., Liva, G. (eds.) Code-Based Cryptography - 9th International Workshop, CBCrypto 2021, Munich, Germany, June 21-22, 2021 Revised Selected Papers. Lecture Notes in Computer Science, vol. 13150, pp. 84–94. Springer (2021). https://doi.org/10.1007/978-3-030-98365-9_5, https://doi.org/10.1007/978-3-030-98365-9_5

[19] Blazy, O., Gaborit, P., Schrek, J., Sendrier, N.: A code-based blind signature. In: 2017 IEEE International Symposium on Information Theory, ISIT 2017, Aachen, Germany, June 25-30, 2017. pp. 2718–2722. IEEE (2017). https://doi.org/10.1109/ISIT.2017.8007023, https://doi.org/10.1109/ISIT.2017.8007023

[20] Bouaziz-Ermann, S., Canard, S., Eberhart, G., Kaim, G., Roux-Langlois, A., Traoré, J.: Lattice-based (partially) blind signature without restart. IACR Cryptol. ePrint Arch. p. 260 (2020), https://eprint.iacr.org/2020/260

[21] Brassard, G., Yung, M.: One-way group actions. In: Advances in Cryptology - CRYPTO 1990. pp. 94–107 (1990). https://doi.org/10.1007/3-540-38424-3_7

[22] Budroni, A., Chi-Domínguez, J., D'Alconzo, G., Scala, A.J.D., Kulkarni, M.: Don't use it twice! solving relaxed linear equivalence problems **15491**, 35–65 (2024). https://doi.org/10.1007/978-981-96-0944-4_2, https://doi.org/10.1007/978-981-96-0944-4_2

[23] Buser, M., Dowsley, R., Esgin, M.F., Gritti, C., Kermanshahi, S.K., Kuchta, V., LeGrow, J.T., Liu, J.K., Phan, R.C., Sakzad, A., Steinfeld, R., Yu, J.: A survey on exotic signatures for post-quantum blockchain: Challenges and research directions. ACM Comput. Surv. **55**(12), 251:1–251:32 (2023). https://doi.org/10.1145/3572771, https://doi.org/10.1145/3572771

[24] Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). pp. 395–427. Springer (2018). https://doi.org/10.1007/978-3-030-03332-3_15

[25] Chairattana-Apirom, R., Hanzlik, L., Loss, J., Lysyanskaya, A., Wagner, B.: Pi-cut-choo and friends: Compact blind signatures via parallel instance cut-and-choose and more. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13509, pp. 3–31. Springer (2022). https://doi.org/10.1007/978-3-031-15982-4_1, https://doi.org/10.1007/978-3-031-15982-4_1

[26] Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D., Rivest, R.L., Sherman, A.T. (eds.) Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, California, USA, August 23-25, 1982. pp. 199–203. Plenum Press, New York (1982). https://doi.org/10.1007/978-1-4757-0602-4_18, https://doi.org/10.1007/978-1-4757-0602-4_18

[27] Chaum, D.: Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In: Günther, C.G. (ed.) Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings. Lecture Notes in Computer Science, vol. 330, pp. 177–182. Springer (1988). https://doi.org/10.1007/3-540-45961-8_15, https://doi.org/10.1007/3-540-45961-8_15

[28] Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. In: Goldwasser, S. (ed.) Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings. Lecture Notes in Computer Science, vol. 403, pp. 319–327. Springer (1988). https://doi.org/10.1007/0-387-34799-2_25, https://doi.org/10.1007/0-387-34799-2_25

[29] Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Brickell, E.F. (ed.) Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings. Lecture Notes in Computer Science, vol. 740, pp. 89–105. Springer (1992). https://doi.org/10.1007/3-540-48071-4_7, https://doi.org/10.1007/3-540-48071-4_7

[30] Chou, T., Niederhagen, R., Persichetti, E., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: Take your meds: Digital signatures from matrix code equivalence. In: Progress in Cryptology - AFRICACRYPT 2023 (2023). https://doi.org/10.1007/978-3-031-37679-5_2

[31] Couveignes, J.M.: Hard homogeneous spaces. IACR Cryptology ePrint Archive (2006), http://eprint.iacr.org/2006/291

