Wombat: Post-Quantum Blind Signature from Standard Group Action Assumptions and More *

Lucjan Hanzlik[1], Yi-Fu Lai[2], Eugenio Paracucchi[1], and Edoardo Persichetti[3]

[1] CISPA Helmholtz Center for Information Security, Germany
[2] Shanghai Jiao Tong University, China
[3] Florida Atlantic University, USA

Abstract. A recent work by Hanzlik et al. (Asiacrypt'25) introduced *Tanuki*, a family of blind-signature frameworks based on non-commutative cryptographic group actions. Tanuki develops new techniques to obtain concurrently secure blind signatures, and admits compact instantiations in two distinct regimes: (i) an isogeny-based instantiation from the CSI-FiSh group action with signatures of about 4.5 KB, and (ii) a code-based instantiation from LESS and the code-equivalence group action with signatures around 64 KB. To the best of our knowledge, these are the first efficient blind-signature constructions in the isogeny- and code-based settings that support concurrent executions. Despite this advance, the Tanuki frameworks rely on a non-standard and interactive assumption, namely the so-called "one more" vectorization assumption. Given several structural attacks and vulnerabilities discovered in various group action instantiations, relying on non-standard assumptions can raise concerns. In this work we present a new framework building upon Tanuki's techniques that achieves concurrent security while achieving better performance, and relying only on the standard group action hardness assumption, the vectorization problem (also known as the group action inversion problem). For the LESS instantiation, we apply dedicated code-based techniques to reduce signature sizes by a factor of 14.5. These improvements come with rigorous reductions to the standard problem, do not weaken the security claims, and are directly applicable to the LESS instantiations of Tanuki. As a result, our isogeny-based and code-based instantiations yield signature sizes of 8.89 and 8.84 KB, respectively, and retain concurrent security under the standard group-action inversion assumption.

1 Introduction

Blind signatures, introduced by Chaum [Cha82], are a fundamental cryptographic primitive that let a signer issue signatures on user-chosen messages without learning the message contents. This strong privacy property underpins a wide

* Part of this work, namely the application of the canonical form, appeared in the preprint available at https://ia.cr/2025/1585; the corresponding part is clearly indicated in this paper. The present paper combines those results with the unpublished results on the schemes based on the vectorization problem.

range of applications, from classic systems such as e-cash, e-voting, and anonymous credentials, to modern deployments including Privacy Pass [DGS+18], Tumblebit [HAB+17], Apple Private Relay [App21], Private Cloud Compute [App24], and Microsoft U-Prove [Mic24].

The key privacy requirement for blind signatures is *blindness*: the signer cannot link an interactive signing sessions with the resulting signatures, nor learn meaningful information about the signed message. At the same time, blind signature schemes must provide *unforgeability*, typically formalized as *one-more unforgeability*. Informally, one-more unforgeability demands that no adversary can produce $\ell + 1$ valid signatures after at most ℓ interactions with the signing oracle. This notion can be considered either in a *sequential* model, where sessions occur one after another, or in a *concurrent* model, where an adversary may interleave many sessions.

Practical blind signatures based on classical assumptions include Chaum's RSA-based construction [Cha82], discrete-logarithm based schemes [PS00, HKL19, CAHL+22], and pairing-based designs [Bol03, FHKS16, HLW23]. By contrast, the post-quantum landscape offers fewer efficient and widely accepted constructions, both in terms of the diversity of assumptions and in efficiency.

There are broadly two paradigms for building blind signatures. The first follows Fischlin's generic template [Fis06]. This approach has seen active development mostly in the lattice setting [AKSY22, dPK22, BLNS23, BFM+25], where recent advances in lattice-friendly non-interactive zero-knowledge proofs for structured languages have substantially reduced signature sizes and improved practicality [BLS19, LNP22]. The main efficiency bottleneck of the approach lies in the use of generic zero-knowledge proofs (ZKPs) during interaction between the signer and the user.

The other approach is to adapt sigma-protocol-based signatures into blind signatures [HKLN20, KLLQ23]. This line of work had long faced two key challenges: the scarcity of general transformation techniques and the lack of clear concurrent-security guarantees. For a considerable time, only two adapting methods were known—one for linear hash-function families [HKLN20] and one using a twist-type operation [KLLQ23]. The more severe challenge lies in the absence of provable security in the concurrent setting, due to the exponential security loss inherent in the proofs. Indeed, two recent attacks [KLR24, DHP24] have shown concrete vulnerabilities in several promising post-quantum instantiations [AEB20, AHJ21, KLLQ23] under only four concurrent executions.

