Post-Quantum Blind Signatures from Matrix Code Equivalence

Veronika Kuchta[1], Jason T. LeGrow[2,3], and Edoardo Persichetti[1]

[1] Department of Mathematics, Florida Atlantic University
[2] Department of Mathematics, Virginia Polytechnic Institute and State University
[3] Virginia Tech Center for Quantum Information Science and Engineering

Abstract. We construct a novel code-based blind signature scheme, using the Matrix Equivalence Digital Signature (MEDS) group action. The scheme is built using similar ideas to the Schnorr blind signature scheme and CSI-Otter, but uses additional public key and commitment information to overcome the difficulties that the MEDS group action faces: lack of module structure (present in Schnorr), lack of a quadratic twist (present in CSI-Otter), and non-commutativity of the acting group. We address security concerns related to public key validation, and prove the security of our protocol in the random oracle model, using the security framework of Kastner, Loss, and Xu, under a variant of the Inverse Matrix Code Equivalence problem and a mild heuristic assumption. We also discuss alternative techniques for constructing a code-based blind signature and consider possible parameter sets and corresponding performance metrics.

Keywords: Post-quantum cryptography, Group action-based cryptography, Code-based cryptography, Code equivalence, Blind signatures.

1 Introduction

A *blind signature scheme* is an interactive protocol between a signer and user, which allows the user to obtain the signer's signature on a message of their choice, without revealing any information about that message. These protocols were first proposed by Chaum [13] for digital currency, and they are now known to have applications to e-voting [13], anonymous credentials [9], and blockchain [38]. Blind signatures have been well studied by the cryptographic community and there are many proposed protocols based on the RSA problem, the Diffie-Hellman problem, and their variants [30,32,18,19,12].

Author list in alphabetical order; see https://www.ams.org/about-us/governance/committees/Statement_JointResearchanditsPublication.pdf. This work began at the Coding Theory and Cryptography Workshop held at Virginia Tech's Steger Center for International Scholarship in July 2023. Jason T. LeGrow was supported in part by an award from the Virginia Tech Academy of Data Science and a faculty fellowship from the Commonwealth Cyber Initiative (CCI), an investment in the advancement of cyber R&D, innovation, and workforce development. For more information about CCI, visit www.cyberinitiative.org.

Unfortunately, schemes based on RSA and the Diffie-Hellman problem are susceptible to quantum attacks due to Shor's algorithm [37]. Given the huge global investment currently being made in the development of large-scale quantum computers, we must develop cryptographic protocols that are believed to resist quantum attacks. We call these protocols *post-quantum*. The vast majority of existing post-quantum blind signatures are lattice-based (e.g, [35,4,20,26,28,2]); outside of these, there exist one multivariate blind signature scheme [29], and one isogeny-based scheme [23,24]. There are also some proposals for code-based schemes built from syndrome decoding (e.g., [8]), but these schemes either have impractically large public key/signature sizes, or lack rigorous security proofs.

Many blind signature schemes are built from a Sigma protocol using a variant of the Fiat-Shamir transform. Unfortunately, unlike the "basic" Fiat-Shamir protocol (which transforms any Sigma protocol into an ordinary digital signature), the transforms used to construct blind signatures typically use additional structure present in the underlying Sigma protocol. In particular, Diffie-Hellman-based protocols and many lattice-based protocols admit a *module* structure, which makes their underlying Sigma protocols more amenable to blinding. In contrast, the CSI-FiSh Sigma protocol [6], which is ubiquitous in isogeny-based signature schemes, is built from a much more restrictive *cryptographic group action*, and so it is not clear how to adapt prior techniques to the CSI-FiSh setting. In [23,24], Katsumata, Lai, LeGrow, and Qin note that the group action used in CSI-FiSh is slightly more expressive than a generic group action; in particular, the presence of the *quadratic twist* makes it possible to compute $[\mathfrak{a}^{-1}] * E_0$ from $[\mathfrak{a}] * E_0$. This additional structure is sufficient for the construction of isogeny-based blind signatures.

The cryptographic group action framework was first introduced by Brassard and Yung [10], considered in the isogeny context by Couveignes [15] and Rostovtsev and Stolbunov [34], and formalized and abstracted by Alamati, De Feo, Montgomery, and Patranabis [3]. The cryptographic group framework has been used almost exclusively in the description and analysis of CSIDH [11] and derivative protocols. However, very recent works by Biasse, Micheli, Persichetti and Santini [7] and Chou et al. [14] propose LESS and MEDS, two new signature schemes from *code-based* group actions. Schemes built from these group actions run much faster than isogeny-based schemes, making them more useful in contexts when space is not at a premium. However, these new group actions some drawbacks compared with CSIDH: the groups are non-commutative, which makes it difficult to construct a key exchange protocol (indeed, no such key exchange protocol has been proposed), and these group actions are less expressive than the CSIDH group action, since they possess no module structure and no analogue of the quadratic twist. Though LESS and MEDS can both naturally be transformed into ring signature and linkable ring signature schemes, the non-commutativity and lack of structure beyond that of an abstract group action limit their flexibility, and so it is not obvious how to construct other exotic signature schemes or other advanced functionalities from LESS and MEDS. This difficulty brings us to the main question of this work:

Is it possible to construct a code-based post-quantum blind signature scheme using the LESS or MEDS group action?

1.1 Our Contributions

We answer this question in the affirmative, through the following contributions. Our primary contribution is to **construct a novel code-based blind signature protocol**, built upon the MEDS group action. Our scheme, MEDS-BS, is similar to the construction of CSI-Otter, but crucially avoids the need for the quadratic twist by including additional information in both the public key and in the commitment phase of the interactive signing protocol. We prove the security of our scheme in the model of Kastner, Loss, and Xu [21], establishing that our scheme is secure in the classical random oracle model against polylogarithmic concurrent signing sessions, assuming the difficulty of (a slight variant of) the Inverse Matrix Code Equivalence problem [14, Problem 3].

Our new blind signature scheme also has a restriction on the kind of secret keys that can be used. In particular, a MEDS public key is a single matrix code $X^{(1)}$. This code should be equivalent to a fixed global public code $X^{(0)}$; that is, $X^{(1)} = (A, B) * X^{(0)}$ for some $(A, B) \in \mathrm{GL}_m(\mathbb{F}_p) \times \mathrm{GL}_n(\mathbb{F}_p)$. This equivalence is implicitly verified in the Sigma protocols used in the MEDS-based constructions of [14]. Our protocol requires that the secret key matrices be symmetric or antisymmetric, and requires additional public key information: in particular, along with $X^{(1)}$, the signer must provide $X^{(-1)} := (A^{-1}, B^{-1}) * X^{(0)}$. The analogous piece of information in CSI-Otter need not be included explicitly, thanks to the *quadratic twist*. Since MEDS has no known equivalent feature, we simply include the information in the public key. This introduces a wrinkle: a dishonest signer could potentially create a malformed public key, which could lead to violations of blindness. To counteract this, our next contribution is to **introduce novel proofs of public key well-formedness** for MEDS-BS. More specifically, we introduce zero-knowledge proofs of knowledge for the relations

$$\mathcal{R}^{(\pm 1)}_{X^{(0)}} = \left\{ \begin{pmatrix} \mathsf{X} = (X^{(1)}, X^{(-1)}), \\ \mathsf{W} = (A, B) \end{pmatrix} : \begin{array}{c} X^{(1)} = (A, B) * X^{(0)}, \\ X^{(-1)} = (A^{-T}, B^{-T}) * X^{(0)} \end{array} \right\}$$

$$\mathcal{R}^{\mathbb{S}}_{X^{(0)}} = \left\{ \begin{pmatrix} \mathsf{X} = X^{(1)}, \\ \mathsf{W} = (A, B) \end{pmatrix} : \begin{array}{c} X^{(1)} = (A, B) * X^{(0)} \\ A^T = \pm A \\ B^T = \pm B \end{array} \right\}$$

which, when used together, prove that a pair $(X^{(1)}, X^{(-1)})$ is a well-formed public key. These proofs of knowledge may be of independent interest for researchers developing other protocols based on MEDS.

1.2 Overview

In this section we provide a brief overview of our journey towards constructing a code-based blind signature scheme.

Schnorr Blind Signatures. Schnorr blind signatures, a conceptual predecessor to CSI-Otter and to the blind signature schemes we propose in this work, are obtained via a modified version of the Schnorr Sigma protocol [36]. The basic Schnorr sigma protocol is depicted in Figure 1.

Alice $\mathsf{sk}, \mathsf{pk} = g^{\mathsf{sk}}$		Bob pk
$r \xleftarrow{\$} \{0, 1, \ldots, p-1\}$		
$\mathsf{comm} \leftarrow g^r$	$\xrightarrow{\mathsf{comm}}$	
	$\xleftarrow{\mathsf{chal}}$	$\mathsf{chal} \xleftarrow{\$} \{0, 1, \ldots, p-1\}$
$\mathsf{resp} \leftarrow r + \mathsf{chal} \cdot \mathsf{sk}$	$\xrightarrow{\mathsf{resp}}$	
		Accept if $g^{\mathsf{resp}} = \mathsf{comm} \cdot \mathsf{pk}^{\mathsf{chal}}$

Fig. 1. The Schnorr Sigma protocol over the group $G = \langle g \rangle$ with $|G| = p$.

From Schnorr to Isogeny-Based Blind Signatures. The verification procedure of Schnorr signatures crucially uses the fact that the public keys and commitments come from a group G, so that $\mathsf{comm} \cdot \mathsf{pk}^{\mathsf{chal}}$ can be computed. This group structure is also used in Bob's blinding procedure. Unfortunately, it is precisely this group structure that makes Schnorr signatures vulnerable to quantum attacks using Shor's algorithm. The *cryptographic group action* framework yields an alternative to Schnorr's construction with many of the same properties, but which is believed to resist quantum attacks (for certain group actions). Given a group action $G \curvearrowright X$, it is easy to define a "Schnorr-like" Sigma protocol [6].

For blind signatures, moving to a generic group action protocol presents a new challenge. In fact, Bob will receive a commitment comm from Alice, and must blind this commitment to construct comm', which is independent from comm. He will derive $\mathsf{chal}' = \mathsf{H}(\mathsf{comm}', \mathsf{msg})$, and then he must construct an independent challenge chal. Then, using Alice's response resp to challenge chal, he must construct a response resp' which yields a valid transcript $(\mathsf{comm}', \mathsf{chal}', \mathsf{resp}')$.

In this more restrictive setting, it seems that Bob is limited to blinding the commitment by acting by a random group element s: $\mathsf{comm}' \leftarrow s * \mathsf{comm}$. Similarly, Bob can construct chal as $\mathsf{chal} = \mathsf{chal}' + r$ with $r \xleftarrow{\$} \{0, 1\}$. But then when $r = 1$ (so that $\mathsf{chal}' \neq \mathsf{chal}$), Bob will not be able to construct the correct response, as we will now show. Suppose without loss of generality that $\mathsf{chal} = 0$, so that $\mathsf{chal}' = 1$. Bob will receive $\mathsf{resp} = g$, which satisfies $\mathsf{resp} * x_0 = \mathsf{comm}$. Bob must construct resp' which satisfies $\mathsf{resp} * \mathsf{pk} = \mathsf{comm}' = s * \mathsf{comm}$. But then

$$\mathsf{pk} = (\mathsf{resp}')^{-1} * (s * \mathsf{comm}) = \left((\mathsf{resp}')^{-1} \cdot s \cdot \mathsf{resp}\right) * x_0$$

so that $(\mathsf{resp}')^{-1} \cdot s \cdot \mathsf{resp}$ is an equivalent secret key to Alice's—thus, Bob must be unable to construct such resp'.

Katsumata, Lai, LeGrow, and Qin [23,24] proposed a technique to overcome this difficulty by working in the setting of CSI-FiSh [6] and exploiting the *quadratic twist*; this notion, denoted $x^{(-1)}$ for $x \in X$, given carefully chosen $x_0 \in X$, allows anyone to efficiently compute $g^{-1} * x_0$ from $g * x_0$ alone. The quadratic twist gives rise to a richer blinding procedure which, when combined with a modified challenge space, yields CSI-Otter, the blind signature scheme of [23,24]. A simplified version of CSI-Otter—which includes this new challenge space and blinding procedure, but omits other, more standard, aspects of the secure construction—is depicted in Figure 2.

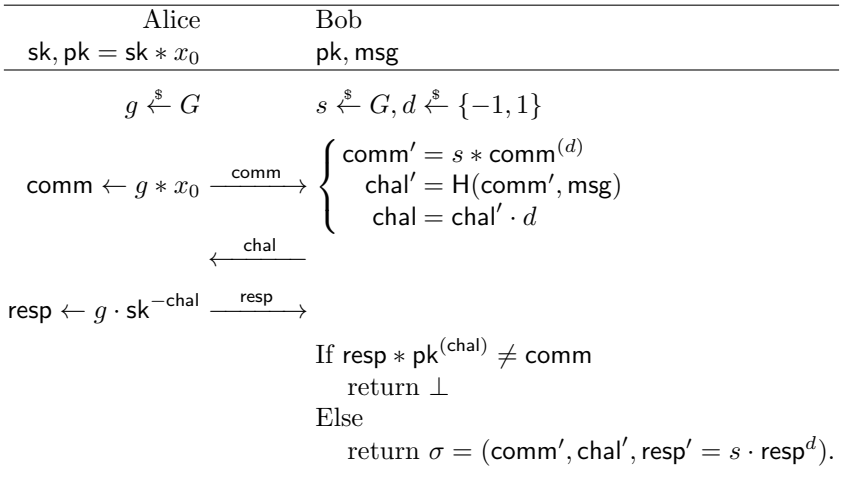


Fig. 2. A simplified version of CSI-Otter.

Structural Difficulties in the Code-Based Setting. The cryptographic group action framework was designed as an abstraction that encompasses the functionality of CSI-FiSh without reference to the particular group action used in that setting, but in principle any abelian group action can be used in place of the CSI-FiSh action. Moreover, many protocols—including Fiat-Shamir-based digital signatures—can be instantiated with nonabelian group actions. In particular, the Linear Equivalence Signature Scheme (LESS) [7] and the Matrix Equivalence Digital Signature (MEDS) [14] use matrix groups acting as isomorphisms on sets of error-correcting codes as the base for the Sigma protocol.