[32] D'Alconzo, G., Di Scala, A.J.: Representations of group actions and their applications in cryptography. Finite Fields and Their Applications **99**, 102476 (2024). https://doi.org/10.1016/j.ffa.2024.102476

[33] Do, K., Hanzlik, L., Paracucchi, E.: M&m's: Mix and match attacks on schnorr-type blind signatures with repetition. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings,

Part VI. Lecture Notes in Computer Science, vol. 14656, pp. 363–387. Springer (2024). https://doi.org/10.1007/978-3-031-58751-1_13, https://doi.org/10.1007/978-3-031-58751-1_13

[34] Engel, G.M., Schneider, H.: Algorithms for testing the diagonal similarity of matrices and related problems. SIAM Journal on Algebraic Discrete Methods **3**(4), 429–438 (1982). https://doi.org/10.1137/0603044

[35] Feo, L.D., Galbraith, S.D.: Seasign: Compact isogeny signatures from class group actions. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. Lecture Notes in Computer Science, vol. 11478, pp. 759–789. Springer (2019). https://doi.org/10.1007/978-3-030-17659-4_26

[36] Feo, L.D., Meyer, M.: Threshold schemes from isogeny assumptions. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) Public-Key Cryptography - PKC 2020 - 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4-7, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12111, pp. 187–212. Springer (2020). https://doi.org/10.1007/978-3-030-45388-6_7

[37] Fuchsbauer, G., Wolf, M.: Concurrently secure blind schnorr signatures. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part II. Lecture Notes in Computer Science, vol. 14652, pp. 124–160. Springer (2024). https://doi.org/10.1007/978-3-031-58723-8_5, https://doi.org/10.1007/978-3-031-58723-8_5

[38] Hallgren, S., Moore, C., Rötteler, M., Russell, A., Sen, P.: Limitations of quantum coset states for graph isomorphism. J. ACM **57**(6), 34:1–34:33 (Nov 2010). https://doi.org/10.1145/1857914.1857918

[39] Hauck, E., Kiltz, E., Loss, J.: A modular treatment of blind signatures from identification schemes. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part III. Lecture Notes in Computer Science, vol. 11478, pp. 345–375. Springer (2019). https://doi.org/10.1007/978-3-030-17659-4_12, https://doi.org/10.1007/978-3-030-17659-4_12

[40] Hauck, E., Kiltz, E., Loss, J., Nguyen, N.K.: Lattice-based blind signatures, revisited. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12171, pp. 500–529. Springer (2020). https://doi.org/10.1007/978-3-030-56880-1_18, https://doi.org/10.1007/978-3-030-56880-1_18

[41] Heilman, E., Baldimtsi, F., Goldberg, S.: Blindly signed contracts: Anonymous on-blockchain and off-blockchain bitcoin transactions. In: Clark, J., Meiklejohn, S., Ryan, P.Y.A., Wallach, D.S., Brenner, M., Rohloff, K. (eds.) Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers. Lecture Notes in Computer

Science, vol. 9604, pp. 43–60. Springer (2016). https://doi.org/10.1007/978-3-662-53357-4_4, https://doi.org/10.1007/978-3-662-53357-4_4

[42] Hhan, M., Morimae, T., Yamakawa, T.: From the hardness of detecting superpositions to cryptography: Quantum public key encryption and commitments. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 639–667. Springer (2023). https://doi.org/10.1007/978-3-031-30545-0_22

[43] Ji, Z., Qiao, Y., Song, F., Yun, A.: General linear group action on tensors: A candidate for post-quantum cryptography. Cryptology ePrint Archive, Report 2019/687 (2019), https://eprint.iacr.org/2019/687

[44] Kastner, J., Loss, J., Xu, J.: On Pairing-Free Blind Signature Schemes in the Algebraic Group Model. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13178, pp. 468–497. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_16, https://doi.org/10.1007/978-3-030-97131-1_16

[45] Kastner, J., Loss, J., Xu, J.: The Abe-Okamoto Partially Blind Signature Scheme Revisited. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 13794, pp. 279–309. Springer (2022). https://doi.org/10.1007/978-3-031-22972-5_10, https://doi.org/10.1007/978-3-031-22972-5_10