In this research line, a recent work [HLM+25] proposes a new framework for concurrently secure blind signatures from cryptographic group actions. Straightforward instantiations based on CSIDH (CSI-FiSh) and LESS yield signature sizes of 4.5 KB and 64 KB respectively. Despite this breakthrough, a few limitations remain. First, the framework relies on a non-standard, interactive assumption called the *one-more* vectorization problem, which gives the adversary access to a vectorization (namely, group action inverse problem) oracle and is therefore not falsifiable. Practically, several distinct variants (non-equivalent) of the group action inverse problem (GAIP) have either been broken by concrete at-

tacks [BCD+24] or shown to admit measurable losses in security under classical attacks [KLLQ24, BBCK24, BCD+24, BCF25]. Hence relying on non-standard assumptions could incur a potential risk in practice.

The Obstacles. Technically, [HLM+25] introduces a GAIP oracle because the simulator needs a way to emulate the signer during a signing query. Unlike Fiat–Shamir-based signatures, where a simulator can program the random oracle and thus choose the challenge, in the blind signature setting the challenge is selected by the user and therefore cannot be programmed even if the user is honest. A GAIP oracle enables the simulator to produce the signer's second message without knowing the secret key.

A classical remedy to this simulation barrier is the parallel OR-proof application developed by Abe and Okamoto [AO00, KLX22]. This is also successfully applied in other contexts, for instance, in CSI-Otter [KLLQ23] without requiring interactive assumptions [AO00, KLX22]. Colloquially, OR-proofs, in general, enable a prover to show that it knows the witness for one of many statements, or that one out of many statements is true. Hence, a simulator, knowing only one of the secrets, can still generate a transcript. Unfortunately, the method is fundamentally incompatible with the Tanuki frameworks.

The incompatibility arises from a structural conflict between two core design choices in Tanuki: the *fixed-weight challenge space* $\mathcal{C}$ and its *permutation-based mechanism* for challenge blinding. OR-proofs are inherently algebraic and require operations (e.g., addition or multiplication) within a group structure. Since $\mathcal{C}$ lacks any natural algebraic properties, one might attempt to impose them by defining a bijection $\phi : \mathcal{C} \to \mathbb{Z}_N$ for $N = |\mathcal{C}|$. However, as detailed in [HLM+25, App. D], this artificial structure conflicts with the permutation-based blinding, which makes the technique invalid for the correctness. Roughly speaking, in the Tanuki's permutation mechanism, the user will use a permutation to permute the commitment com as $\pi(\mathsf{com})$, and mask the hash value ch as $\pi^{-1}(\mathsf{ch})$ and forward it to the signer. Then, after obtaining the response resp, the user will permute it again $\pi(\mathsf{resp})$. If one wants to apply the parallel OR-proof [AO00] here by defining an artificial group operation $\oplus$ to decompose $\mathsf{ch} = \mathsf{ch}_0 \oplus \mathsf{ch}_1$, an immediate obstacle follows as it requires further artificial operation $\oplus'$ for the "distributive law" to have $\pi^{-1}(\mathsf{ch}) = \pi_0^{-1}(\mathsf{ch}_0 \oplus \mathsf{ch}_1) = \pi_0^{-1}(\mathsf{ch}_0) \oplus' \pi_1^{-1}(\mathsf{ch}_1)$ for a predetermined π_0 or π_1. However, such operations do not seem to exist in general. The only exception noted in [HLM+25, App. D] is a highly restricted scenario with two public keys and a binary challenge space, where the blinding permutation (a swap) can be encoded as an operation in $\mathbb{Z}_2$. This special case, however, is of limited practical interest as it fails to support key/signature size trade-offs and, critically, does not achieve concurrent security. Also, other variants of OR-proofs [AOS02, BKP20, BS24] do not apply to blind signatures. A similar issue caused by the structured challenge space is also mentioned [BDK+22, Footnote 6]. Thus, to the best of our knowledge, the unique design of the Tanuki framework prevents the application of any existing techniques to

achieve concurrent security under standard assumptions.

Contributions.

We make contributions both in theory and in practice. In this work, we develop a new blind signature framework that achieves concurrent security in the *algebraic group action model* (AGAM) under the standard *group action inverse problem* (GAIP) assumption. We name the framework Wombat, a reordered acronym for "Blind signatures from group Actions With Original assumptions, compatible with Tanuki Mechanisms". In comparison with the Tanuki frameworks [HLM+25], our scheme offers clear improvements. First, while Tanuki relies on the non-standard *one-more vectorization assumption* (OMVP), our framework is based on GAIP, a canonical and well-studied assumption for cryptographic group actions, thereby yielding stronger and more trustworthy security guarantees. Second, our proof establishes concurrent security for arbitrary values of the parameters κ and μ (the number of keys and repetitions, respectively), whereas Tanuki required the restriction $\kappa = \mathsf{poly}(\lambda)$ and was essentially applicable only to the special cases $\kappa = \mu$ and $\kappa = \mu/2$ (see Remark 4.6). Having this freedom on κ and μ is important both theoretically and practically: it enables the full parameter choice for the trade-off between the key size and the signature size for provable security.