In the case of MEDS, the group $G = \mathrm{GL}_m(\mathbb{F}_q) \times \mathrm{GL}_n(\mathbb{F}_q)$ acts on the set $\mathcal{X}$ of k-dimensional $m \times n$ matrix codes defined over $\mathbb{F}_q$. The action is by multiplication on the left and right simultaneously: $(A, B) * \mathcal{C} = A\mathcal{C}B^T = \{AXB^T : X \in \mathcal{C}\}$. The MEDS group action, unfortunately, cannot be used as a drop-in replacement for the CSI-FiSh group action in the CSI-Otter Sigma protocol. The first complication is straightforward: there is no quadratic twist analogue, and thus Bob has no way to compute $\mathsf{comm}^{(-1)} = g^{-1} * x_0$ or $\mathsf{pk}^{(-1)} = \mathsf{sk}^{-1} * x_0$. But even if Bob

could compute them, the protocol is not correct for nonabelian group actions. To see this, suppose that Alice and Bob act honestly, and that Bob chooses $d = -1$. Then the transcript $(\mathsf{comm}, \mathsf{chal}, \mathsf{resp})$ will satisfy $\mathsf{resp} * \mathsf{pk}^{(\mathsf{chal})} = \mathsf{comm}$. But in Bob's signature, we will have $\mathsf{resp}' = s \cdot \mathsf{resp}^{-1}$, so that

$$\mathsf{resp}' * \mathsf{pk}^{(\mathsf{chal}')} = (s \cdot \mathsf{resp}^{-1} \cdot \mathsf{sk}^{-\mathsf{chal}}) * x_0 = (s \cdot \mathsf{sk}^{\mathsf{chal}} \cdot g^{-1} \cdot \mathsf{sk}^{-\mathsf{chal}}) * x_0 \neq \mathsf{comm}'.$$

The problem that has arisen is that inversion is order reversing, and so in the nonabelian setting the $\mathsf{sk}^{\mathsf{chal}}$ which appears in resp^{-1} does not cancel with the $\mathsf{sk}^{-\mathsf{chal}}$ that appears in $\mathsf{pk}^{(-\mathsf{chal})}$, because g^{-1} appears between them.

Overcoming Noncommutativity and Lack of Quadratic Twist. Setting aside the issue of computing $\mathsf{comm}^{(d)}$ and $\mathsf{pk}^{(\mathsf{chal})}$, we first attempt to tackle the problem of noncommutativity. The MEDS group $G = \mathrm{GL}_m(\mathbb{F}_q) \times \mathrm{GL}_n(\mathbb{F}_q)$ admits another order-reversing involution given by transposition in both components: $(A, B) \mapsto (A^T, B^T)$. Combining inversion with transposition yields an order-*preserving* involution $(A, B) \mapsto (A^{-T}, B^{-T})$. Writing $g = (A, B)$, we denote $g^{(1)} = g = (A, B)$ and $g^{(-1)} = (A^{-T}, B^{-T})$. If we replace all inversions in Figure 2 with our order-preserving involution and redefine $\mathsf{comm}^{(d)}$ and $\mathsf{pk}^{(\mathsf{chal})}$ as $\mathsf{comm}^{(d)} = g^{(d)} * x_0$ and $\mathsf{pk}(\mathsf{chal}) = \mathsf{sk}^{(d)} * \mathsf{chal}$, then Bob's signature satisfies

$$\begin{aligned}\mathsf{resp}' * \mathsf{pk}^{(\mathsf{chal}')} &= (s \cdot g^{(d)} \cdot \mathsf{sk}^{(-d \cdot \mathsf{chal})}) * (\mathsf{sk}^{(d \cdot \mathsf{chal})} * x_0) \\ &= (s \cdot g^{(d)}) * \left((\mathsf{sk}^{(-d \cdot \mathsf{chal})} \cdot \mathsf{sk}^{(d \cdot \mathsf{chal})}) * x_0\right)\end{aligned}$$

At this point, we must consider the product $\mathsf{sk}^{(-d \cdot \mathsf{chal})} \cdot \mathsf{sk}^{(d \cdot \mathsf{chal})}$. Note that either $-d \cdot \mathsf{chal} = 1$ and $d \cdot \mathsf{chal} = -1$ or vise versa; without loss of generality, suppose we are in the first case. Writing $\mathsf{sk} = (A, B)$, we have

$$\mathsf{sk}^{(-d \cdot \mathsf{chal})} \cdot \mathsf{sk}^{(d \cdot \mathsf{chal})} = (A, B) \cdot (A^{-T}, B^{-T}) = (AA^{-T}, BB^{-T}).$$

If we can guarantee that (AA^{-T}, BB^{-T}) is in the stabilizer of x_0, then we will have $(\mathsf{sk}^{(-d \cdot \mathsf{chal})} \cdot \mathsf{sk}^{(d \cdot \mathsf{chal})}) * x_0 = x_0$, so that

$$\mathsf{resp}' * \mathsf{pk}^{(\mathsf{chal}')} = (s \cdot g^{(d)}) * x_0 = s \cdot \mathsf{comm}^{(d)} = \mathsf{comm}'.$$

This can be achieved by enforcing that A and B be (anti)symmetric, so that $AA^{-T} = \pm I_m$ and $BB^{-T} = \pm I_n$. Thus, by slightly retooling the basic CSI-Otter protocol and enforcing a constraint on the secret key, we obtain a protocol that is correct—provided that Bob has access to $\mathsf{comm}^{(-1)}$ and $\mathsf{pk}^{(-1)}$. In the CSI-FiSh setting Bob computed $\mathsf{comm}^{(-1)}$ and $\mathsf{pk}^{(-1)}$ using the quadratic twist; however, they can actually be included as part of Alice's first message and public key, respectively. At the cost of some additional communication, Bob will be able to use $\mathsf{comm}^{(-1)}$ and $\mathsf{pk}^{(-1)}$ without having to compute them himself.

While the above approach plausibly leads to a correct and secure blind signature protocol, it has introduced some new problems: the secret key matrices must be (anti)symmetric, and the public key and commitment must satisfy

$$\mathsf{pk} = \left(\mathsf{pk}^{(1)} = \mathsf{sk}^{(1)} * x_0,\ \mathsf{pk}^{(-1)} = \mathsf{sk}^{(-1)} * x_0\right)$$

$$\mathsf{comm} = \left(\mathsf{comm}^{(1)} = g^{(1)} * x_0,\, \mathsf{comm}^{(-1)} = g^{(-1)} * x_0\right).$$

There is no obvious way for Bob to check these conditions, and so a dishonest signer could construct a secret key, public key, or commitment that does not satisfy the structural requirements, which could lead to security vulnerabilities. In the isogeny-based setting this was not a problem, because the secret key did not have any special structure and Bob was not relying on Alice for $\mathsf{pk}^{(-1)}$ and $\mathsf{comm}^{(-1)}$, since he could construct them himself. To overcome this potential security issue, we design a new noninteractive proof protocol that ensures that Alice's secret key matrices are (anti)symmetric, and her public key is well-formed. As for the commitments, Bob checks their well-formedness when the response is revealed, so no additional noninteractive proofs are required.

The Security Model and Security Proof. The Schnorr blind signature, simplified CSI-Otter protocol we describe in Figure 2, and blind signature protocol we describe in the preceding sections are not known to be secure in the random oracle model (ROM) alone [5]. Several variants have been proposed [31,33,1], and the OR-proof construction of [1] which was used in CSI-Otter also generalizes to the setting of this work. We prove the security of our protocol in a slightly modified version of the model of Kastner, Loss, and Xu [21]. We will recap security notions in Section 2.2.

2 Preliminaries

2.1 Sigma Protocols and Blind Signatures

Definition 2.1 (Sigma Protocol). *A Sigma protocol for an NP relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ is a three-move interactive protocol which consists of two PPT algorithms $\mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2), \mathsf{V}$ which works as follows:*

1. *On input $(\mathsf{X}, \mathsf{W}) \in \mathcal{R}$, the prover runs algorithm P_1 to obtain a commitment comm and state state. It sends comm to the verifier.*
2. *On input comm the verifier chooses a challenge $\mathsf{chal} \in \mathcal{C}$ uniformly at random and sends it to P.*
3. *On input $(\mathsf{chal}, \mathsf{state})$, the prover runs algorithm P_2 to construct a response resp which it sends to the verifier.*
4. *The verifier runs $\mathsf{V}(\mathsf{X}, \mathsf{comm}, \mathsf{chal}, \mathsf{resp})$ and outputs a bit b, indicating whether the proof is valid (1) or invalid (0).*

A correct and secure Sigma protocol satisfies the following properties:

Correctness: If the prover and verifier follow the protocol description, then the verifier will output 1 with probability 1.

Honest Verifier Zero-Knowledge (HVZK): There exists a PPT simulator Sim that, given a statement X outputs a valid transcript $(\mathsf{comm}, \mathsf{chal}, \mathsf{resp})$, in such a way that the simulator's distribution of outputs is identical to the distribution of outputs of honest transcripts.

Special Soundness: There exists a deterministic polynomial-time extractor Ext that, given valid transcripts $(\mathsf{comm}, \mathsf{chal}_0, \mathsf{resp}_0)$ and $(\mathsf{comm}, \mathsf{chal}_1, \mathsf{resp}_1)$ with respect to X, satisfying $\mathsf{chal}_0 \neq \mathsf{chal}_1$, outputs $\hat{\mathsf{W}}$ such that $(\mathsf{X}, \hat{\mathsf{W}}) \in \mathcal{R}$.

A related property is witness indistinguishability, which requires transcripts coming from different witnesses to the same statement to be indistinguishable.

Definition 2.2 ([23]). *An NP relation $\mathcal{R}$ is associated with an instance generator (IG) if IG, given as input the security parameter 1^n, outputs a statement-witness pair (X, W). The instance generator is hard if the following holds for any PPT adversary $\mathcal{A}$: $\Pr[(\mathsf{X}, \mathsf{W}) \leftarrow \mathsf{IG}(1^n), \mathsf{W}' \leftarrow \mathcal{A}(\mathsf{X}) : (\mathsf{X}, \mathsf{W}') \in \mathcal{R}] = \mathsf{negl}(n)$.*

We are now prepared to define blind signatures.

Definition 2.3 (Blind Signature). *A blind signature scheme consists of the four algorithms $\Sigma_{\mathsf{BS}} = (\mathsf{BS.KGen}, \mathsf{BS.S}, \mathsf{BS.U}, \mathsf{BS.Verify})$ which are as follows:*

$\mathsf{BS.KGen}(1^\lambda)$: *On input a security parameter 1^λ it outputs a private signing key sk and a public verification key pk*

$\mathsf{BS.S} = (\mathsf{BS.S}_1, \mathsf{BS.S}_2)$: *An interactive protocol consisting of two phases.*
- $\mathsf{BS.S}_1(\mathsf{sk})$*: On input a secret key sk, it outputs an internal state $\mathsf{state}_\mathsf{S}$ and a first-sender response $\rho_{\mathsf{S},1}$.*
- $\mathsf{BS.S}_2(\mathsf{state}_\mathsf{S}, \rho_\mathsf{U})$*: On input a state $\mathsf{state}_\mathsf{S}$, and a user message ρ_U, it outputs a second-sender response $\rho_{\mathsf{S},2}$.*

$\mathsf{BS.U} = (\mathsf{BS.U}_1, \mathsf{BS.U}_2)$: *An interactive protocol consisting of two phases:*
- $\mathsf{BS.U}_1(\mathsf{pk}, \mathsf{msg}, \rho_{\mathsf{S},1})$: *On input a public key pk, a message msg, and a response $\rho_{\mathsf{S},1}$, it outputs a user state $\mathsf{state}_\mathsf{U}$ and a user message ρ_U.*
- $\mathsf{BS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2})$: *On input a user state $\mathsf{state}_\mathsf{U}$ and a second-sender response $\rho_{\mathsf{S},2}$ it outputs a signature σ on a message msg.*

$\mathsf{BS.Verify}(\mathsf{pk}, \mathsf{msg}, \sigma)$: *On input a public key pk, a message msg and a signature σ, it outputs 1 if σ is a valid signature on msg or 0 otherwise.*

2.2 Security Definitions for Blind Signatures

A correct and secure blind signature scheme must satisfy the following properties:

Completeness: A blind signature scheme Σ_{BS} is complete if for any $\lambda \in \mathbb{N}$, $\mathsf{msg} \in \mathcal{M}$, and $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{BS.KGen}(1^\lambda)$ we have

$$\mathbb{P}\left[\mathsf{BS.Verify}(\mathsf{pk}, \mathsf{msg}, \sigma) = 1 \;\middle|\; \begin{array}{r} (\mathsf{state}_\mathsf{S}, \rho_{\mathsf{S},1}) \leftarrow \mathsf{BS.S}_1(\mathsf{sk}) \\ (\mathsf{state}_\mathsf{U}, \rho_\mathsf{U}) \leftarrow \mathsf{BS.U}_1(\mathsf{pk}, \mathsf{msg}, \rho_{\mathsf{S},1}) \\ \rho_{\mathsf{S},2} \leftarrow \mathsf{BS.S}_2(\mathsf{state}_\mathsf{S}, \rho_\mathsf{U}) \\ \sigma \leftarrow \mathsf{BS.U}_2(\mathsf{state}_\mathsf{U}, \rho_{\mathsf{S},2}) \end{array}\right] = 1.$$

Blindness: For a three-move blind signature scheme Σ_{BS}, we define the blindness game $\mathsf{Blind}_{\mathsf{BS}}$ with an adversary $\mathcal{S}$ (playing the signer) as follows:

Setup: Sample a bit $b \xleftarrow{\$} \{0, 1\}$. Then run $\mathcal{S}$ on input 1^λ.