[46] Katsumata, S., Lai, Y., LeGrow, J.T., Qin, L.: CSI-Otter: Isogeny-Based (Partially) Blind Signatures from the Class Group Action with a Twist. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III. Lecture Notes in Computer Science, vol. 14083, pp. 729–761. Springer (2023). https://doi.org/10.1007/978-3-031-38548-3_24

[47] Katsumata, S., Lai, Y., Reichle, M.: Breaking parallel ROS: implication for isogeny and lattice-based blind signatures. In: Tang, Q., Teague, V. (eds.) Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part I. Lecture Notes in Computer Science, vol. 14601, pp. 319–351. Springer (2024). https://doi.org/10.1007/978-3-031-57718-5_11, https://doi.org/10.1007/978-3-031-57718-5_11

[48] Katz, J., Loss, J., Rosenberg, M.: Boosting the security of blind signature schemes. In: Tibouchi, M., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 13093, pp. 468–492. Springer (2021). https://doi.org/10.1007/978-3-030-92068-5_16, https://doi.org/10.1007/978-3-030-92068-5_16

[49] Kucharczyk, M.: Blind signatures in electronic voting systems. In: Kwiecien, A., Gaj, P., Stera, P. (eds.) Computer Networks - 17th Conference, CN 2010, Ustroń, Poland, June 15-19, 2010. Proceedings. Communications in Computer and Information Science, vol. 79, pp. 349–358. Springer (2010). https://doi.org/10.1007/978-3-642-13861-4_37, https://doi.org/10.1007/978-3-642-13861-4_37

[50] Kuchta, V., LeGrow, J.T., Persichetti, E.: Post-quantum blind signatures from matrix code equivalence. IACR Cryptol. ePrint Arch. p. 274 (2025), https://eprint.iacr.org/2025/274

[51] Le, H.Q., Susilo, W., Khuc, T.X., Bui, M.K., Duong, D.H.: A blind signature from module latices. In: 2019 IEEE Conference on Dependable and Secure Computing, DSC 2019, Hangzhou, China, November 18-20, 2019. pp. 1–8. IEEE (2019). https://doi.org/10.1109/DSC47296.2019.8937613, https://doi.org/10.1109/DSC47296.2019.8937613

[52] Lyubashevsky, V., Nguyen, N.K., Plançon, M.: Efficient lattice-based blind signatures via gaussian one-time signatures. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13178, pp. 498–527. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_17, https://doi.org/10.1007/978-3-030-97131-1_17

[53] Papachristoudis, D., Hristu-Varsakelis, D., Baldimtsi, F., Stephanides, G.: Leakage-resilient lattice-based partially blind signatures. IET Inf. Secur. **13**(6), 670–684 (2019). https://doi.org/10.1049/IET-IFS.2019.0156, https://doi.org/10.1049/iet-ifs.2019.0156

[54] Petzoldt, A., Szepieniec, A., Mohamed, M.S.E.: A practical multivariate blind signature scheme. In: Kiayias, A. (ed.) Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10322, pp. 437–454. Springer (2017). https://doi.org/10.1007/978-3-319-70972-7_25, https://doi.org/10.1007/978-3-319-70972-7_25

[55] Pfitzmann, B.P., Sadeghi, A.R.: Anonymous fingerprinting with direct non-repudiation. In: Advances in Cryptology—ASIACRYPT 2000: 6th International Conference on the Theory and Application of Cryptology and Information Security Kyoto, Japan, December 3–7, 2000 Proceedings 6. pp. 401–414. Springer (2000). https://doi.org/https://doi.org/10.1007/3-540-44448-3_31

[56] del Pino, R., Katsumata, S.: A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13508, pp. 306–336. Springer (2022). https://doi.org/10.1007/978-3-031-15979-4_11, https://doi.org/10.1007/978-3-031-15979-4_11