For the practicability of the code-based instantiations, we develop new properties of the canonical forms. Since the use of canonical form breaks the group action structure, which is essential for the proofs both in our work and [HLM+25], we give two distinct non-black-box reductions that establish blindness and one-more unforgeability. As a result, we reduce the resulting signature size by a factor of 14.5 without compromising security guarantee. Our techniques also extend to Tanuki's LESS instantiations; we give details and the technical overview in Section 5.1 and Appendix B.

Technical Overview.

To address the first obstacle mentioned above, we develop a novel "OR-proof" from the base sigma protocol. Rather than proving knowledge of a witness for one of two statements, we prove knowledge of a pair of witnesses for an unordered pair of statement-pairs. We consider a public statement contains two arrays of set elements

$$(\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa}),$$

and the extraction witness is a triple (g_0, g_1, b) with $b \in \{0, 1\}$ such that, for some distinct indices $i, j \in [\kappa]$,

$$(g_0, g_1) \star (\mathsf{pk}_{b,i}, \mathsf{pk}_{1-b,i}) = (\mathsf{pk}_{0,j}, \mathsf{pk}_{1,j}).$$

Formally, for a full statement

$$\big((x_0, x_1), (\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa})\big)$$

the prover's secret key is

$$\mathsf{sk} = \big((g_{0,1}, \ldots, g_{0,\kappa}), (g_{1,1}, \ldots, g_{1,\kappa}), \mathsf{b}\big) \in G^{2\kappa} \times \{0, 1\}^{\kappa},$$

where $\mathsf{b} = (\mathsf{b}_1, \ldots, \mathsf{b}_\kappa)$ and, for each i, $\mathsf{pk}_{0,i} = g_{0,i} \star x_{\mathsf{b}_i}$ and $\mathsf{pk}_{1,i} = g_{1,i} \star x_{1-\mathsf{b}_i}$.

First, the prover samples $h_{b,i} \in G$ for every $b \in \{0,1\}$ and $i \in [\mu]$, sets $(y_{0,i}, y_{1,i}) = (h_{0,i}, h_{1,i}) \star (x_0, x_1)$, applies an independent and random swap to each pair $(y_{0,i}, y_{1,i})$, and commits to $\mathbf{y} = ((y_{0,1}, y_{1,1}), \ldots, (y_{0,\mu}, y_{1,\mu}))$. After the verifier issues a challenge string $c \in \{1, \ldots, \kappa\}^\mu$, for each $i \in [\mu]$ the prover computes $(s_{0,i}, s_{1,i}) = \left(h_{\mathsf{b}_{c(i)},i}\, g_{0,c(i)}^{-1},\ h_{1-\mathsf{b}_{c(i)},i}\, g_{1,c(i)}^{-1}\right)$ and forwards to the verifier. The verifier accepts iff for every $i \in [\mu]$ the unordered pair obtained by $(s_{0,i}, s_{1,i}) \star (\mathsf{pk}_{0,c(i)}, \mathsf{pk}_{1,c(i)})$ matches the committed pair up to order, i.e. either $(y_{0,i}, y_{1,i})$ or $(y_{1,i}, y_{0,i})$.

A key feature of our protocol is that it does not rely on an HVZK (honest-verifier zero-knowledge) simulator. Consequently, it avoids operations over the challenge space to hide the simulated challenge. The random swapping in the commitment enable transcript simulation, and thereby the protocol is witness indistinguishable. Hence, the design is compatible with fixed-weight challenge spaces, which are widely used in group action literature [BKP20, BDK+23, BPS+23]. Then, under the fixed-weight challenge space setting, we apply the permutation mechanism and the key-randomization of [HLM+25] to turn it into a blind signature. Looking ahead, in the reduction, the simulator assigns the GAIP challenge as x_1 and constructs the key arrays $(\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa})$ and $(\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa})$. Upon witness extraction, with probability $1/2$, we extract a group element linking the key chains $(\mathsf{pk}_{\mathsf{b}_1,1}, \ldots, \mathsf{pk}_{\mathsf{b}_\kappa,\kappa})$ and $(\mathsf{pk}_{1-\mathsf{b}_1,1}, \ldots, \mathsf{pk}_{1-\mathsf{b}_\kappa,\kappa})$, thereby giving the GAIP solution.

For the LESS instantiation, we exploit canonical forms for the linear-equivalence problem [CPS23, BEP+25] to compress signatures. The high-level idea is to replace explicit group elements by canonical representatives: fixing a subgroup $S \leq G$, we can represent an object x via a representative of its coset in G/S, and we represent the image $x_1 = g \star x$ by the canonicalization map $\mathsf{CF}(x_1)$. With an appropriate choice of S, this yields an immediate size reduction whenever G/S is substantially smaller than G.

The difficulty is that this compression no longer fits the standard "group action" abstraction. Canonicalization discards information needed for composition such that the induced mapping is not associative: given $x \xrightarrow{g_1} x_1$ and $x_1 \xrightarrow{g_2} x_2$, the composition of the chosen representatives for g_1 and g_2 typically does not define a consistent representative that maps x to x_2. This creates two obstacles. First, the group-action-based frameworks do not apply verbatim, since it relies on associativity on both the signer and user sides. Second, the existing proof frameworks (including ours and that of [HLM+25]) crucially use the group-action model; after canonical-form compression, the modified scheme falls outside their scope regardless of how small the syntactic changes appear.