Online Phase: When $\mathcal{S}$ outputs messages $\tilde{\mathsf{msg}}_0, \tilde{\mathsf{msg}}_1$ and a public key pk, the game checks if pk is valid and if so, it assigns $\mathsf{msg}_0 := \tilde{\mathsf{msg}}_b$, $\mathsf{msg}_1 := \tilde{\mathsf{msg}}_{1-b}$. If pk is not valid, the game aborts and outputs 0. $\mathcal{S}$ is given access to oracles $\mathcal{O}_{\mathsf{U}_1}$ and $\mathcal{O}_{\mathsf{U}_2}$, which work as follows:

- **Oracle** $\mathcal{O}_{\mathsf{U}_1}$: On input a bit $b' \in \{0,1\}$ and a S_1 response $\rho_{\mathsf{S},1}$, if the session b' is not yet open, the oracle marks session b' as open and generates a state and a challenge as $(\mathsf{state}_{\mathsf{U},b'}, \rho_{\mathsf{U},b}) \xleftarrow{\$} \mathsf{BS.U}_1(\mathsf{pk}, \mathsf{msg}_{b'}, \rho_{\mathsf{S},1})$. It returns $\rho_{\mathsf{U},b}$ to $\mathcal{S}$. Otherwise, it returns $\perp$.
- **Oracle** $\mathcal{O}_{\mathsf{U}_2}$: On input a bit $b' \in \{0,1\}$ and a second-signer message $\rho_{\mathsf{S},2,b'}$, if the session b' is open, it computes a signature $\sigma_{b'} := \mathsf{BS.U}_2(\mathsf{pk}, \mathsf{state}_{\mathsf{U},b'}, \rho_{\mathsf{S},2,b'})$. It marks session b' as closed and saves $\sigma_{b'}$. If both sessions are closed and they have produced signatures, the oracle outputs the two signatures σ_0, σ_1 to $\mathcal{S}$.

Output Determination. If both sessions are closed and produced signatures, the game outputs 1 iff $\mathcal{S}$ outputs a bit $b' = b$. Otherwise, return 0.

We define the advantage of $\mathcal{S}$ as $\mathsf{Adv}^{\mathsf{Blind}}_{\mathcal{S},\mathsf{BS}} = |\Pr[\mathsf{Blind}_{\mathsf{BS}} = 1] - 1/2|$, where the probability goes over the randomness of the game as well as the randomness of the adversary $\mathcal{S}$. We say the scheme Σ_{BS} is (t,ϵ)-blind if for any adversary $\mathcal{S}$ running in time t, $\mathsf{Adv}^{\mathsf{Blind}}_{\mathcal{S},\mathsf{BS}} \leq \epsilon$.

One-More Unforgeability: Let $\Sigma_{\mathsf{BS}} = (\mathsf{BS.KGen}, \mathsf{BS.S}, \mathsf{BS.U}, \mathsf{BS.Verify})$ be a blind signature and λ be the security parameter. We define the ℓ-one more unforgeability game ℓ-$\mathsf{OMUF}_{\mathsf{PBS}}$ with an adversary $\mathcal{A}$ (playing the user) as follows:

Setup: Sample a pair of keys $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{BS.KGen}(1^\lambda)$. Initialize $\ell_{\mathsf{closed}} := 0$ and run $\mathcal{A}$ on input pk.

Online Phase: $\mathcal{A}$ is given access to oracles $\mathcal{O}_{\mathsf{S}_1}$ and $\mathcal{O}_{\mathsf{S}_2}$, which work as follows:

- **Oracle** $\mathcal{O}_{\mathsf{S}_1}$: The oracle samples a fresh session identifier sid. It sets $\mathsf{open}_{\mathsf{sid}} := \mathsf{true}$ and generates $(\mathsf{state}_{\mathsf{S},\mathsf{sid}}, \rho_{\mathsf{S},1}) \xleftarrow{\$} \mathsf{BS.S}_1(\mathsf{sk})$. Then it returns the response $\rho_{\mathsf{S},1}$ to $\mathcal{A}$.
- **Oracle** $\mathcal{O}_{\mathsf{S}_2}$: If $\ell_{\mathsf{closed}} < \ell$ the oracle takes as input a user message ρ_{U} and a session identifier sid. If $\ell_{\mathsf{closed}} \geq \ell$ or $\mathsf{open}_{\mathsf{sid}} = \mathsf{false}$, it returns $\perp$. Otherwise, it sets $\ell_{\mathsf{closed}}++$ and $\mathsf{open}_{\mathsf{sid}} := \mathsf{false}$. Then it computes the response $\rho_{\mathsf{S},2} \xleftarrow{\$} \mathsf{BS.S}_2(\mathsf{state}_{\mathsf{S},\mathsf{sid}}, \rho_{\mathsf{U}})$ and returns $\rho_{\mathsf{S},2}$ to $\mathcal{A}$.

Output Determination. When $\mathcal{A}$ outputs distinct tuples $(\mathsf{msg}_1, \sigma_1), \ldots, (\mathsf{msg}_k, \sigma_k)$, return 1 if $k \geq \ell_{\mathsf{closed}}+1$ and $\mathsf{BS.Verify}(\mathsf{pk}, \mathsf{msg}_i, \sigma_i) = 1$ for all $1 \leq i \leq k$. Otherwise, return 0.

The advantage of $\mathcal{A}$ is $\mathsf{Adv}^{\ell-\mathsf{OMUF}_{\mathsf{BS}}}_{\mathcal{A},\mathsf{BS}}(\lambda) = \Pr[\ell - \mathsf{OMUF}_{\mathsf{BS}} = 1]$, where the probability goes over the randomness of the game as well as the randomness of the adversary $\mathcal{A}$. We say the scheme Σ_{BS} is ℓ-one-more unforgeable if for any adversary $\mathcal{A}$ that makes at most ℓ queries to $\mathcal{O}_{\mathsf{S}_1}$, we have $\mathsf{Adv}^{\ell\text{-}\mathsf{OMUF}_{\mathsf{PBS}}}_{\mathcal{A},\mathsf{BS}}(\lambda) \leq \mathsf{negl}(\lambda)$.

2.3 Proof Technique for Blind Signatures

In a blind signature scheme built from a Sigma protocol, the role of the signer is played by the prover, while the role of the user is played by the Sigma protocol's

verifier. To prove one-more unforgeability, we reduce the unforgeability property to the special soundness of the underlying Sigma protocol. While in the standard Fiat-Shamir-like blind signatures the simulation of a signature follows from the HVZK property of the scheme, in our case this is not possible since the adversary is controlling the challenge. To circumvent these difficulties and to simulate the conversation between the adversary and a user we use a public key with two valid secret keys, where the underlying hard problem is embedded into one of these keys and the other is a tag key which is used for simulation.

The general idea for extracting the witness is borrowed from [21]. The reduction in the security proof uses the widely-known forging technique for rewinding the adversary and solving the underlying hard problem. First we define a deterministic wrapper which provides a simplified interface to the reduction. A wrapper $\mathcal{W}$ takes as input an instance I, a random tape rand and a random hash vector $\mathbf{h}$. The reduction will run the wrapper on inputs $(I, \mathsf{rand}, \mathbf{h})$ and output an index $J \in [|\mathbf{h}|]$. It then resamples the vector $\mathbf{h}$ from position J and obtains $\tilde{\mathbf{h}}$, while the first $J-1$ positions are kept unchanged. The reduction returns the wrapper on inputs $(I, \mathsf{rand}, \tilde{\mathbf{h}})$. The two tuples $(I, \mathsf{rand}, \mathbf{h})$ and $(I, \mathsf{rand}, \tilde{\mathbf{h}})$ are called partnering runs and the witness extracted from these runs is independent the fact which witness was used by the reduction. Next, we define the basic definitions needed for the security proof:

Definition 2.4 (Instances [23]). *Assume the public key of our blind signature scheme has exactly two corresponding secret keys* $\mathsf{sk}_0 = (0, W_0), \mathsf{sk}_1 = (1, W_1)$. *There are two types of instances I: A* 0*-side instance consists of* sk_0 *(and* sk_1*, respectively) and the randomness used by the honest signer algorithm when the secret key is fixed to* sk_0 *(or* sk_1*, respectively)*

Let $\mathsf{Succ} := \{(I, \mathsf{rand}, \mathbf{h}) \,|\, \mathcal{W}(I, \mathsf{rand}, \mathbf{h}) \neq \perp\}$ be the set of all successful tuples used as input to the wrapper $\mathcal{W}$.

Definition 2.5 (Query Transcript [21]). *Consider the wrapper $\mathcal{W}$ running on input $(I, \mathsf{rand}, \mathbf{h})$. The query transcript, denoted by $\mathbf{e}(I, \mathsf{rand}, \mathbf{h})$ is the vector of signing queries queries e_{sid} made to $\mathcal{O}_{\mathsf{sign}_2}$ by the adversary $\mathcal{A}$.*

Definition 2.6 (Full Transcript [21]). *Consider the wrapper $\mathcal{W}$ running on input tuple $(I, \mathsf{rand}, \mathbf{h})$. We denote by $\mathsf{tr}(I, \mathsf{rand}, \mathbf{h})$ the transcript produced between $\mathcal{W}$ and the adversary $\mathcal{A}$.*

Definition 2.7 (Successful Forking [21]). *Two successful input tuples $(I, \mathsf{rand}, \mathbf{h})$, $(I, \mathsf{rand}, \hat{\mathbf{h}}) \in \mathsf{Succ}$ fork from each other at index $i \in [\ell + 1]$ if $\mathbf{h}_{[i-1]} = \hat{\mathbf{h}}_{[i-1]}$, but $h_i \neq h_i$. The set of hash vectors $(\mathbf{h}, \hat{\mathbf{h}})$ whose input tuples fork at index i is denoted as $F_i(I, \mathsf{rand})$.*

Definition 2.8 (Partners [21]). *Two successful tuples $(I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\mathbf{h}})$ are partners at index $i \in [\ell + 1]$ if the following holds:*

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

Let $\mathsf{prt}_i(I, \mathsf{rand})$ *denote the set of hash vector pairs* $(\mathbf{h}, \hat{\mathbf{h}})$ *such that* $(I, \mathsf{rand}, \mathbf{h})$ *and* $(I, \mathsf{rand}, \mathbf{h})$ *are partners.*

Definition 2.9 (Triangles [21]). *A triangle at index* $i \in [\ell+1]$ *with respect to* I, rand *is a tuple of three successful tuples in the following set:*

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

For a triangle $((I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\mathbf{h}}), (I, \mathsf{rand}, \hat{\hat{\mathbf{h}}})) \in \Delta_i(I, \mathsf{rand})$ *we call the pair of tuples* $((I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\mathbf{h}}))$ *the base, and* $((I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\hat{\mathbf{h}}}))$ *and* $((I, \mathsf{rand}, \hat{\mathbf{h}}), (I, \mathsf{rand}, \hat{\hat{\mathbf{h}}}))$ *the sides.*

Next, we define a transformation from b-side to $(1-b)$-side instances.

Definition 2.10 (Mapping Instances via transcript [21,23]). *For* $(I, \mathsf{rand}, \mathbf{h}) \in \mathsf{Succ}$*, we define* $\Phi_{\mathsf{rand},\mathbf{h}}$ *as a function that maps a* 0*-side instance* I *(resp.* 1*-side instance* I*) to a* 1*-side instance* I' *(resp.* 0*-side instance* I'*).*

If $\Phi_{\mathsf{rand},\mathbf{h}}$ is a bijection preserving transcripts for rand and $\mathbf{h}$, then a partner tuple with a b-side instance maps to an instance with a $(1-b)$-side according to [21, Corollary 1 and Lemma 3]. This would imply that the extracted witness from the partner tuple is independent of the secret key of the reduction.

The original framework of Kastner, Loss, and Xu defines a notion of witness extractor, and it is slightly generalized in [23]. We present this in Definition 2.11.

Definition 2.11 ([23]). *Fix* I, rand *and let* $\mathbf{h}, \hat{\mathbf{h}} \in F_i(I, \mathsf{rand})$ *for some* $i \in [\ell+1]$*. Moreover, denote by* $\sigma_i, \hat{\sigma}_i$ *the signatures that correspond to* $h_i, \hat{h}_i$*, respectively. We say that the deterministic algorithms* $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ *are* witness extractors *if* $(\mathsf{Ext}_0(\sigma_i, \hat{\sigma}_i), \mathsf{Ext}_1(\sigma_i, \hat{\sigma}_i)) \in \{(\mathsf{sk}_0, \perp), (\perp, \mathsf{sk}_1), (\mathsf{sk}_0, \mathsf{sk}_1)\}$*. For* $b \in \{0,1\}$*, we say that the* b*-side witness can be extracted from* $(I, \mathsf{rand}, \mathbf{h})$ *and* $(I, \mathsf{rand}, \hat{\mathbf{h}})$ *at the index* i *if* $\mathsf{Ext}_b(\sigma_i, \hat{\sigma}_i)$ *outputs* sk_b*.*

Our protocol uses a more general *partial* witness extractor. In contrast with [23,24], our blind signature is not built on an HVZK Sigma protocol: in particular, a secret key cannot be extracted from $\sigma_i, \hat{\sigma}_i$. Nevertheless, we can extract significant *partial information* about a secret key—more precisely, $f(\mathsf{sk})$ for some fixed function f. This can be enough to break a plausible cryptographic assumption. We define partial witness extractors precisely in Definition 2.12.

Definition 2.12. *Fix* I, rand *and let* $\mathbf{h}, \hat{\mathbf{h}} \in F_i(I, \mathsf{rand})$ *for some* $i \in [\ell+1]$*. Denote by* $\sigma_i, \hat{\sigma}_i$ *the signatures that correspond to* $h_i, \hat{h}_i$*, respectively. The deterministic algorithms* $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ *are* partial witness extractors *for the function* f *if* $(\mathsf{Ext}_0(\sigma_i, \hat{\sigma}_i), \mathsf{Ext}_1(\sigma_i, \hat{\sigma}_i)) \in \{(f(\mathsf{sk}_0), \perp), (\perp, f(\mathsf{sk}_1)), (f(\mathsf{sk}_0), f(\mathsf{sk}_1))\}$*. For* $b \in \{0,1\}$*, we say that the* b*-side partial witness information can be extracted from* $(I, \mathsf{rand}, \mathbf{h})$ *and* $(I, \mathsf{rand}, \hat{\mathbf{h}})$ *at the index* i *if* $\mathsf{Ext}_b(\sigma_i, \hat{\sigma}_i)$ *outputs* sk_b*.*

During extraction, we will use the sides of a triangle instead of the base. This follows from the observation that if a b-side (partial) witness can be extracted from a base of the triangle, then it can also be extracted from at least one of the sides. A reduction with a b-side (partial) witness hits one corner of the base of the triangle in the first run, and after rewinding it hits the top of the triangle, yielding a side with a $(1-b)$-(partial) witness with probability roughly $1/2$.