[57] Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. Journal of cryptology **13**(3), 361–396 (2000)
[58] Qiao, Y., Sun, X.: Canonical forms for matrix tuples in polynomial time. In: 65th IEEE Symposium on Foundations of Computer Science (FOCS) 2024 (2024), arXiv:2409.12457. To appear.
[59] Rostovtsev, A., Stolbunov, A.: Public-key cryptosystem based on isogenies. IACR Cryptol. ePrint Arch. p. 145 (2006), http://eprint.iacr.org/2006/145
[60] Rückert, M.: Lattice-based blind signatures. In: Abe, M. (ed.) Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6477, pp. 413–430. Springer (2010). https://doi.org/10.1007/978-3-642-17373-8_24, https://doi.org/10.1007/978-3-642-17373-8_24
[61] Schnorr, C.: Security of blind discrete log signatures against interactive attacks. In: Qing, S., Okamoto, T., Zhou, J. (eds.) Information and Communications Security, Third International Conference, ICICS 2001, Xian, China, November 13-16, 2001. Lecture Notes in Computer Science, vol. 2229, pp. 1–12. Springer (2001). https://doi.org/10.1007/3-540-45600-7_1, https://doi.org/10.1007/3-540-45600-7_1
[62] Tang, G., Duong, D.H., Joux, A., Plantard, T., Qiao, Y., Susilo, W.: Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 582–612. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_21
[63] Tessaro, S., Zhu, C.: Short pairing-free blind signatures with exponential security. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13276, pp. 782–811. Springer (2022). https://doi.org/10.1007/978-3-031-07085-3_27, https://doi.org/10.1007/978-3-031-07085-3_27
[64] Yi, X., Lam, K.: A new blind ECDSA scheme for bitcoin transaction anonymity. In: Galbraith, S.D., Russello, G., Susilo, W., Gollmann, D., Kirda, E., Liang, Z. (eds.) Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security, AsiaCCS 2019, Auckland, New Zealand, July 09-12, 2019. pp. 613–620. ACM (2019). https://doi.org/10.1145/3321705.3329816, https://doi.org/10.1145/3321705.3329816

A Proof of Proposition 4

Proposition 4. *There is a polynomial-time canonical form algorithm for the left-right action of* $\mathrm{D}(m,q) \times \mathrm{D}(n,q)$ *on* $\mathrm{Mat}(m \times n, q)$.

Proof. We first collect basic notation. Let $A \in \mathrm{Mat}(m \times n, q)$. For each $i \in [m]$ and $j \in [n]$, denote by $R_j \subseteq [m]$ the set of row indices of non-zero entries in the jth column and by $C_i \subseteq [n]$ the set of column indices of non-zero entries in the ith row, both in increasing order. Besides, denote by $R \subseteq [m]$ the set of row indices marked as touched and by $C \subseteq [n]$ the set of column indices marked as touched, which will be initialized as empty sets and stay updated during the algorithm. We also use $A_{i,j}$ to denote the *initial* (i,j)th entry of A, use α_i to denote the ith diagonal entry of the left diagonal matrix acting on A, and use β_j to denote the jth diagonal entry of the right diagonal matrix acting on A. Our goal is to systemically set a left-right action of $\mathrm{D}(m,q) \times \mathrm{D}(n,q)$ to transform any given A into a canonical form.

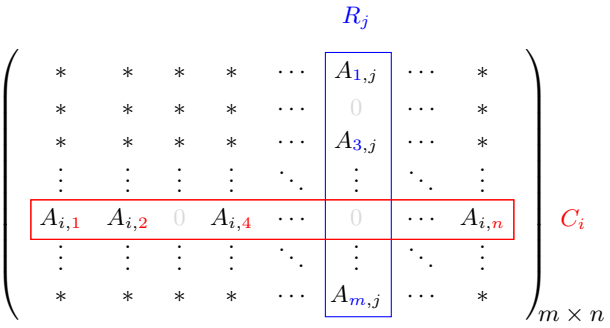

Fig. 6: Illustration of how the sets C_i and R_j are defined. Please note that they consist of the row/column indices of those non-zero entries rather than the values.

Step 1. We start from the first non-zero row in A, say the i_0th row. For each $j \in C_{i_0}$, we can set β_j to scale the jth column such that $\beta_j A_{i_0,j} = 1$. Simultaneously, we mark each index appearing in the aforementioned procedure, and it turns out that $R := \{i_0\}$ and $C := C_{i_0}$ upon completing this step. Note that we won't rescale any column whose index already exists in C during the subsequent steps, so β_j will then be fixed by the equation and the (i_0, j)th entry of A will be fixed at 1 after the scaling of β_j for all $j \in C_{i_0}$.