We address these issues via three strategies. To handle the associativity issue and reduce the signature size, we enforce an adaptive update of the signer and user randomness. In both cases we prove that these adaptive steps will not compromise blindness. To see this, we develop a few new properties of canonical forms and rewrite the proof accordingly: instead of fixing the signer's view and comparing transcript probabilities, we show that for any two fixed signer views

the induced signature distributions are identical. For one-more unforgeability, we give non-black-box reductions from our optimized schemes to the corresponding original group action-based framework instantiations. Even though canonicalization removes a strict group-action structure, these reductions show that forging in our scheme is no easier than forging in the original instantiation, thereby transferring one-more unforgeability to the compressed setting.

2 Preliminaries

Notations. We let $\mathbb{N}$ denote the set of natural numbers. For $n \in \mathbb{N}$, we denote the set $\{1, \cdots, n\}$ as $[n]$. For a subset $J \subseteq [n]$, we denote the complement by $\bar{J} := [n] - J$. For $\kappa \in \mathbb{N}$ we denote with S_κ the set of all the permutations of $[\kappa]$. If $\sigma, \sigma' \in S_\kappa$ are permutations then we denote with $\sigma \circ \sigma'$ their composition (as functions). For two vectors $\mathbf{b}_1, \mathbf{b}_2$, we denote $\mathbf{b}_1 =_\mathsf{B} \mathbf{b}_2$ if the zeros in the vectors are at the same positions. For an element $b \in \{0,1\}$, we define its conjugation as $\bar{b} = 1 - b \in \{0,1\}$. For a group action $(G, X, \star)$ and $x \in X$, we extend the action to an array $(g_1, \cdots, g_n) \star x := (g_1 \star x, \cdots, g_n \star x) \in X^n$. For a list $C = [c_1, \ldots, c_n]$ with $c_i \in G$ and set element $x \in X$ we also define $C \star x := [c_1 \star x, \ldots, c_n \star x]$.

2.1 Cryptographic Group Actions

The study of post-quantum group actions has seen growing interest in recent years. This line of research began in the commutative setting, notably through isogeny-based constructions [CLM+18, ADMP20], and has since demonstrated broad applicability to advanced cryptographic primitives [ADMP20, BKP20, DM20, BDK+22, Lai24, MR24]. Parallel or the inspired developments [BBB+23, CNP+23, BDN+23, BBN+22, BCD+24, BBMP24, BBD+25, BCF25] have emerged in the non-commutative setting, with increasing attention in both code-based and lattice-based cryptography. Along with the rapid development, a number of abstract models have been introduced to streamline security analyses and to formalize hardness assumptions or impossibility results [ADMP20, BGZ23, DHK+23, BBD+25]. We describe the cryptographic group action used in this work. Unlike the setting considered in [ADMP20], we do not require the group to be commutative. Consequently, there is no known general subexponential algorithm for solving the underlying assumptions.

Definition 2.1 (Group action). *Let $(G, \cdot)$ be a group. We say G* acts *on a set X by a function $\star : G \times X \to X$ or $(G, X, \star)$ is a* group action *if it satisfies the following properties. 1) Identity: for any $x \in X$, we have $e \star x = x$ where e is the identity element of G. 2) Compatibility: for any $g, h \in G$ and any $x \in X$, we have $(gh) \star x = g \star (h \star x)$.*

Definition 2.2 (Cryptographic Group Action). *Let $(x, G, X, \star)$ be given where $(G, X, \star)$ defines a group action and $x \in X$. We say that $(x, G, X, \star)$ is a* cryptographic group action *if there exist computationally efficient algorithms for the following tasks:*

- ***Unique set element representation:*** *Given any element $x \in X$, compute a unique representation of x.*
- ***Unique group element representation:*** *Given any element $g \in G$, compute a unique representation of g.*
- ***Group sampling:*** *Sampling over G uniformly at random (or statistically).*
- ***Group element inversion:*** *Given any $g \in G$, compute its inverse g^{-1}.*
- ***Group action:*** *Given any $x \in X$ and $g \in G$, compute $g \star x$.*

2.2 Hardness Assumptions and the Algebraic Group Action Model

Definition 2.3 (Group action inverse problem (GAIP/GA-DLog)). *Let $\Pi = (x, G, X, \star)$ be a cryptographic group action. The* group action inverse problem *asks, given $y = g \star x$ where $g \leftarrow\!\$\ G$ to find an element $g' \in G$ such that $g' \star x = y$. We define the advantage of a p.p.t. adversary $\mathcal{A}$ in solving* GAIP *with respect to Π as $\mathsf{Adv}^{\mathsf{GAIP}}_{\Pi,\mathcal{A}}(\lambda) := \Pr[\mathcal{A}(y) \in G \text{ such that } \mathcal{A}(y) \star x = y]$, where the probability is over the randomness of sampling over G and the internal randomness of $\mathcal{A}$.*