The following two lemmas from [21] will be required to invoke the main theorem proving one-more unforgeability. The first shows that the blind signature is perfectly witness-indistinguishable. The second shows the above-described observation that if a (partial) witness can be extracted from a base of a triangle then it can also be extracted from at least one of the triangle's sides.

Lemma 2.13 ([21, Lemma 2]). *Fix* $\mathsf{rand}, \mathbf{h}$. *For all tuples* $(I, \mathsf{rand}, \mathbf{h}) \in \mathsf{Succ}$, $\Phi_{\mathsf{rand},\mathbf{h}}$ *is a self-inverse bijection and* $\mathsf{trans}(I, \mathsf{rand}, \mathbf{h}) = \mathsf{trans}(\Phi_{\mathsf{rand},\mathbf{h}}(I), \mathsf{rand}, \mathbf{h})$.

Lemma 2.14 ([21, Corollary 3]). *Fix* I, rand *and let* $(\mathbf{h}, \hat{\mathbf{h}}, \hat{\hat{\mathbf{h}}}) \in \Delta_i(I, \mathsf{rand})$, *for some* $i \in [\ell+1]$. *If the* 0*-side (*1*-side) witness can be extracted from the base* $(I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\mathbf{h}})$ *of the triangle at index* i, *then one can also extract the* 0*-side (*1*-side) witness from at least one of the sides* $(I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\hat{\mathbf{h}}})$ *or* $(I, \mathsf{rand}, \hat{\mathbf{h}}), (I, \mathsf{rand}, \hat{\hat{\mathbf{h}}})$ *at index* i.

2.4 Matrix Codes

Let $\mathcal{M}_{m,n}(\mathbb{F})$ be the set of $m \times n$ matrices over a field $\mathbb{F}_q$. A $[m \times n, k]$ *matrix code* is a k-dimensional subspace $\mathcal{C}$ of $\mathcal{M}_{m,n}(\mathbb{F}_q)$, measured with the rank metric. In this metric the *distance* between two codewords, i.e. matrices $A, B \in \mathcal{M}_{m,n}(\mathbb{F}_q)$, is defined as $\mathsf{d}(A, B) = \mathsf{rank}(A - B)$. A matrix code is generated by any set of k linearly independent codewords, that is, by a basis $\langle C_1, \ldots, C_k \rangle$ for the subspace.

Matrix codes can be represented via a *generator matrix*, as follows. Let vec denote the vectorization operator, which "flattens" matrices row-wise:

$$\mathsf{vec}\left(\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}\right) = [x_{11} \ \cdots \ x_{1n} \ x_{21} \ \cdots \ x_{2n} \ \cdots \ x_{m1} \ \cdots \ x_{mn}]$$

If $\mathcal{C}$ is an $[m \times n, k]$ matrix code, then $\mathsf{vec}(\mathcal{C}) := \{\mathsf{vec}(X) : X \in \mathcal{C}\}$ is an $[mn, k]$ linear code[†], which can be represented by an $k \times mn$ generator matrix G—that is, $\mathsf{vec}(\mathcal{C}) = \{\mathbf{y}G : \mathbf{y} \in \mathbb{F}^k\}$. We call G a generator matrix of $\mathcal{C}$.

Given a generator matrix G for a code $\mathsf{vec}(\mathcal{C})$, we can apply any row operation to G without affecting the code; that is, for any $S \in \mathrm{GL}_k$, the matrix MG is also a generator matrix for $\mathsf{vec}(\mathcal{C})$. To represent codes uniquely, we will bring our generator matrices into *systematic form* (or *reduced row echelon form*). For

[†] The usual metric on linear codes is the Hamming metric $\mathsf{d}(\mathbf{x}, \mathbf{y}) = |\{1 \le i \le mn : x_i \ne y_i\}|$. This bears no relation to the rank metric used for the unvectorized code.

a matrix G, we denote its systematic form by $\mathsf{sf}(G)$. From now on, we assume that each generator matrix is of the form $G = [G_1 \mid G_2]$ where $G_1 \in \mathrm{GL}_k$ so that $\mathsf{sf}(G) = [I \mid G_0^{-1}G_1]$. This will dramatically reduce the communication requirements of our protocols.
The group $\mathrm{GL}_m(\mathbb{F}) \times \mathrm{GL}_n(\mathbb{F})$ acts on the set $\mathcal{X}$ of codes by conjugation:

$$(A, B) * \mathcal{C} = A\mathcal{C}B^T = \{AXB^T \,:\, X \in \mathcal{C}\}$$

(note that it is necessary to take the transpose of B here, to ensure that the resulting operation is a left group action). Indeed, this action is an *isometry*, i.e. a map which preserves the distances of the code. This leads to the following notion of isomorphism between matrix codes.

Definition 2.15. *Let $\mathcal{C}$ and $\mathcal{C}'$ be two $[m \times n, k]$ matrix codes over $\mathbb{F}_q$. We say that $\mathcal{C}$ and $\mathcal{C}'$ are* equivalent *if there exist two matrices $A \in \mathrm{GL}_m(\mathbb{F}_q)$ and $B \in \mathrm{GL}_n(\mathbb{F}_q)$ such that $\mathcal{C}' = (A, B)*\mathcal{C}$, i.e. for all codewords $C \in \mathcal{C}$, $ACB^T \in \mathcal{C}'$.*

The equivalence between two matrix codes can be compactly expressed using the Kronecker product $A^T \otimes B^T$. If $\mathcal{C}$ and $\mathcal{C}'$ are equivalent with $\mathcal{C}' = A\mathcal{C}B^T$, and respective generator matrices G and G', then since $\mathsf{vec}(ACB^T) = \mathsf{vec}(C)(A^T \otimes B^T)$, there must exist $S \in \mathrm{GL}_k(\mathbb{F}_q)$ such that $G' = SG(A^T \otimes B^T)$.

The notion of code isomorphism naturally leads to the notion of the automorphism group of a code:

Definition 2.16 (Automorphism Group). *Let $\mathcal{C} \in \mathcal{X}$. The* automorphism group *of $\mathcal{C}$ is* $\mathrm{Aut}(\mathcal{C}) = \{(A, B) \in \mathrm{GL}_m(\mathbb{F}) \times \mathrm{GL}_n(\mathbb{F}) \,:\, (A, B) * \mathcal{C} = \mathcal{C}\}$.

Since any $[m \times n, k]$ code $\mathcal{C}$ over $\mathbb{F}$ is a linear subspace of $\mathcal{M}_{m,n}(\mathbb{F})$, we see that $\gamma C \in \mathcal{C}$ whenever $C \in \mathcal{C}$. Thus, if $A = \alpha I_m$ and $B = \beta I_n$ with $\alpha, \beta \in \mathbb{F}$, then $ACB = \alpha\beta C \in \mathcal{C}$ whenever $C \in \mathcal{C}$. Therefore, for any code $\mathcal{C}$ we have

$$\mathrm{Aut}(\mathcal{C}) \supseteq \{\alpha I_m \,:\, \alpha \in \mathbb{F}\} \times \{\beta I_n \,:\, \beta \in \mathbb{F}\} = Z\left(\mathrm{GL}_m(\mathbb{F}) \times \mathrm{GL}_n(\mathbb{F})\right) \quad (1)$$

Codes which satisfy Equation (1) with equality are called *rigid* [27]:

Definition 2.17 (Rigid Matrix Code). *A matrix code $\mathcal{C}$ over a field $\mathbb{F}$ is* rigid *if* $\mathrm{Aut}(\mathcal{C}) = \{\alpha I_m \,:\, \alpha \in \mathbb{F}\} \times \{\beta I_n \,:\, \beta \in \mathbb{F}\}$.

2.5 Code Equivalence and Related Computational Problems

Definition 2.18 (Computational Matrix Code Equivalence Problem (MCE)). *Given two equivalent matrix codes $\mathcal{C}, \mathcal{C}' \in \mathcal{X}$, find $(A, B) \in \mathrm{GL}_m(\mathbb{F}) \times \mathrm{GL}_n(\mathbb{F})$ such that $\mathcal{C}' = (A, B) * \mathcal{C}$.*

Definition 2.19 (Computational Inverse Matrix Code Equivalence Problem (IMCE)). *Given three equivalent matrix codes $\mathcal{C}, \mathcal{C}', \mathcal{C}'' \in \mathcal{X}$ such that $\mathcal{C}' = (A, B) * \mathcal{C}$ and $\mathcal{C}'' = (A^{-1}, B^{-1}) * \mathcal{C}$, find $(\hat{A}, \hat{B}) \in \mathrm{GL}_m(\mathbb{F}) \times \mathrm{GL}_n(\mathbb{F})$ such that $\mathcal{C}' = (\hat{A}, \hat{B}) * \mathcal{C}$ and $\mathcal{C}'' = (\hat{A}^{-1}, \hat{B}^{-1}) * \mathcal{C}$.*

Let $\mathbb{S}_k(\mathbb{F}) = \{A \in \mathrm{GL}_k(\mathbb{F}) : \exists \alpha \in \mathbb{F} \text{ s.t. } A^T = \alpha A\}$ denote the set of invertible matrices which are *symmetric up to a scalar*. In our blind signature scheme, we require the secret keys to satisfy $(A, B) \in \mathbb{S}_m(\mathbb{F}) \times \mathbb{S}_n(\mathbb{F})$. For an invertible matrix A over a field, the equation $A^T = \alpha A$ implies $\alpha = \pm 1$. Thus, we more precisely say that A and B must be *symmetric or antisymmetic*.

Definition 2.20 (Modified Inverse Matrix Code Equivalence Problem (MIMCE)). *Let $\mathcal{C} \in \mathcal{X}$, and let $(A, B) \in \mathbb{S}_m(\mathbb{F}) \times \mathbb{S}_n(\mathbb{F})$. Given $\mathcal{C}' = (A, B) * \mathcal{C}$ and $\mathcal{C}'' = (A^{-1}, B^{-1}) * \mathcal{C}$, find $(D, F) \in \mathbb{S}_m(\mathbb{F}) \times \mathbb{S}_n(\mathbb{F})$ such that $(D, F) * \mathcal{C}'' = \mathcal{C}'$.*

The MIMCE Problem of Definition 2.20 is essentially a computational version of IMCE [14], except that the secret key matrices are symmetric or antisymmetric. The best currently-known way to solve this problem is an extended algebraic attack which recovers A, B, us a technique similar to the one used for MCE. We expect that the difficulty of this problem is closely related to that of MCE, after accounting for an inevitable loss that will arise from the additional equations available, which will effect a consequent increase in parameters. An exact quantification of this loss factor is beyond the scope of this paper.

Remark 2.21. Given the pair (A^2, B^2), it is likely possible to recover (A, B) in polynomial time from the original instance $\mathcal{C}, \mathcal{C}' = (A, B) * \mathcal{C}$. Thus, if $\mathcal{C}$ is a rigid code, the MIMCE problem of Definition 2.20 should be equivalent to Inverse Matrix Code Equivalence problem. We do not prove this result here.

2.6 Notation

The multiplicative group $\{-1, 1\}$ acts on $\mathrm{GL}_k(\mathbb{F})$ by the inversion-transposition action, which we denote by an exponent in parentheses; in particular, for $A \in \mathrm{GL}_k(\mathbb{F})$, we will denote $A^{(1)} = A$ and $A^{(-1)} = A^{-T}$. This notation will allow us to introduce much more compact notation for our signature scheme in Section 4.

We will also make use of vectorized operations to further condense our notation. In particular, for $A \in \mathrm{GL}_k(\mathbb{F}), \mathbf{A} = (A_i)_{i=1}^{\kappa}$ and $\mathbf{B} = (B_i)_{i=1}^{\kappa} \in \mathrm{GL}_k(\mathbb{F})^{\kappa}, c \in \{-1, 1\}$, and $\mathbf{c} = (c_i)_{i=1}^{\kappa} \in \{-1, 1\}^{\kappa}$ we define

$$A^{(\mathbf{c})} = (A^{(c_i)})_{i=1}^{\kappa}, \quad \mathbf{A}^{(c)} = (A_i^{(c)})_{i=1}^{\kappa}, \quad \mathbf{A} \odot \mathbf{B} = (A_i B_i)_{i=1}^{\kappa}, \quad \mathbf{A}^{(\mathbf{c})} = (A_i^{(c_i)})_{i=1}^{\kappa}$$

3 Public Key Validation

Our blind signature protocol is constructed by applying a variant of the Fiat-Shamir heuristic, as in [23]. Unlike in [23], however, our protocol will require private and public keys with special structure, and there is no known mechanism to determine whether a public key of the form we require is well-formed. In this section we introduce Sigma protocols which can be compiled (by the Fiat-Shamir heuristic) into non-interactive zero-knowledge proofs of public-key well-formedness in our protocol. For a fixed code $X^{(0)} \in \mathcal{X}$, define the relations

$$\mathcal{R}_{X^{(0)}}^{(\pm 1)} = \left\{ \begin{pmatrix} \mathsf{X} = (X^{(1)}, X^{(-1)}), \\ \mathsf{W} = (A, B) \end{pmatrix} : \begin{matrix} X^{(1)} = (A, B) * X^{(0)}, \\ X^{(-1)} = (A^{-T}, B^{-T}) * X^{(0)} \end{matrix} \right\}$$

$$\mathcal{R}^{\mathbb{S}}_{X^{(0)}} = \left\{ \begin{pmatrix} \mathsf{X} = X^{(1)}, \\ \mathsf{W} = (A, B) \end{pmatrix} : \begin{matrix} X^{(1)} = (A, B) * X^{(0)} \\ A \in \mathbb{S}_m(\mathbb{F}), B \in \mathbb{S}_n(\mathbb{F}) \end{matrix} \right\}$$

with corresponding languages $\mathcal{L}^{(\pm 1)}_{X^{(0)}}$ and $\mathcal{L}^{\mathbb{S}}_{X^{(0)}}$, respectively. A valid public key for our blind signature will have the form $((A, B) * X^{(0)}, (A^{-1}, B^{-1}) * X^{(0)})$, with A and B symmetric or antisymmetric. In contrast with CSI-Otter, where it is easy to verify the validity of a public key (any supersingular curve with the correct number of points is a valid key), there is no obvious way to verify that a public key in our protocol is valid. Thus, we will include with the public key additional NIZKs which ensure that a public key is well-formed. In particular, we include two separate proofs: one for each of the two relations above. To achieve this, in this section, we introduce novel Sigma protocols for these relations.