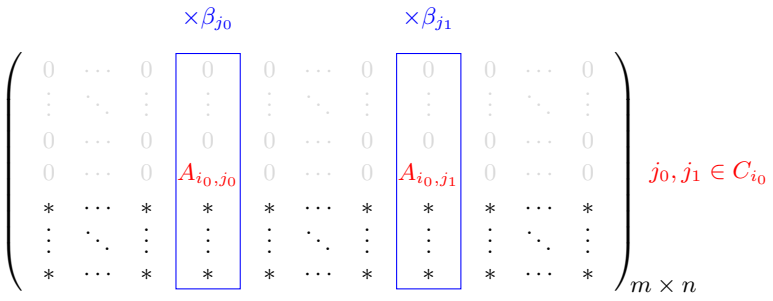

Fig. 7: Illustration of the column scaling in Step 1.

Step 2. Now we focus on the row indices of non-zero entries in each marked column. Specifically, for each $j \in C$, we check whether there are any row indices in $R_j \setminus R$, i.e., the newly appearing row indices to be touched for the first time. At each check, if such a row index exists, say i, we add it into R and set α_i to scale the ith row such that $\alpha_i \beta_j A_{i,j} = 1$ where β_j was fixed in the previous step. Note that once i added into R, we won't rescale the ith row in the future steps, so α_i will then be fixed by this equation and (i, j)th entry of A will be fixed at 1 after the scaling of α_i. Upon completing the loop over C, if the size of R increases, then go to Step 3, because in those newly marked rows, there could be non-zero entries corresponding to some new column indices outside C; if the size of R remains the same, then move on to Step 4.

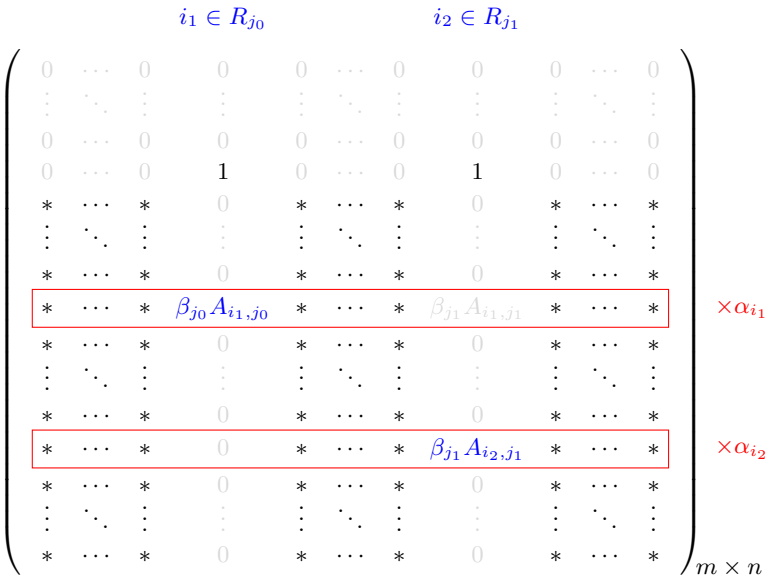

Fig. 8: Illustration of the row scaling in Step 2.

Step 3. In a manner similar to what we just did over C, we proceed to loop over the newly updated row index set R. That is, for each $i \in R$, we check whether there are any column indices in $C_i \setminus C$, i.e., the newly appearing column indices to be touched for the first time. At each check, if such a column index exists, say j', we add it into C and set $\beta_{j'}$ to rescale the j'th column such that $\alpha_i \beta_{j'} A_{i,j'} = 1$ where α_i was fixed in the previous step. Note again that once j' added into C, we won't rescale the j'th column in the future steps, so $\beta_{j'}$ will then be fixed by this equation and the (i, j')th entry of A will be fixed at 1 after the scaling of $\beta_{j'}$. Upon completing the loop over R, if the size of C increases, then go back to Step 2, because in those newly marked columns, there could be non-zero entries corresponding to some new row indices outside R; if the size of C remains the same, then move on to Step 4.