Definition 2.4 (Algebraic Algorithm [HLM+25]). *Let the cryptographic group action $\Pi = (x, G, X, \star)$ be given. We say that an algorithm $\mathcal{A}$ is* algebraic *if,* for any set element $y \in X$ *that it outputs, it also outputs a group element g and an index $i \in \mathbb{N}$, i.e., $(y, (g, i)) \leftarrow\!\$\ \mathcal{A}(\overrightarrow{x})$ such that $y = g \star x_i$.*

For an adversary $\mathcal{A}$ that has access to oracles during its runtime, we impose the above restriction on all group elements that it outputs to an oracle. Similarly, all group elements that $\mathcal{A}$ receives through oracle interactions are treated as inputs to $\mathcal{A}$; hence, such group elements become part of $\overrightarrow{x}$ when $\mathcal{A}$ outputs set elements (and hence algebraic coefficients) at a later point. In the *algebraic group action model* (AGAM), all algorithms are treated as algebraic algorithms. The model is an analogue to algebraic group model [FKL18] and it has been considered in [DHK+23] in the commutative setting. The model is also used in [HLM+25].

2.3 Sigma Protocol

Definition 2.5 (Sigma Protocol). *A sigma protocol Π_Σ is a three-move proof system for a relation R, and it consists of PPT algorithms $(\mathsf{Prove} = (\mathsf{Prove}_1, \mathsf{Prove}_2), \mathsf{Verify} = (\mathsf{Verify}_1, \mathsf{Verify}_2))$, where Verify_2 is deterministic. We assume Prove_1 and Prove_2 share states and so does Verify_1 and Verify_2. Let $\mathcal{C}$ denote the challenge space. Then, Π_Σ proceeds as follows.*

- *The prover, on input $(\mathsf{st}, \mathsf{wt}) \in R$, runs $\mathsf{com} \leftarrow \mathsf{Prove}_1(\mathsf{st}, \mathsf{wt})$ and sends a* commitment com *to the verifier.*
- *The verifier runs $\mathsf{ch} \leftarrow\!\$\ \mathsf{Verify}_1(1^\lambda)$, drawing a random challenge from $\mathcal{C}$, and sends it to the prover.*
- *The prover, given ch, runs $\mathsf{resp} \leftarrow\!\$\ \mathsf{Prove}_2(\mathsf{st}, \mathsf{wt}, \mathsf{ch})$ and returns a* response resp *to the verifier.*

– *The verifier runs* $\mathsf{Verify}_2(\mathsf{st}, \mathsf{com}, \mathsf{ch}, \mathsf{resp})$*, outputs* $\top$ *(accept) or* $\bot$ *(reject).*

We assume the statement st *is always given as input to both the prover and the verifier. The protocol* transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *is said to be valid in case* $\mathsf{Verify}_2(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *outputs* $\top$*.*

Definition 2.6 (Correctness). *A sigma protocol* Π_Σ *is said to be correct if for all* $\lambda \in \mathbb{N}$*,* $(\mathsf{st}, \mathsf{wt}) \in R$ *and the prover and the verifier both follow the protocol specification, the verifier always outputs* $\top$*.*

Definition 2.7 (Honest Verifier Zero-Knowledge). *We say* Π_Σ *is* honest-verifier-zero-knowledge *for relation* R *if there exists a PPT simulator* $\mathcal{S}$ *such that for any statement-witness pair* $(\mathsf{st}, \mathsf{wt}) \in R$*,* $\mathsf{ch} \in \mathcal{C}$*,* $\lambda \in \mathbb{N}$ *and any computationally-unbounded adversary* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{HVZK}}_{\Pi_\Sigma}(\mathcal{A}) := |\Pr[\mathcal{A}(P(\mathsf{st}, \mathsf{wt}, \mathsf{ch})) = 1] - \Pr[\mathcal{A}(\mathcal{S}(\mathsf{st}, \mathsf{ch})) = 1]| = \mathsf{negl}(\lambda),$$

where $P = (P_1, P_2)$ *is a prover running on* $(\mathsf{st}, \mathsf{wt})$ *with a challenge fixed to* ch *and the probability is taken over the randomness used by* (P, V)*.*

Below, for the special soundness property, the extraction algorithm is only required to recover a "weaker" witness in R_{ss} rather than in R used in the real protocol. In many applications including ours, the capability of extracting from this wider relation suffices as long as recovering a weaker witness remains hard.