For space reasons, we simply present the two Sigma protocols here in Figures 3 and 4, and defer their proofs of correctness and security to Appendix A.

$\mathsf{P}:$	$\mathsf{W} = (A \in \mathrm{GL}_m(\mathbb{F}_p), B \in \mathrm{GL}_n(\mathbb{F}_p))$ $\mathsf{X} = \begin{pmatrix} X^{(1)} = (A, B) * X^{(0)}, \\ X^{(-1)} = (A^{-T}, B^{-T}) * X^{(0)} \end{pmatrix}$		$\mathsf{V}: \mathsf{X} = (X^{(1)}, X^{(-1)})$
	$(\tilde{A}, \tilde{B}) \xleftarrow{\$} \mathrm{GL}_m(\mathbb{F}_q) \times \mathrm{GL}_n(\mathbb{F}_q)$		
	$\mathsf{comm}^{(1)} \leftarrow (\tilde{A}, \tilde{B}) * X^{(0)}$	$\xrightarrow{\mathsf{comm}}$	
	$\mathsf{comm}^{(-1)} \leftarrow (\tilde{A}^{-T}, \tilde{B}^{-T}) * X^{(0)}$		
		$\xleftarrow{\mathsf{chal}}$	$\mathsf{chal} \xleftarrow{\$} \{0, 1\}.$
	$\mathsf{resp} = (\tilde{A}A^{-\mathsf{chal}}, \tilde{B}B^{-\mathsf{chal}})$	$\xrightarrow{\mathsf{resp}}$	Accept if
			$\begin{cases} \mathsf{resp} * X^{(\mathsf{chal})} = \mathsf{comm}^{(1)} \\ \mathsf{resp}^{-T} * X^{(-\mathsf{chal})} = \mathsf{comm}^{(-1)} \end{cases}$

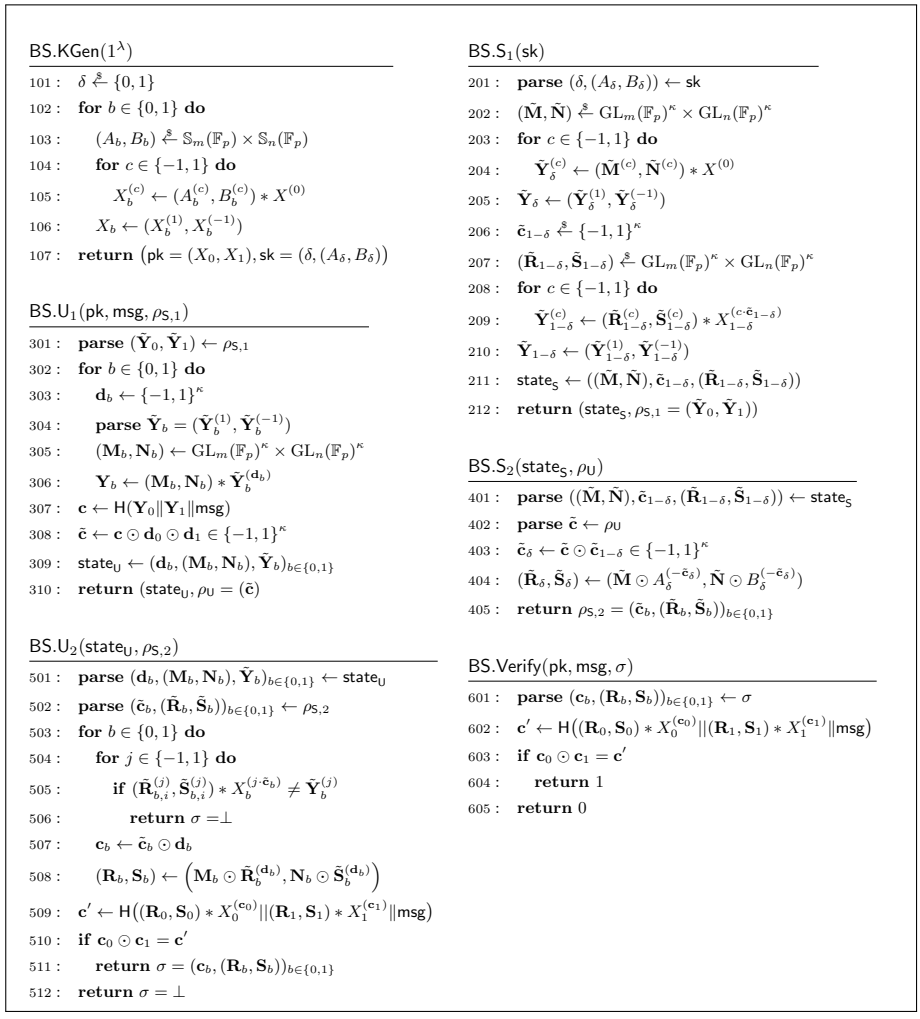
Fig. 3. Our Sigma protocol for the relation $\mathcal{R}^{(\pm 1)}_{X^{(0)}}$.

$\mathsf{P}:$	$\mathsf{W} = (A \in \mathbb{S}_m(\mathbb{F}_p), B \in \mathbb{S}_n(\mathbb{F}_p))$ $\mathsf{X} = X^{(1)} = (A, B) * X^{(0)}$		$\mathsf{V}: \mathsf{X} = X^{(1)}$
	$(\tilde{A}, \tilde{B}) \xleftarrow{\$} \mathrm{GL}_m(\mathbb{F}_q) \times \mathrm{GL}_n(\mathbb{F}_q)$		
	$\mathsf{comm}^{(1)} \leftarrow (\tilde{A}, \tilde{B}) * X^{(0)}$	$\xrightarrow{\mathsf{comm}}$	
	$\mathsf{comm}^{(-1)} \leftarrow (\tilde{A}^{-T}, \tilde{B}^{-T}) * X^{(1)}$		
		$\xleftarrow{\mathsf{chal}}$	$\mathsf{chal} \xleftarrow{\$} \{0, 1\}.$
	$\mathsf{resp} = (\tilde{A}A^{-\mathsf{chal}}, \tilde{B}B^{-\mathsf{chal}})$	$\xrightarrow{\mathsf{resp}}$	Accept if
			$\begin{cases} \mathsf{resp} * X^{(\mathsf{chal})} = \mathsf{comm}^{(1)} \\ \mathsf{resp}^{-T} * X^{(1-\mathsf{chal})} = \mathsf{comm}^{(-1)} \end{cases}$

Fig. 4. Our Sigma protocol for the relation $\mathcal{R}^{\mathbb{S}}_{X^0}$.

4 Our Blind Signature Scheme

Let $p, n, m, X^{(0)}$ be the public parameters and let $\mathsf{H} : \{0,1\}^* \to \{-1,1\}^\kappa$ be a hash function that is modeled as a random oracle in the security proof. Our blind signature scheme is depicted in Figure 5.



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

$\delta \xleftarrow{\$} \{0,1\}$
for $b \in \{0,1\}$ **do**
  $(A_b, B_b) \xleftarrow{\$} \mathbb{S}_m(\mathbb{F}_p) \times \mathbb{S}_n(\mathbb{F}_p)$
  for $c \in \{-1,1\}$ **do**
    $X_b^{(c)} \leftarrow (A_b^{(c)}, B_b^{(c)}) * X^{(0)}$
  $X_b \leftarrow (X_b^{(1)}, X_b^{(-1)})$
return $(\mathsf{pk} = (X_0, X_1), \mathsf{sk} = (\delta, (A_\delta, B_\delta)))$

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

parse $(\tilde{\mathbf{Y}}_0, \tilde{\mathbf{Y}}_1) \leftarrow \rho_{\mathsf{S},1}$
for $b \in \{0,1\}$ **do**
  $\mathbf{d}_b \leftarrow \{-1,1\}^\kappa$
  parse $\tilde{\mathbf{Y}}_b = (\tilde{\mathbf{Y}}_b^{(1)}, \tilde{\mathbf{Y}}_b^{(-1)})$
  $(\mathbf{M}_b, \mathbf{N}_b) \leftarrow \mathrm{GL}_m(\mathbb{F}_p)^\kappa \times \mathrm{GL}_n(\mathbb{F}_p)^\kappa$
  $\mathbf{Y}_b \leftarrow (\mathbf{M}_b, \mathbf{N}_b) * \tilde{\mathbf{Y}}_b^{(\mathbf{d}_b)}$
$\mathbf{c} \leftarrow \mathsf{H}(\mathbf{Y}_0 \| \mathbf{Y}_1 \| \mathsf{msg})$
$\tilde{\mathbf{c}} \leftarrow \mathbf{c} \odot \mathbf{d}_0 \odot \mathbf{d}_1 \in \{-1,1\}^\kappa$
$\mathsf{state}_\mathsf{U} \leftarrow (\mathbf{d}_b, (\mathbf{M}_b, \mathbf{N}_b), \tilde{\mathbf{Y}}_b)_{b\in\{0,1\}}$
return $(\mathsf{state}_\mathsf{U}, \rho_\mathsf{U} = (\tilde{\mathbf{c}}))$

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

parse $(\mathbf{d}_b, (\mathbf{M}_b, \mathbf{N}_b), \tilde{\mathbf{Y}}_b)_{b\in\{0,1\}} \leftarrow \mathsf{state}_\mathsf{U}$
parse $(\tilde{\mathbf{c}}_b, (\tilde{\mathbf{R}}_b, \tilde{\mathbf{S}}_b))_{b\in\{0,1\}} \leftarrow \rho_{\mathsf{S},2}$
for $b \in \{0,1\}$ **do**
  for $j \in \{-1,1\}$ **do**
    if $(\tilde{\mathbf{R}}_{b,i}^{(j)}, \tilde{\mathbf{S}}_{b,i}^{(j)}) * X_b^{(j\cdot\tilde{\mathbf{c}}_b)} \neq \tilde{\mathbf{Y}}_b^{(j)}$
      return $\sigma = \perp$
  $\mathbf{c}_b \leftarrow \tilde{\mathbf{c}}_b \odot \mathbf{d}_b$
  $(\mathbf{R}_b, \mathbf{S}_b) \leftarrow \left(\mathbf{M}_b \odot \tilde{\mathbf{R}}_b^{(\mathbf{d}_b)}, \mathbf{N}_b \odot \tilde{\mathbf{S}}_b^{(\mathbf{d}_b)}\right)$
$\mathbf{c}' \leftarrow \mathsf{H}\big((\mathbf{R}_0, \mathbf{S}_0) * X_0^{(\mathbf{c}_0)} || (\mathbf{R}_1, \mathbf{S}_1) * X_1^{(\mathbf{c}_1)} \| \mathsf{msg}\big)$
if $\mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}'$
  return $\sigma = (\mathbf{c}_b, (\mathbf{R}_b, \mathbf{S}_b))_{b\in\{0,1\}}$
return $\sigma = \perp$

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

parse $(\delta, (A_\delta, B_\delta)) \leftarrow \mathsf{sk}$
$(\tilde{\mathbf{M}}, \tilde{\mathbf{N}}) \xleftarrow{\$} \mathrm{GL}_m(\mathbb{F}_p)^\kappa \times \mathrm{GL}_n(\mathbb{F}_p)^\kappa$
for $c \in \{-1,1\}$ **do**
  $\tilde{\mathbf{Y}}_\delta^{(c)} \leftarrow (\tilde{\mathbf{M}}^{(c)}, \tilde{\mathbf{N}}^{(c)}) * X^{(0)}$
$\tilde{\mathbf{Y}}_\delta \leftarrow (\tilde{\mathbf{Y}}_\delta^{(1)}, \tilde{\mathbf{Y}}_\delta^{(-1)})$
$\tilde{\mathbf{c}}_{1-\delta} \xleftarrow{\$} \{-1,1\}^\kappa$
$(\tilde{\mathbf{R}}_{1-\delta}, \tilde{\mathbf{S}}_{1-\delta}) \xleftarrow{\$} \mathrm{GL}_m(\mathbb{F}_p)^\kappa \times \mathrm{GL}_n(\mathbb{F}_p)^\kappa$
for $c \in \{-1,1\}$ **do**
  $\tilde{\mathbf{Y}}_{1-\delta}^{(c)} \leftarrow (\tilde{\mathbf{R}}_{1-\delta}^{(c)}, \tilde{\mathbf{S}}_{1-\delta}^{(c)}) * X_{1-\delta}^{(c\cdot\tilde{\mathbf{c}}_{1-\delta})}$
$\tilde{\mathbf{Y}}_{1-\delta} \leftarrow (\tilde{\mathbf{Y}}_{1-\delta}^{(1)}, \tilde{\mathbf{Y}}_{1-\delta}^{(-1)})$
$\mathsf{state}_\mathsf{S} \leftarrow ((\tilde{\mathbf{M}}, \tilde{\mathbf{N}}), \tilde{\mathbf{c}}_{1-\delta}, (\tilde{\mathbf{R}}_{1-\delta}, \tilde{\mathbf{S}}_{1-\delta}))$
return $(\mathsf{state}_\mathsf{S}, \rho_{\mathsf{S},1} = (\tilde{\mathbf{Y}}_0, \tilde{\mathbf{Y}}_1))$

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

parse $((\tilde{\mathbf{M}}, \tilde{\mathbf{N}}), \tilde{\mathbf{c}}_{1-\delta}, (\tilde{\mathbf{R}}_{1-\delta}, \tilde{\mathbf{S}}_{1-\delta})) \leftarrow \mathsf{state}_\mathsf{S}$
parse $\tilde{\mathbf{c}} \leftarrow \rho_\mathsf{U}$
$\tilde{\mathbf{c}}_\delta \leftarrow \tilde{\mathbf{c}} \odot \tilde{\mathbf{c}}_{1-\delta} \in \{-1,1\}^\kappa$
$(\tilde{\mathbf{R}}_\delta, \tilde{\mathbf{S}}_\delta) \leftarrow (\tilde{\mathbf{M}} \odot A_\delta^{(-\tilde{\mathbf{c}}_\delta)}, \tilde{\mathbf{N}} \odot B_\delta^{(-\tilde{\mathbf{c}}_\delta)})$
return $\rho_{\mathsf{S},2} = (\tilde{\mathbf{c}}_b, (\tilde{\mathbf{R}}_b, \tilde{\mathbf{S}}_b))_{b\in\{0,1\}}$

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

parse $(\mathbf{c}_b, (\mathbf{R}_b, \mathbf{S}_b))_{b\in\{0,1\}} \leftarrow \sigma$
$\mathbf{c}' \leftarrow \mathsf{H}\big((\mathbf{R}_0, \mathbf{S}_0) * X_0^{(\mathbf{c}_0)} || (\mathbf{R}_1, \mathbf{S}_1) * X_1^{(\mathbf{c}_1)} \| \mathsf{msg}\big)$
if $\mathbf{c}_0 \odot \mathbf{c}_1 = \mathbf{c}'$
  return 1
return 0

Fig. 5. Our blind signature scheme. We assume that the algorithms return $\perp$ and terminate if **parse** is not in the correct format.