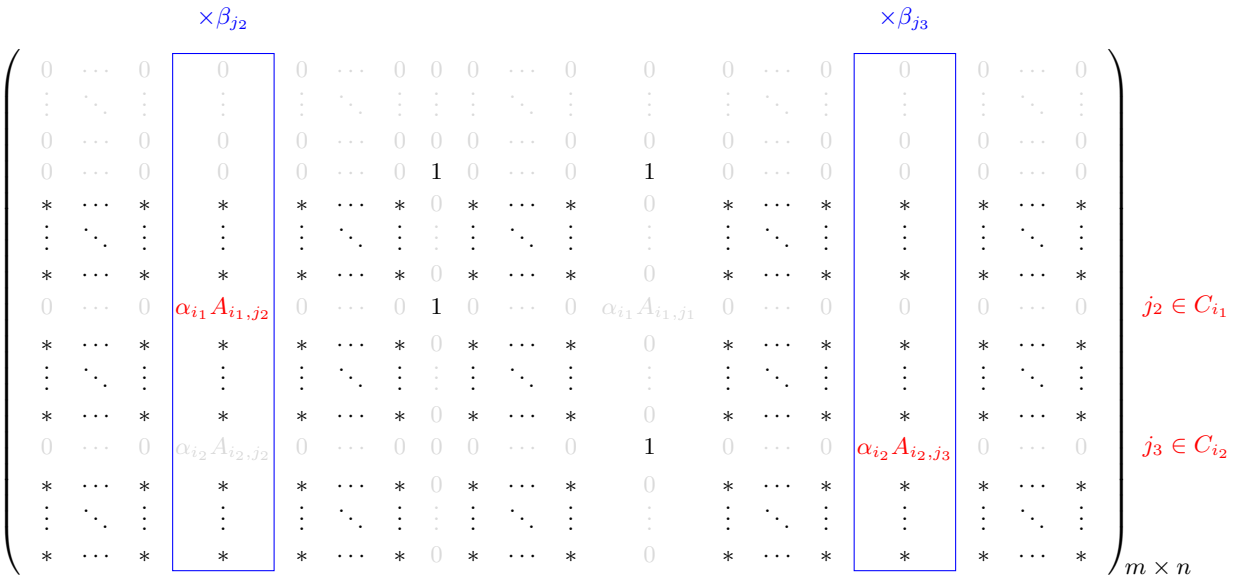

Fig. 9: Illustration of the column scaling in Step 3.

Step 4. After the first three steps, we would have touched and collected some row indices in R and some column indices in C. Starting from a single non-zero entry of A, it is possible to stop at some stage before covering all the indices, i.e., ending up with $R \times C \subset [m] \times [n]$. In this case, we can proceed to the next non-zero row whose index is not in R, which would give us another starting point. Then we can replace (i_0, j_0) in Step 1 with this new starting point and repeat the first three steps. Additionally, we collect the row indices of all such starting points into a set R_S. Denote by R_0 and C_0 the sets of indices corresponding to all-zero rows and all-zero columns, respectively. If $R = [m] \setminus R_0$, we move on to Step 5. Note that our algorithm implies that $R = [m] \setminus R_0$ would also guarantee $C = [n] \setminus C_0$, because we went over each $r \in R$ and non-repetitively added the elements in each C_r to C. It is straightforward to see $C = [n] \setminus C_0$ by the fact that $\bigcup_{r \in [m] \setminus R_0} C_r = [n] \setminus C_0$.

Step 5. Upon reaching this step, we would have fixed all diagonal entries of the two diagonal matrices acting on A, except those corresponding to the indices of all-zero rows/columns in $R_0 \times C_0$ and the indices of the starting non-zero rows in R_S. For $(i, j) \in R_0 \times C_0$, it is free to set α_i and β_j since they are only scaling all-zero rows/columns. For any $i_0 \in R_S$, it is clear that the non-zero entries in the i_0th row are all equal to 1 after the first four steps (according to Step 1), motivating us to simply set $\alpha_{i_0} = 1$ to maintain this structure. This concludes the algorithm, which runs in time $O(mn)$.