Definition 2.8 ((Relaxed) Special Soundness). *We say a sigma protocol* Π_Σ *has (resp, relaxed) special soundness for the relation* R *if there exists a polynomial-time extraction algorithm* $\mathcal{E}$ *such that, given a statement* st *and any two valid transcripts* $(\mathsf{com}, \mathsf{ch}_1, \mathsf{resp}_1), (\mathsf{com}, \mathsf{ch}_2, \mathsf{resp}_2)$ *relative to* st *with distinct* $\mathsf{ch}_1, \mathsf{ch}_2$*,* $\mathcal{E}$ *outputs a witness* wt *satisfying* $(\mathsf{st}, \mathsf{wt})$ *in* R *(resp,* $R \subseteq R_{\mathsf{ss}}$*).*

Proposition 2.9 ([CDS94, Prop. 1]). *Let* Π_Σ *be a sigma protocol for relation* R*. If* Π_Σ *is honest-verifier zero-knowledge, then* Π_Σ *is witness-indistinguishable: for every statement* x *with arbitrary two valid witnesses* w_0, w_1*, the distributions of transcripts produced through* w_0 *or* w_1 *are indistinguishable.*

2.4 Three-move Blind Signature Schemes

We will now recall the notion of blind signatures. We focus on a subclass called three-move blind signatures, where the exchange between the user and signer is finished in three moves, similar to Sigma protocols [HLM+25].

Definition 2.10. *A three-moves blind signature scheme* $\mathsf{BS} = (\mathsf{KeyGen}, \mathsf{Signer}, \mathsf{User}, \mathsf{Verify})$ *consists of the following p.p.t. algorithms:*

Key generation. *On input the security parameter* 1^λ*, the probabilistic algorithm* KeyGen *outputs a public key* pk *and a secret key* sk*.*

Signer. *The* $\mathsf{Signer} = (\mathsf{Signer}_1, \mathsf{Signer}_2)$ *consists of two algorithms:*

$\mathsf{ExpBlind}(\lambda)$

$b \leftarrow\!\!\$\ \{0,1\}$
$b_1 \leftarrow b; b_2 \leftarrow 1-b$
$b' \leftarrow \mathcal{A}^{\mathsf{Init},\mathsf{U}_1,\mathsf{U}_2}(\lambda)$
return $b = b'$

$\mathsf{U}_1(sid, R)$

if $sid \notin \{1,2\} \vee \mathbf{sess}_{sid} :\neq \text{init}$: **return** $\perp$
$\mathbf{sess}_{sid} \leftarrow \text{open}$
$R_{sid} \leftarrow R$
$(c_{sid}, \mathsf{st}_{sid}) \leftarrow \mathsf{User}_1(\mathsf{pk}, m_{b_{sid}}, R_{sid})$
return (sid, c_{sid})

$\mathsf{Init}(\tilde{m_0}, \tilde{m_1}, \tilde{\mathsf{pk}})$

$m_0 \leftarrow \tilde{m_0}, m_1 \leftarrow \tilde{m_1}$
$\mathsf{pk} \leftarrow \tilde{\mathsf{pk}}$
$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow \mathsf{init}$

$\mathsf{U}_2(sid, s)$

if $\mathbf{sess}_{sid} \neq \text{open}$: **return** $\perp$
$s_{sid} \leftarrow s$
$\mathsf{sig}_{b_{sid}} \leftarrow \mathsf{User}_2(\mathsf{pk}, s_{sid}, \mathsf{st}_{sid})$
$\mathbf{sess}_{sid} \leftarrow \mathsf{closed}$
if $\mathbf{sess}_1 = \mathbf{sess}_2 = \mathsf{closed}$:
  if $\mathsf{sig}_0 = \perp \vee \mathsf{sig}_1 = \perp$: **return** $(\perp, \perp)$
  return $(\mathsf{sig}_0, \mathsf{sig}_1)$
return (sid, closed)

Fig. 1. Malicious-Key Blindness experiment $\mathsf{ExpBlind}$.

$\mathsf{Signer}_1(\mathsf{sk})$**:** *on input a secret key* sk*, the probabilistic algorithm* Signer_1 *outputs an internal state* $\mathsf{st}_{\mathsf{Signer}}$ *and a commitment* R*.*

$\mathsf{Signer}_2(\mathsf{sk}, c, \mathsf{st}_{\mathsf{Signer}})$**:** *on input a challenge* $c \in \mathcal{C}$*, a secret key* sk *and an internal state* st*, the algorithm* Signer_2 *outputs a response* s*.*

User. *The* $\mathsf{User} = (\mathsf{User}_1, \mathsf{User}_2)$ *consists of two algorithms:*

$\mathsf{User}_1(\mathsf{pk}, m, R)$**:** *on input a public key* pk*, a message* m *and a signer commitment* R*, the probabilistic algorithm* User_1 *outputs a challenge* $c \in \mathcal{C}$ *and an internal state* $\mathsf{st}_{\mathsf{User}}$*.*

$\mathsf{User}_2(\mathsf{pk}, s, \mathsf{st}_{\mathsf{User}})$**:** *on input a public key* pk*, a signer response* s *and an internal state* $\mathsf{st}_{\mathsf{User}}$*, the probabilistic algorithm* User_2 *outputs a signature* sig *on message* m*.*