We have the following correctness and security results.

Theorem 4.1. *The scheme of Figure 5 is correct.*

Proof. We must show that if the signer and user act honestly, BS.Verify will always return 1. First, we claim that $\tilde{\mathbf{Y}}_b^{(c)} = (\tilde{\mathbf{R}}_b^{(c)}, \tilde{\mathbf{S}}_b^{(c)}) * X_b^{(c\cdot\tilde{\mathbf{c}}_b)}$ for $b \in \{0,1\}$ and $c \in \{-1,1\}$. The $b = 1-\delta$ case follows from Figure 5 lines 208–210. For the $b = \delta$ case, by Figure 5 lines 105, 202–203, and 404, the fact that A_δ and B_δ are symmetric or antisymmetric, and the fact that $c^2 = 1$ for $c \in \{-1,1\}$, we have

$$\begin{aligned}\tilde{\mathbf{Y}}_\delta^{(c)} &= (\tilde{\mathbf{M}}^{(c)}, \tilde{\mathbf{N}}^{(c)}) * X^{(0)} = (\tilde{\mathbf{M}}^{(c)} \odot A_\delta^{(-c\cdot\tilde{\mathbf{c}}_\delta)}, \tilde{\mathbf{N}}^{(c)} \odot B_\delta^{(-c\cdot\tilde{\mathbf{c}}_\delta)}) * X_\delta^{(c\cdot\tilde{\mathbf{c}}_\delta)} \\ &= \left((\tilde{\mathbf{M}} \odot A_\delta^{(-\tilde{\mathbf{c}}_\delta)})^{(c)}, (\tilde{\mathbf{N}} \odot B_\delta^{(-\tilde{\mathbf{c}}_\delta)})^{(c)}\right) * X_\delta^{(c\cdot\tilde{\mathbf{c}}_\delta)} = (\tilde{\mathbf{R}}_\delta^{(c)}, \tilde{\mathbf{S}}_\delta^{(c)}) * X_\delta^{(c\cdot\tilde{\mathbf{c}}_\delta)}\end{aligned}$$

as required. Then, by lines 507 and 306, we have

$$\mathbf{Y}_b = (\mathbf{M}_b, \mathbf{N}_b) * \tilde{\mathbf{Y}}_b^{(\mathbf{d}_b)} = \left(\mathbf{M}_b \odot \tilde{\mathbf{R}}_b^{(\mathbf{d}_b)}, \mathbf{N}_b \odot \tilde{\mathbf{S}}_b^{(\mathbf{d}_b)}\right) * X_b^{(\mathbf{d}_b \odot \tilde{\mathbf{c}}_b)} = (\mathbf{R}_b, \mathbf{S}_b) * X_b^{(\mathbf{c}_b)}$$

so that that the values computed on lines 307 and 602 are the same:

$$\mathbf{c} = H(\mathbf{Y}_0 || \mathbf{Y}_1 || \mathsf{msg}) = H((\mathbf{R}_0, \mathbf{S}_0) * X_0^{(\mathbf{c}_0)} || (\mathbf{R}_1, \mathbf{S}_1) * X_1^{(\mathbf{c}_1)} || \mathsf{msg}) = \mathbf{c}'.$$

Finally, by considering lines 308 and 507, we see that

$$\mathbf{c}' = \mathbf{c} = \tilde{\mathbf{c}} \odot \mathbf{d}_0 \odot \mathbf{d}_1 = (\tilde{\mathbf{c}}_0 \odot \mathbf{d}_0) \odot (\tilde{\mathbf{c}}_1 \odot \mathbf{d}_1) = \mathbf{c}_0 \odot \mathbf{c}_1$$

so that the check on line 603 returns true, and the signature always verifies. □

Theorem 4.2. *The scheme of Figure 5 is perfectly blind.*

Proof. We will show that for any valid public key $\mathsf{pk} = (X_0, X_1)$, any signer messages $\rho_{\mathsf{S},1} = (\tilde{\mathbf{Y}}_0, \tilde{\mathbf{Y}}_1)$ and $\rho_{\mathsf{S},2} = (\tilde{\mathbf{c}}_b, (\tilde{\mathbf{R}}_b, \tilde{\mathbf{S}}_b))_{b\in\{0,1\}}$, and any valid message/signature pair $\mathsf{msg}, \sigma = (\mathbf{c}_b, (\mathbf{R}_b, \mathbf{S}_b))$, there is exactly one user state $\mathsf{state}_\mathsf{U} = (\mathbf{d}_b, (\mathbf{M}_b, \mathbf{N}_b))_{b\in\{0,1\}}$ which is consistent with pk, $\rho_{\mathsf{S},1}, \rho_{S,2}$, and σ.

Indeed, line 507 uniquely determines $\mathbf{d}_b = \mathbf{c}_b \odot \tilde{\mathbf{c}}_b$ for $b \in \{0,1\}$, and then line 508 uniquely determines $\mathbf{M}_{b,i} = \mathbf{R}_{b,i} \odot (\tilde{\mathbf{R}}_{b,i}^{(\mathbf{d}_{b+i})})^{-1}$ and $\mathbf{N}_{b,i} = \mathbf{S}_{b,i} \odot (\tilde{\mathbf{S}}_{b,i}^{(\mathbf{d}_{b+i})})^{-1}$. It is straightforward to check that this indeed yields a valid signature. Thus, every interaction S has with U is compatible with every output (msg, σ) pair, and each correspondence is equally likely; thus, the protocol is blind.

Theorem 4.3. *Assume that the public key consists of two instance of the NP relation $\mathcal{R}$ generated by the hard instance generator IG and the underlying Sigma protocol has challenge space $\mathcal{C}$. If Lemmas 2.13 and 2.14 hold, then for all $\ell \in \mathbb{N}$, if there exists an adversary $\mathcal{A}$ that issues Q hash queries to the random oracle and breaks the ℓ-one more unforgeability of the blind signature scheme with advantage $\epsilon_\mathcal{A} \geq \frac{C_1}{|\mathcal{C}|} \cdot \binom{Q}{\ell+1}$, then there exists an efficient algorithm $\mathcal{B}$ that breaks the hard instance generator for the MIMCE problem (Definition 2.20) with advantage $\epsilon_\mathcal{B} \geq C_2 \cdot \frac{\epsilon_\mathcal{A}^2}{\binom{Q}{\ell+1}^2 \cdot (\ell+1)^3}$, where C_1 and C_2 are some universal global constants.*

Proof. To prove Theorem 4.3 we first need to define instances, the map $\Phi_{\mathsf{rand},\mathbf{h}}$, the partial witness extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ and prove the lemmas 2.13, 2.14. We assume an adversary $\mathcal{A}$ against the one-more unforgeability game issuing ℓ-queries to the signing oracle. We use the notation $\overrightarrow{\mathbf{Z}}$ to denote a vector $(\mathbf{Z}^{(1)}, \ldots, \mathbf{Z}^{(\ell)})$ which inherits the operations of $\mathbf{Z}^{(k)}$.

Instances: We prepare a b-side instance I_b, $b \in \{0,1\}$ as follows:
$I_0 = (0, (A_0, B_0), X_1, (\vec{\tilde{\mathbf{M}}}, \vec{\tilde{\mathbf{N}}}), \vec{\tilde{\mathbf{c}}}_1, (\vec{\tilde{\mathbf{R}}}_1, \vec{\tilde{\mathbf{S}}}_1))$ consisting of:

- $(0, (A_0, B_0))$ is the secret key sk for $\delta = 0$.
- X_1 is the public key component with unknown secret key.
- $\left((\tilde{\mathbf{M}}^{(c)})^{(k)}, (\tilde{\mathbf{N}}^{(c)})^{(k)}\right)$ is a randomness used in the calculation of commitment $(\tilde{\mathbf{Y}}_0{}^{(c)})^{(k)} = \left((\tilde{\mathbf{M}}^{(c)})^{(k)}, (\tilde{\mathbf{N}}^{(c)})^{(k)}\right) * (X^{(0)})^{(k)}$.
- $\tilde{\mathbf{c}}_1^{(k)}$ is the simulated challenge in the k-th first-sender message for $\delta = 0$.
- $\left((\tilde{\mathbf{R}}_1^{(c)})^{(k)}, (\tilde{\mathbf{S}}_1^{(c)})^{(k)}\right)$ is a randomness used in computation of commitment $(\tilde{\mathbf{Y}}_1{}^{(c)})^{(k)} = \left((\tilde{\mathbf{R}}_1^{(c)})^{(k)}, (\tilde{\mathbf{S}}_1^{(c)})^{(k)}\right) * (X_1^{(c\cdot\tilde{\mathbf{c}}_1)})^{(k)}$.

$I_1 = (1, (A_1, B_1), X_0, (\vec{\tilde{\mathbf{M}}}, \vec{\tilde{\mathbf{N}}}), \vec{\tilde{\mathbf{c}}}_0, (\vec{\tilde{\mathbf{R}}}_0, \vec{\tilde{\mathbf{S}}}_0))$, consisting of:

- $(1, (A_1, B_1))$ is the secret key sk for $\delta = 1$.
- X_0 is the public key component with unknown secret key.
- $\left((\tilde{\mathbf{M}}^{(c)})^{(k)}, (\tilde{\mathbf{N}}^{(c)})^{(k)}\right)$ is a randomness used in computation of commitment $(\tilde{\mathbf{Y}}_1{}^{(c)})^{(k)} = \left((\tilde{\mathbf{M}}^{(c)})^{(k)}, (\tilde{\mathbf{N}}^{(c)})^{(k)}\right) * (X^{(0)})^{(k)}$.
- $\tilde{\mathbf{c}}_0^{(k)}$ is the simulated challenge in the k-th first-sender message for $\delta = 1$.
- $\left((\tilde{\mathbf{R}}_0^{(c)})^{(k)}, (\tilde{\mathbf{S}}_0^{(c)})^{(k)}\right)$ is a randomness used in computation of commitment $(\tilde{\mathbf{Y}}_0{}^{(c)})^{(k)} = \left((\tilde{\mathbf{R}}_0^{(c)})^{(k)}, (\tilde{\mathbf{S}}_0^{(c)})^{(k)}\right) * (X_0^{(c\cdot\tilde{\mathbf{c}}_1)})^{(k)}$.

Next, we define the map $\Phi_{\mathsf{rand},\mathbf{h}}$ that maps a 0-side instance I_0 into a 1-side instance I_1 and vice versa.

Map $\Phi_{\mathsf{rand},\mathbf{h}}$: We define the map $\Phi_{\mathsf{rand},\mathbf{h}}$ which maps a $0-$instance $I_0 = (0, (A_0, B_0), X_1, (\vec{\tilde{\mathbf{M}}}, \vec{\tilde{\mathbf{N}}}), \vec{\tilde{\mathbf{c}}}_1, (\vec{\tilde{\mathbf{R}}}_1, \vec{\tilde{\mathbf{S}}}_1))$ into a 1-side instance $I_1 = (1, (A_1, B_1), X_0, (\vec{\tilde{\mathbf{M}}}, \vec{\tilde{\mathbf{N}}}), \vec{\tilde{\mathbf{c}}}_0, (\vec{\tilde{\mathbf{R}}}_0, \vec{\tilde{\mathbf{S}}}_0))$ and vice versa.
In the map of a 0-side instance into a 1-side instance the following holds: $X_1 = (X_1^{(1)}, X_1^{(-1)})$ where $X_1^{(c)} = (A_1^{(c)}, B_1^{(c)}) * X^{(0)}$ and $\vec{\tilde{\mathbf{c}}} = \vec{\mathbf{e}}(I_0, \mathsf{rand}, \mathbf{h})$.
In the map of a 1-side instance into a 0-side instance the following holds: $X_0 = (X_0^{(1)}, X_0^{(-1)})$ where$X_0^{(c)} = (A_0^{(c)}, B_0^{(c)}) * X^{(0)}$ and and $\vec{\tilde{\mathbf{c}}} = \vec{\mathbf{e}}(I_1, \mathsf{rand}, \mathbf{h})$.

Partial Witness Extractors: We fix I, rand and let $(\mathbf{h}, \hat{\mathbf{h}}) \in F_i(I, \mathsf{rand})$ for some $i \in [\ell + 1]$. Let $\sigma = (\mathbf{c}_b, (\mathbf{R}_b, \mathbf{S}_b))_{b\in\{0,1\}}$ and $\hat{\sigma} = (\hat{\mathbf{c}}_b, (\hat{\mathbf{R}}_b, \hat{\mathbf{S}}_b))_{b\in\{0,1\}}$ denote the signatures corresponding to $\mathbf{c}^{(i)}$ and $\hat{\mathbf{c}}^{(i)}$, respectively. (Note: $\mathbf{c}^{(i)}$ and $\hat{\mathbf{c}}^{(i)}$ denotes the i-th entry of $\mathbf{h}$ and $\hat{\mathbf{h}}$ respectively.) The partial witness extractor is defined as in Figure 6.