Without loss of omitting the all-zero rows/columns, the algorithm described above can give us a certain left-right action of $\mathrm{D}(m,q) \times \mathrm{D}(n,q)$ for any matrix $A \in \mathrm{Mat}(m \times n, q)$. Denote by $c(A)$ the resulting form of A after our algorithm. Now we prove that $c(A)$ is a canonical form of A. It suffices to show that for $A, B \in \mathrm{Mat}(m \times n, q)$, there exist $D_1 \in \mathrm{D}(m,q)$ and $D_2 \in \mathrm{D}(n,q)$ such that $B = D_1 A D_2$ if and only if $c(A) = c(B)$. The if direction is straightforward, because our canonical algorithm can guarantee the left-right actions of $\mathrm{D}(m,q) \times \mathrm{D}(n,q)$ for A and B, and the product of invertible diagonal matrices is still an invertible diagonal matrix. For the only if direction, since the pattern of non-zero entries shared by A and B must be the same, those entries to be scaled to 1 are also located at the same positions in $c(A)$ and $c(B)$. The only difference lies in the row/column scalars which depend on the values of entries. Suppose $c(A) = D_1' A D_2'$ for some $D_1' \in \mathrm{D}(m,q)$ and $D_2' \in \mathrm{D}(n,q)$, then our algorithm implies that $c(B) = c(D_1 A D_2) = D_1' D_1^{-1} (D_1 A D_2) D_2^{-1} D_2' = D_1' A D_2' = c(A)$. □

B Base Sigma Protocol for a 2-Out-of-3 Relation

We suspect that our framework can be extended to a partially blind signature, as in [46]. In fact, In this section, we present an analogous base sigma protocol for a 2-out-of-3 relation to the isogeny-based one in [46, Section 5.1]. This was the first step in constructing the isogeny-based partially blind signature in [46]. Here, we show that it is possible for such a sigma protocol for a 2-out-of-3 relation in the abstract group action setting and hence it is potential for constructing a partially blind signature from abstract group actions, which we will leave as a future work.

A sigma protocol for a 2-out-of-3 relation is a sigma protocol in which the prover knows at least two out of three secrets corresponding to the public statement. Using the same notation as in Section 4, we fix a group action $* : G \times S \to S$ and fix an element $E \in S$. Our public statement is $\mathsf{X} = (A_k^{(c)})$ with $A_k^{(c)} = g_k^c * E$ where $k \in \{0,1,2\}$ and $c \in \{-1,1\}$. The secret g_2 will be known by the signer and user, so we assume that the prover always knows the secret g_2 and proves knowledge of one of g_0 and g_1 in our protocol which is depicted in Fig. 10. For simplicity, we present the protocol for the challenge space $\{-1,1\}$ only. Here we use $[a]_3$ for a modulo 3 in $\mathbb{Z}_3$, and denote by $[0:2]$ the set $\{0,1,2\}$. The correctness, soundness and HVZK can be checked easily, so we omit them.

$\mathcal{P}(\mathsf{X}, \mathsf{W} = (\delta, g_\delta, g_2))$		$\mathcal{V}(\mathsf{X})$
For $j \in \{0,1\}$		
$(h_{\delta,j}, h_{2,j}) \leftarrow\!\!\$\ G^2$		
$Y_{\delta,j} = h_{\delta,j} * E$		
$Y_{2,j} = h_{2,j} * E$		
$(c_{[1-\delta+j]_3}, r_{1-\delta,j}) \leftarrow\!\!\$\ \{-1,1\} \times G$		
$Y_{1-\delta,j} = r_{1-\delta,j} * A_{1-\delta}^{(c_{[1-\delta+j]_3})}$	$\xrightarrow{(Y_{k,j})_{\substack{k\in[0:2]\\ j\in\{0,1\}}}}$	
	$\xleftarrow{c}$	$c \leftarrow\!\!\$\ \{-1,1\}$
$c_{[3-\delta]_3} = c \odot c_{[1-\delta]_3} \odot c_{[2-\delta]_3}$		
For $j \in \{0,1\}$		
$r_{\delta,j} = h_{\delta,j} g_\delta^{-c_{[\delta+j]_3}}$		
$r_{2,j} = h_{2,j} g_2^{-c_{[2+j]_3}}$	$\xrightarrow{(r_{k,j})_{\substack{k\in[0:2]\\ j\in\{0,1\}}}, (c_k)_{k\in[0:2]}}$	Accept if $c = c_0 \odot c_1 \odot c_2$ and
		$\forall (k,j) \in [0:2] \times \{0,1\}$,
		$r_{k,j} * A_k^{(c_{[k+j]_3})} = Y_{k,j}$

Fig. 10: Base 2-Out-of-3 Sigma Protocol