Verification. *On input a public key* pk*, a message* m*, and a signature* sig*, the algorithm* Verify *outputs* 1 *to indicate the prover is valid, and* 0 *otherwise.*

A three-moves blind signature scheme BS *is correct if for every* λ*, for every key pair* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda)$*, for every message and signature pair* (m, sig) *generated from the interaction between an user and a signer, we have* $\mathsf{Verify}(\mathsf{pk}, m, \mathsf{sig}) = 1$*. We will omit "three-moves" and will say blind signature for brevity.*

We consider blind signatures secure if they have the following two security properties. On the one hand, we want the signature output by the user to be not linkable to its identity: this is called the *blindness* property. On the other hand, after ℓ interactions with the signer, we require that the user cannot output more than ℓ valid signatures. This property is called *one-more-unforgeability*. We formalize it using the definition from [KLX22].

$\mathsf{ExpOUnf}(\lambda)$

$(\mathsf{sk}, \mathsf{pk}) \leftarrow\!\!\$\ \mathsf{KeyGen}(\lambda)$
$sid \leftarrow 0$
$((m_1, \mathsf{sig}_1), \ldots, (m_\ell, \mathsf{sig}_\ell)) \leftarrow \mathcal{A}^{\mathsf{Sign}_1, \mathsf{Sign}_2}(\mathsf{pk})$
if $\exists i \neq j : m_i = m_j$: **return** 0
if $\exists i \in [\ell] : \mathsf{Verify}(\mathsf{pk}, m_i, \mathsf{sig}_i) = 0$: **return** 0
$Q_{\mathsf{Sign}_2} \leftarrow |\{k | \mathbf{sess}_k = \mathsf{closed}\}|$
if $\ell \geq Q_{\mathsf{Sign}_2} + 1$: **return** 1
return 0

$\mathsf{Sign}_1(\cdot)$

$sid \leftarrow sid + 1$
$\mathbf{sess}_{sid} \leftarrow \mathsf{open}$
$(\mathsf{st}_{sid}, R_{sid}) \leftarrow \mathsf{Signer}_1(\mathsf{sk})$
return (sid, R_{sid})

$\mathsf{Sign}_2(sid, c)$

if $\mathbf{sess}_{sid} \neq \mathsf{open}$: **return** $\perp$
$\mathbf{sess}_{sid} \leftarrow \mathsf{closed}$
$s_{sid} \leftarrow \mathsf{Signer}_2(\mathsf{sk}, c, \mathsf{st}_{sid})$
return s_{sid}

Fig. 2. Unforgeability experiment $\mathsf{ExpOUnf}$.

Definition 2.11 (Malicious-Key Blindness). *A blind signature scheme* BS *is blind under malicious key if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for all* λ *and all PPT adversaries* $\mathcal{A}$*, the following holds:* $\mathsf{Adv}^{\mathbf{blind}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{Adv}^{\mathbf{blind}}_{\mathcal{A}}(\lambda) = |\Pr[\mathsf{ExpBlind}(\lambda) = 1] - \frac{1}{2}|$ *and the experiment* $\mathsf{ExpBlind}$ *is defined in Figure 1. When* $\mathsf{Adv}^{\mathbf{blind}}_{\mathcal{A}}(\lambda) = 0$*, we say the blind signature scheme* BS *has perfect blindness.*

Definition 2.12 (Concurrent ℓ-one-more unforgeability). *A blind signature* BS *is* ℓ*-one-more unforgeable in the concurrent setting if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for all* λ *and all PPT adversaries* $\mathcal{A}$*, the following holds:* $\mathsf{Adv}^{\mathsf{cOMUF}}_{\mathcal{A},\ell}(\lambda) \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{Adv}^{\mathsf{cOMUF}}_{\mathcal{A},\ell}(\lambda) = \Pr[\mathsf{ExpOUnf}(\lambda) = 1]$ *and* $\mathsf{ExpOUnf}$ *is the experiment defined in Figure 2.*

3 Our Base Sigma Protocol

We start with Fig. 3 the base sigma protocol underlying our blind signature where we introduce the subroutine $\mathsf{rswap}(x, y)$ that swaps the values of variables x and y with probability $1/2$. Let $(x_0, X, G, \star)$ be a cryptographic group action, which is the public parameter.