Lemma 4.4. *If $X^{(0)}$ is a rigid code, then the functions $(\mathsf{Ext}_0, \mathsf{Ext}_1)$ are partial witness extractors for $f(A, B) = (A^2, B^2)$, as defined in Definition 2.12.*

$\mathsf{Ext}_0(\sigma, \hat{\sigma})$	$\mathsf{Ext}_1(\sigma, \hat{\sigma})$
101 : **if** $\exists t \in [\kappa]$ s.t. $c_{0,t} \neq \hat{c}_{0,t}$	201 : **if** $\exists t \in [\kappa]$ s.t. $c_{1,t} \neq \hat{c}_{1,t}$
102 : $D_0 \leftarrow (R_{0,t}^{-1} \hat{R}_{0,t})^{\frac{c_{0,t}-\hat{c}_{0,t}}{2}}$	202 : $D_1 \leftarrow (R_{1,t}^{-1} \hat{R}_{1,t})^{\frac{c_{1,t}-\hat{c}_{1,t}}{2}}$
103 : $F_0 \leftarrow (S_{0,t}^{-1} \hat{S}_{0,t})^{\frac{c_{0,t}-\hat{c}_{0,t}}{2}}$	203 : $F_1 \leftarrow (S_{1,t}^{-1} \hat{S}_{1,t})^{\frac{c_{1,t}-\hat{c}_{1,t}}{2}}$
104 : **return** (D_0, F_0)	204 : **return** (D_1, F_1)
105 : **return** $\perp$	205 : **return** $\perp$

Fig. 6. Partial witness extractors for our blind signature. We assume the signatures take the form $\sigma = (\mathbf{c}_b, (\mathbf{R}_b, \mathbf{S}_b))_{b\in\{0,1\}}$ and $\hat{\sigma} = (\hat{\mathbf{c}}_b, (\hat{\mathbf{R}}_b, \hat{\mathbf{S}}_b))_{b\in\{0,1\}}$ with $\mathbf{c}_b, \hat{\mathbf{c}}_b \in \{1, -1\}^\kappa$, $\mathbf{R}_b, \hat{\mathbf{R}}_b \in \mathrm{GL}_m(\mathbb{F}_p)^\kappa$ and $\mathbf{S}_b, \hat{\mathbf{S}}_b \in \mathrm{GL}_n(\mathbb{F}_p)^\kappa$ for $b \in \{0, 1\}$. Note that the exponents that appear on Lines 102, 103, 202, and 203 are always 1 or -1, so no fractional powers of matrices are being computed.

Proof. According to the definition of successful forking 2.7 two successful input tuples $(I, \mathsf{rand}, \mathbf{h})$ and $(I, \mathsf{rand}, \hat{\mathbf{h}}) \in \mathsf{Succ}$ fork from each other at index $i \in [\ell+1]$ and the set of corresponding hash vectors $(\mathbf{h}, \hat{\mathbf{h}})$ is denoted by $F_i(I, \mathsf{rand})$. Therefore, for the two successful input tuples we have $\mathbf{c}^{(i)} \neq \hat{\mathbf{c}}^{(i)}$ and the corresponding two signatures $\sigma, \hat{\sigma}$ are valid. We have $\mathsf{H}\big((\mathbf{R}_0, \mathbf{S}_0) * X_0^{(\mathbf{c}_0)} || (\mathbf{R}_1, \mathbf{S}_1) * X_1^{(\mathbf{c}_1)} \| \mathsf{msg}\big)$ and $\mathsf{H}\big((\hat{\mathbf{R}}_0, \hat{\mathbf{S}}_0) * X_0^{(\hat{\mathbf{c}}_0)} || (\hat{\mathbf{R}}_1, \hat{\mathbf{S}}_1) * X_1^{(\hat{\mathbf{c}}_1)} \| \mathsf{msg}\big)$. By definition, $\mathbf{h}$ and $\hat{\mathbf{h}}$ agree up to the i-th entry and the randomnesses of the challenger and the adversary are fixed, so the input to the hash function must agree as well. Therefore, we have

$$(\mathbf{R}_b, \mathbf{S}_b) * X_b^{(\mathbf{c}_b)} = (\hat{\mathbf{R}}_b, \hat{\mathbf{S}}_b) * X_b^{(\hat{\mathbf{c}}_b)} \tag{2}$$

for $b \in \{0, 1\}$. Since $\mathbf{c}^{(i)} \neq \hat{\mathbf{c}}^{(i)}$, we must have $\mathbf{c}_b \neq \hat{\mathbf{c}}_b$ for some $b \in \{0, 1\}$. Let $t \in [\kappa]$ be an index for which $c_{b,t} \neq \hat{c}_{b,t}$, and assume without loss of generality that $c_{b,t} = 1$ and $\hat{c}_{b,t} = -1$. By considering the t^{th} component of Equation (2) and rearranging, we find that $X^{(1)} = (R_{b,t}^{-1} \hat{R}_{b,t}, S_{b,t}^{-1} \hat{S}_{b,t}) * X^{(-1)}$. Since $X^{(0)}$ is rigid, so are $X^{(1)}$ and $X^{(-1)}$. Thus, up to scalar factors we have

$$\mathsf{Ext}_b(\sigma, \hat{\sigma}) = (R_{b,t}^{-1} \hat{R}_{b,t}, S_{b,t}^{-1} \hat{S}_{b,t}) = (A^2, B^2).$$

A totally analogous argument applies when $c_{b,t} = -1$ and $\hat{c}_{b,t} = 1$. Indeed, these functions output valid partial witnesses for f. □

To complete the proof of one-more unforgeability of our blind signature scheme, we must show that Lemmas 2.13 and 2.14 hold for the definitions of the map $\Phi_{\mathsf{rand},\mathbf{h}}$ and the extractors $(\mathsf{Ext}_0, \mathsf{Ext}_1)$, respectively.

Lemma 4.5. *Lemma 2.13 holds for our* $\Phi_{\mathsf{rand},\mathbf{h}}$.

Proof. To prove this lemma we need to consider the 0-side and 1-side instances, namely I_0 and I_1, respectively. However since the proof technique for both instances is very similar we will focus on one of them, specifically I_0.

We consider the vector of signing queries on user message ρ_{U} issued by the adversary to the oracle $\mathcal{O}_{\mathsf{sign}_2}$. The vector is denoted by $\overrightarrow{e}(I_0, \mathsf{rand}, \mathbf{h}) = \overrightarrow{\tilde{\mathbf{c}}}$. As shown in Theorems A.4 and A.8 the underlying Sigma protocols are honest-verifier zero-knowledge which implies perfect witness indistinguishability. Then for each index $i \in [\ell]$ and $\overrightarrow{\tilde{\mathbf{c}}}^{(i)}$, there exists a set of randomness described by $\Phi_{\mathsf{rand},\mathbf{h}}(I_0)$, that a signer with the secret key $(1, (A_1, B_1))$ could produce the same view to the adversary $\mathcal{A}$. Therefore the transcript $\mathsf{trans}(I_0, \mathsf{rand}, \mathbf{h}) = \mathsf{trans}(\Phi_{\mathsf{rand},\mathbf{h}}(I_0), \mathsf{rand}, \mathbf{h})$. From the definition of $\Phi_{\mathsf{rand},\mathbf{h}}$ it can be checked that $\Phi_{\mathsf{rand},\mathbf{h}}(\Phi_{\mathsf{rand},\mathbf{h}}(I_0)) = I_0$, which is a bijection. □

Lemma 4.6. *Lemma 2.14 holds for our partial witness extractors* $(\mathsf{Ext}_0, \mathsf{Ext}_1)$.

Proof. Just as in the proof of the previous lemma, we only consider the 0-side case. We prove this lemma by contradiction by assuming that we can extract the 0-side witness from the base $(I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\mathbf{h}})$ but not from either of the sides $(I, \mathsf{rand}, \hat{\mathbf{h}}), (I, \mathsf{rand}, \hat{\hat{\mathbf{h}}})$ or $(I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\hat{\mathbf{h}}})$. By Lemma 4.4 this assumption holds if and only if $\mathbf{c}_0 = \hat{\hat{\mathbf{c}}}_0$ and $\hat{\mathbf{c}}_0 = \hat{\hat{\mathbf{c}}}_0$. This would imply that $\mathbf{c}_0 = \hat{\mathbf{c}}_0$. However, by Lemma 4.4, the 0-side witness cannot be extracted from $(I, \mathsf{rand}, \mathbf{h}), (I, \mathsf{rand}, \hat{\mathbf{h}})$ which contradicts our assumption. □

This completes the proof of Theorem 4.3. □

Theorem 4.7. *Our blind signature scheme in Figure 5 is one more unforgeable. In particular, if there exists an efficient adversary $\mathcal{A}$ that issues Q hash queries to the random oracle and breaks the ℓ-one more unforgeability of the blind signature scheme with advantage $\epsilon_{\mathcal{A}} \geq \frac{C_1}{|\mathcal{C}|} \cdot \binom{Q}{\ell+1}$, then there exists an efficient algorithm $\mathcal{B}$ that solves the MIMCE problem with advantage $\epsilon_{\mathcal{B}} \geq C_2 \cdot \frac{\epsilon_{\mathcal{A}}^2}{\binom{Q}{\ell+1}^2 \cdot (\ell+1)^3}$, where C_1 and C_2 are some universal global constants.*

Proof. The instance generator IG outputs an instance of the MIMCE problem, and Theorem 4.7 follows from the above Lemmas 4.5 and 4.6 and from Theorem 4.3, noting that the partial witnesses obtained from Ext_0 and Ext_1 are solutions to the corresponding instances of MIMCE, as per Definition 2.20. □

5 System Parameters

Because our protocols use a restricted secret keyspace, it is likely necessary to increase the MEDS parameter sizes to ensure that our protocol achieves the same security as existing MEDS-based protocols. Analysis of brute force attacks is straightforward: when the private key matrices A and B are required to be symmetric, there are only $O(q^{\binom{m+1}{2}})$ possible values for A, and $O(q^{\binom{n+1}{2}})$ possible values for B. Since knowledge of A or of B is sufficient to find the other in polynomial time [16, Algorithm 2], a brute force attack requires time $\tilde{O}(q^{\min\{\binom{m+1}{2}, \binom{n+1}{2}\}})$. When the private key matrices are not restricted to being

(anti)symmetric, the brute force attack requires time $\tilde{O}(q^{\min\{n^2,m^2\}})$—to maintain the attack complexity in the (anti)symmetric setting, it suffices to increase n and m by a factor of approximately $\sqrt{2}$. Since the matrix arithmetic used in our protocol runs in time $O((m+n)^2\mathsf{polylog}(p))$, this would lead only to a twofold increase in runtime for each basic algorithm. When the private keys are chosen from $\mathbb{S}^{++}_{\Gamma}(\mathbb{F}_p)\times\mathbb{S}^{++}_{\Delta}(\mathbb{F}_p)$ (where $\mathbb{S}^{++}_{\Gamma}(\mathbb{F})$ and $\mathbb{S}^{++}_{\Delta}(\mathbb{F}_p)$ are the sets of symmetric matrices in $\mathrm{GL}_k(\mathbb{F}_p)$ which are diagonalizable by a matrix $P\in\mathrm{GL}_k(\mathbb{F}_p)$, and whose diagonal form is precisely Γ or Δ, respectively), the result is similar provided that Γ and Δ each have distinct diagonal entries, since the number of orthogonal matrices over $\mathbb{F}_p$ is $\Omega(p^{\frac{(n-1)^2}{2}})$. For other attacks discussed in [14], it is not immediately clear how to use the (anti)symmetry of A and B to achieve a speedup, making the required changes to parameter sizes difficult to quantify. We leave this for future work.

Beyond the MEDS parameters, we must also choose κ, i.e. the number of repetitions of the Σ protocol used in each signing session. In ordinary digital signatures, to achieve λ bits of security, it suffices to take $\kappa=\lambda\log_{|\mathcal{C}|}2$, where $\mathcal{C}$ is the challenge space for a single iteration of the Σ protocol—in our setting, we would take $\kappa=\lambda$. However, as was pointed out in [17], this is not sufficient in the setting of blind signatures; instead, we must take $\kappa=4\lambda$ in order to have λ bits of quantum security in the sequential setting. For NIST category 1 parameters, we would require $\kappa=512$. The setting of concurrent sessions is more difficult to analyze; the results of [22,17] indicate that a transformation such as [25] would be necessary for practical security, but the effect on parameter sizes is again difficult to quantify. We leave this for future work as well.

References

1. M. Abe and T. Okamoto. Provably secure partially blind signatures. In M. Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 271–286, Aug. 2000.
2. S. Agrawal, E. Kirshanova, D. Stehlé, and A. Yadav. Practical, round-optimal lattice-based blind signatures. In H. Yin, A. Stavrou, C. Cremers, and E. Shi, editors, *Proceedings of the 2022 ACM SIGSAC*, pages 39–53. ACM, 2022.
3. N. Alamati, L. De Feo, H. Montgomery, and S. Patranabis. Cryptographic group actions and applications. In S. Moriai and H. Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 411–439, Dec. 2020.
4. N. A. Alkadri, R. E. Bansarkhani, and J. Buchmann. BLAZE: practical lattice-based blind signatures for privacy-preserving applications. In J. Bonneau and N. Heninger, editors, *Financial Cryptography and Data Security - FC 2020*, volume 12059 of *LNCS*, pages 484–502. Springer, 2020.
5. F. Baldimtsi and A. Lysyanskaya. On the security of one-witness blind signature schemes. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 82–99, Dec. 2013.
6. W. Beullens, T. Kleinjung, and F. Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In S. D. Galbraith and S. Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247, Dec. 2019.

7. J.-F. Biasse, G. Micheli, E. Persichetti, and P. Santini. LESS is more: Code-based signatures without syndromes. In A. Nitaj and A. M. Youssef, editors, *AFRICACRYPT 20*, volume 12174 of *LNCS*, pages 45–65, July 2020.
8. O. Blazy, P. Gaborit, and D. T. Mac. A correction to a code-based blind signature scheme. In *Code-Based Cryptography Workshop*, pages 84–94. Springer, 2021.
9. S. Brands. Untraceable off-line cash in wallets with observers (extended abstract). In D. R. Stinson, editor, *CRYPTO*, volume 773 of *LNCS*, pages 302–318. Springer, 1993.
10. G. Brassard and M. Yung. One-way group actions. In A. J. Menezes and S. A. Vanstone, editors, *CRYPTO'90*, volume 537 of *LNCS*, pages 94–107, Aug. 1991.
11. W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes. CSIDH: An efficient post-quantum commutative group action. In T. Peyrin and S. Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427, Dec. 2018.
12. R. Chairattana-Apirom, L. Hanzlik, J. Loss, A. Lysyanskaya, and B. Wagner. Pi-cut-choo and friends: Compact blind signatures via parallel instance cut-and-choose and more. In Y. Dodis and T. Shrimpton, editors, *CRYPTO 2022*, volume 13509 of *LNCS*, pages 3–31. Springer, 2022.
13. D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *Advances in Cryptology: Proceedings of CRYPTO 1982*, pages 199–203. Plenum Press, New York, 1982.
14. T. Chou, R. Niederhagen, E. Persichetti, T. H. Randrianarisoa, K. Reijnders, S. Samardjiska, and M. Trimoska. Take your MEDS: Digital signatures from matrix code equivalence. In N. El Mrabet, L. De Feo, and S. Duquesne, editors, *AFRICACRYPT 23*, volume 14064 of *LNCS*, pages 28–52, July 2023.
15. J.-M. Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Paper 2006/291, 2006.
16. A. Couvreur, T. Debris-Alazard, and P. Gaborit. On the hardness of code equivalence problems in rank metric, 2021.
17. K. Do, L. Hanzlik, and E. Paracucchi. M&m's: Mix and match attacks on schnorr-type blind signatures with repetition. *IACR Cryptol. ePrint Arch.*, page 1588, 2023.
18. M. Fischlin. Round-optimal composable blind signatures in the common reference string model. In C. Dwork, editor, *Advances in Cryptology - CRYPTO 2006, Proceedings*, volume 4117 of *LNCS*, pages 60–77. Springer, 2006.
19. M. Fischlin and D. Schröder. On the impossibility of three-move blind signature schemes. In H. Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, Proceedings*, volume 6110 of *LNCS*, pages 197–215. Springer, 2010.
20. E. Hauck, E. Kiltz, J. Loss, and N. K. Nguyen. Lattice-based blind signatures, revisited. In D. Micciancio and T. Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - Part II*, volume 12171 of *LNCS*, pages 500–529. Springer, 2020.
21. J. Kastner, J. Loss, and J. Xu. The abe-okamoto partially blind signature scheme revisited. In S. Agrawal and D. Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 279–309, Dec. 2022.
22. S. Katsumata, Y. Lai, and M. Reichle. Breaking parallel ROS: implication for isogeny and lattice-based blind signatures. In Q. Tang and V. Teague, editors, *Public-Key Cryptography - PKC 2024*, volume 14601 of *LNCS*, pages 319–351. Springer, 2024.
23. S. Katsumata, Y.-F. Lai, J. T. LeGrow, and L. Qin. CSI -otter: Isogeny-based (partially) blind signatures from the class group action with a twist. In H. Handschuh and A. Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 729–761, Aug. 2023.

24. S. Katsumata, Y.-F. Lai, J. T. LeGrow, and L. Qin. Csi-otter: isogeny-based (partially) blind signatures from the class group action with a twist. *Designs, Codes and Cryptography*, pages 1–57, 2024.
25. J. Katz, J. Loss, and M. Rosenberg. Boosting the security of blind signature schemes. In M. Tibouchi and H. Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 468–492, Dec. 2021.
26. H. Q. Le, D. H. Duong, W. Susilo, H. T. N. Tran, V. C. Trinh, J. Pieprzyk, and T. Plantard. Lattice blind signatures with forward security. In J. K. Liu and H. Cui, editors, *ACISP 2020*, volume 12248 of *LNCS*, pages 3–22. Springer, 2020.
27. H. Lefmann, K. T. Phelps, and V. Rödl. Rigid linear binary codes. *Journal of Combinatorial Theory, Series A*, 63(1):110–128, 1993.
28. V. Lyubashevsky, N. K. Nguyen, and M. Plançon. Efficient lattice-based blind signatures via gaussian one-time signatures. In G. Hanaoka, J. Shikata, and Y. Watanabe, editors, *Public-Key Cryptography - PKC 2022*, volume 13178 of *LNCS*, pages 498–527. Springer, 2022.
29. A. Petzoldt, A. Szepieniec, and M. S. E. Mohamed. A practical multivariate blind signature scheme. In A. Kiayias, editor, *FC 2017, Revised Selected Papers*, volume 10322 of *LNCS*, pages 437–454. Springer, 2017.
30. D. Pointcheval and J. Stern. Provably secure blind signature schemes. In K. Kim and T. Matsumoto, editors, *Advances in Cryptology - ASIACRYPT '96, Proceedings*, volume 1163 of *LNCS*, pages 252–265. Springer, 1996.
31. D. Pointcheval and J. Stern. Security proofs for signature schemes. In U. M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398, May 1996.
32. D. Pointcheval and J. Stern. New blind signatures equivalent to factorization (extended abstract). In R. Graveman, P. A. Janson, C. Neuman, and L. Gong, editors, *ACM CCS '97, Proceedings*, pages 92–99. ACM, 1997.
33. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000.
34. A. Rostovtsev and A. Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Paper 2006/145, 2006.
35. M. Rückert. Lattice-based blind signatures. In M. Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - Proceedings*, volume 6477 of *LNCS*, pages 413–430. Springer, 2010.
36. C.-P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252, Aug. 1990.
37. P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.
38. X. Yi and K. Lam. A new blind ECDSA scheme for bitcoin transaction anonymity. In S. D. Galbraith, G. Russello, W. Susilo, D. Gollmann, E. Kirda, and Z. Liang, editors, *Proceedings of the 2019 ACM AsiaCCS 2019*, pages 613–620. ACM, 2019.

Appendices

A Public Key Validation—Proofs

A.1 The Protocol for $\mathcal{R}^{(\pm 1)}_{X^{(0)}}$

Proposition A.1. *The protocol of Figure 3 is correct.*

Proof. Suppose that both parties follow the protocol description. Then, with notation as in Figure 3,

$$\begin{aligned}\mathsf{resp} * X^{(\mathsf{chal})} &= (\tilde{A}A^{-\mathsf{chal}}, \tilde{B}B^{-\mathsf{chal}}) * (A^{\mathsf{chal}}, B^{\mathsf{chal}}) * X^{(0)} = \mathsf{comm}^{(1)} \\ \mathsf{resp}^{-T} * X^{(-\mathsf{chal})} &= (\tilde{A}^{-T}(A^T)^{\mathsf{chal}}, \tilde{B}^{-T}(B^T)^{\mathsf{chal}}) * \left(((A^T)^{-\mathsf{chal}}, (B^T)^{-\mathsf{chal}}) * X^{(0)}\right) \\ &= (\tilde{A}^{-T}, \tilde{B}^{-T}) * X^{(0)} = \mathsf{comm}^{(-1)}\end{aligned}$$

and so the verifier will accept the proof, as required. □

Theorem A.2. *The protocol of Figure 3 is 2-special sound.*

Proof. Let $\tau = (\mathsf{comm}, \mathsf{chal} = 0, \mathsf{resp} = (R, S))$ and $\hat{\tau} = (\mathsf{comm}, \widehat{\mathsf{chal}} = 1, \widehat{\mathsf{resp}} = (\hat{R}, \hat{S}))$ be a pair of accepting transcripts. We must have

$$\begin{aligned}(R, S) * X^{(0)} = \ \mathsf{comm}^{(1)} \ &= (\hat{R}, \hat{S}) * X^{(1)} \\ (R^{-T}, S^{-T}) * X^{(0)} = \mathsf{comm}^{(-1)} &= (\hat{R}^{-T}, \hat{S}^{-T}) * X^{(-1)}.\end{aligned}$$

Rearranging, this yields $X^{(1)} = (\hat{R}^{-1}R, \hat{S}^{-1}S) * X^{(0)}$ and

$$X^{(-1)} = (\hat{R}^T R^{-T}, \hat{S}^T S^{-T}) * X^{(0)} = \left((\hat{R}^{-1}R)^{-T}, (\hat{S}^{-1}S)^{-T}\right) * X^{(0)}.$$

So we see that $\mathsf{W} = (\hat{R}^{-1}R, \hat{S}^{-1}S)$ is a witness to $\mathsf{X} = (X^{(1)}, X^{(-1)})$ for the relation $\mathcal{R}^{(\pm 1)}_{X^{(0)}}$, as required. □

Corollary A.3. *The protocol of Figure 3 is perfectly sound.*

Theorem A.4. *The protocol in Figure 3 is honest verifier zero-knowledge (HVZK).*

Proof. Consider the following simulation algorithm Sim:

Comparing lines 103 and 104, with the verifier's acceptance condition in Figure 3, we see that the transcripts produced by Sim will always accept. Moroever, the challenges and responses are distributed identically to those in honest transcripts, since they are uniformly random in both cases. Finally, with the challenge and response fixed, the accepting commitments are uniquely determined, so the distribution of transcripts is in fact the same between the simulator and the honest protocol. Thus the protocol is honest verifier zero-knowledge. □

$\mathsf{Sim}(X^{(0)}; X^{(1)}, X^{(-1)})$

101 : $\mathsf{chal} \xleftarrow{\$} \{0,1\}$

102 : $\mathsf{resp} \xleftarrow{\$} \mathrm{GL}_m(\mathbb{F}_p) \times \mathrm{GL}_n(\mathbb{F}_p)$

103 : $\mathsf{comm}^{(1)} = \mathsf{resp} * X^{(\mathsf{chal})}$

104 : $\mathsf{comm}^{(-1)} = \mathsf{resp}^{-T} * X^{(-\mathsf{chal})}$

105 : **return** $\tau = ((\mathsf{comm}^{(1)}, \mathsf{comm}^{(-1)}), \mathsf{chal}, \mathsf{resp})$

Fig. 7. The simulator for the Sigma protocol of Figure 3.

A.2 The Protocol for $\mathcal{R}^{\mathbb{S}}_{X^{(0)}}$

Proposition A.5. *The protocol of Figure 4 is correct.*

Proof. Suppose that both parties follow the protocol description. Then, with notation as in Figure 4,

$$\begin{aligned}
\mathsf{resp} * X^{(\mathsf{chal})} &= (\tilde{A}A^{-\mathsf{chal}}, \tilde{B}B^{-\mathsf{chal}}) * \left((A^{\mathsf{chal}}, B^{\mathsf{chal}}) * X^{(0)}\right) = \mathsf{comm}^{(1)} \\
\mathsf{resp}^{-T} * X^{(1-\mathsf{chal})} &= (\tilde{A}^{-T}A^{\mathsf{chal}}, \tilde{B}^{-T}B^{\mathsf{chal}}) * \left((A^{1-\mathsf{chal}}, B^{1-\mathsf{chal}}) * X^{(0)}\right) \\
&= (\tilde{A}^{-T}, \tilde{B}^{-T}) * \left((A, B) * X^{(0)}\right) = \mathsf{comm}^{(-1)}
\end{aligned}$$

(where, on the second line, we have used the fact that A and B are symmetric to obtain $(A^{-\mathsf{chal}})^{-T} = (A^{-\mathsf{chal}})^{-1} = A^{\mathsf{chal}}$ (and similar for B)) and so the verifier will accept the proof, as required. □

Theorem A.6. *If $X^{(0)}$ is rigid, then the protocol of Figure 4 is 2-special sound.*

Proof. Let $\tau = (\mathsf{comm}, \mathsf{chal} = 0, \mathsf{resp} = (R, S))$ and $\hat{\tau} = (\mathsf{comm}, \widehat{\mathsf{chal}} = 1, \widehat{\mathsf{resp}} = (\hat{R}, \hat{S}))$ be a pair of accepting transcripts. We must have

$$\begin{aligned}
(R, S) * X^{(0)} = \ \mathsf{comm}^{(1)} \ &= (\hat{R}, \hat{S}) * X^{(1)} \\
(R^{-T}, S^{-T}) * X^{(1)} = \mathsf{comm}^{(-1)} &= (\hat{R}^{-T}, \hat{S}^{-T}) * X^{(0)}
\end{aligned}$$

Rearranging these, we have

$$X^{(1)} = (\hat{R}^{-1}R, \hat{S}^{-1}S) * X^{(0)} = (R^T\hat{R}^{-T}, S^T\hat{S}^{-T}) * X^{(0)}.$$

Since $X^{(0)}$ is rigid, then for some $\alpha, \beta \in \mathbb{F}$ we have

$$\begin{aligned}
\hat{R}^{-1}R = \alpha R^T\hat{R}^{-T} = \alpha(\hat{R}^{-1}R)^T \\
\hat{S}^{-1}S = \beta S^T\hat{S}^{-T} = \beta(\hat{S}^{-1}S)^T
\end{aligned}$$

Thus $(\hat{R}^{-1}R, \hat{S}^{-1}S)$ is a witness to the fact that $X^{(1)} \in \mathcal{L}^{\mathbb{S}}_{X^{(0)}}$. □

Corollary A.7. *The protocol of Figure 4 is perfectly sound.*

$\mathsf{Sim}(X^{(0)}; X^{(1)})$

101 : $\mathsf{chal} \xleftarrow{\$} \{0,1\}$

102 : $\mathsf{resp} \xleftarrow{\$} \mathrm{GL}_m(\mathbb{F}_p) \times \mathrm{GL}_n(\mathbb{F}_p)$

103 : $\mathsf{comm}^{(1)} = \mathsf{resp} * X^{(\mathsf{chal})}$

104 : $\mathsf{comm}^{(-1)} = \mathsf{resp}^{-T} * X^{(1-\mathsf{chal})}$

105 : **return** $\tau = ((\mathsf{comm}^{(1)}, \mathsf{comm}^{(-1)}), \mathsf{chal}, \mathsf{resp})$

Fig. 8. The simulator for the Sigma protocol of Figure 4.

Theorem A.8. *The protocol of Figure 4 is honest verifier zero-knowledge (HVZK).*

Proof. Consider the following simulation algorithm Sim:

Comparing lines 103 and 104, with the acceptance condition of the verifier in Figure 4, we see that the transcripts produced by Sim will always accept. Morever, the challenges and responses are distributed identically to those in honest transcripts, since they are uniformly random in both cases. Finally, with the challenge and response fixed, the commitments which lead to acceptance are uniquely determined, so the distribution of transcripts is in fact the same between the simulator and the honest protocol. Thus the protocol of Figure 4 is honest verifier zero-knowledge. □