For a statement involving $\mathsf{pk} = ((\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa}))$, the base sigma protocol enable the prover to show the (partial) knowledge of a witness for the relation $R_{\mathsf{ss}} \subseteq X^{2\kappa} \times G^2 \times \{0, 1\}$

$$R_{\mathsf{ss}} = \left\{ ((\mathsf{pk}_{0,m}, \mathsf{pk}_{1,m})_m, (g_0, g_1, b)) \;\middle|\; \begin{array}{c} (g_0, g_1) \star (\mathsf{pk}_{b,i}, \mathsf{pk}_{1-b,i}) = (\mathsf{pk}_{0,j}, \mathsf{pk}_{1,j}) \\ \text{for some distinct } i, j \in [\kappa] \end{array} \right\}.$$

Formally, R_{ss} is said to be the *relaxed* relation for the sigma protocol in the literature. The full witness of the prover consists of a set element $x_1 \in X$ in the same

Prover $(\mathsf{pk}, \mathsf{sk})$		**Verifier** (pk)
$\mathbf{h}_0, \mathbf{h}_1 \xleftarrow{\$} G^\mu$		
for $i \in [\mu]$ **do**: $(y_{0,i}, y_{1,i}) \leftarrow$		
$\quad (h_{0,i}, h_{1,i}) \star (x_0, x_1)$; $\mathsf{rswap}(y_{0,i}, y_{1,i})$		
$\mathbf{y} \leftarrow ((y_{0,1}, y_{1,1}), \ldots, (y_{0,\mu}, y_{1,\mu}))$	$\xrightarrow{\mathbf{y}}$	
	$\xleftarrow{c}$	$c \leftarrow\!\$\ \{1, .., \kappa\}^\mu$
for $i \in [\mu]$ **do**: $(s_{0,i}, s_{1,i}) \leftarrow$		
$\quad (h_{\mathsf{b}_{c(i)},i} g_{0,c(i)}^{-1},\ h_{1-\mathsf{b}_{c(i)},i} g_{1,c(i)}^{-1})$		**accept iff for all** $i \in [\mu]$:
$\mathbf{s} \leftarrow ((s_{0,1}, s_{1,1}), \ldots, (s_{0,\mu}, s_{1,\mu}))$,	$\xrightarrow{\mathbf{s}}$	$(s_{0,i}, s_{1,i}) \star (\mathsf{pk}_{0,c(i)}, \mathsf{pk}_{1,c(i)}) =$
		$(y_{0,i}, y_{1,i})$ or $(y_{1,i}, y_{0,i})$

Fig. 3. The base Σ-protocol $\Pi_\Sigma = (\mathsf{Prove} = (\mathsf{Prove}_1, \mathsf{Prove}_2), \mathsf{Verify} = (\mathsf{Verify}_1, \mathsf{Verify}_2))$. Let κ be the number of keys and μ the number of repetitions. The secret key is $\mathsf{sk} = (x_1, (g_{0,1}, \ldots, g_{0,\kappa}), (g_{1,1}, \ldots, g_{1,\kappa}), \mathsf{b}) \in X \times G^{2\kappa} \times \{0,1\}^\kappa$; the public key is $\mathsf{pk} = ((\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa}))$ with $\mathsf{pk}_{0,i} = g_{0,i} \star x_{\mathsf{b}_i}$ and $\mathsf{pk}_{1,i} = g_{1,i} \star x_{1-\mathsf{b}_i}$.

orbit as x_0, a binary string $\mathsf{b} \in \{0,1\}^\kappa$, and group elements $(h_{b,i})_{(b,i)\in\{0,1\}\times G^{2\kappa}}$ such that the statement is generated by $\mathsf{pk}_{0,i} = g_{0,i} \star x_{\mathsf{b}_i}$ and $\mathsf{pk}_{1,i} = g_{1,i} \star x_{1-\mathsf{b}_i}$, which defines the full relation R.

Theorem 3.1. *The sigma protocol presented in Fig. 3 has correctness, 2-special soundness and HVZK for the relation* R_{ss}.

Proof. **Perfect completeness.** For an arbitrary $i \in [\kappa]$, we have

$$\begin{aligned}(s_{0,i}, s_{1,i}) \star (\mathsf{pk}_{0,c(i)}, \mathsf{pk}_{1,c(i)}) &= (h_{\mathsf{b}_{c(i)},i} g_{0,c(i)}^{-1},\ h_{1-\mathsf{b}_{c(i)},i} g_{1,c(i)}^{-1}) \star (\mathsf{pk}_{0,c(i)}, \mathsf{pk}_{1,c(i)}) \\ &= \left(h_{\mathsf{b}_{c(i)},i},\ h_{1-\mathsf{b}_{c(i)},i}\right) \star \left(x_{\mathsf{b}_{c(i)}},\ x_{1-\mathsf{b}_{c(i)}}\right).\end{aligned}$$

The pair is exactly $(y_{\mathsf{b}_{c(i)},i}, y_{1-\mathsf{b}_{c(i)},i})$ before rswap. Hence, the scheme has perfect completeness.

Special soundness. Let $(\mathbf{y}, c, \mathbf{s})$ and $(\mathbf{y}, c', \mathbf{s}')$ be two accepting transcripts that share the same commitment $\mathbf{y}$ with $c \neq c'$. Then there exists an index $k \in [\kappa]$ such that $i := c(k) \neq j := c'(k)$. The group element pairs $(s_{0,k}, s_{1,k})$ and $(s'_{0,k}, s'_{1,k})$ link the same commitment component $(y_{0,k}, y_{1,k})$ respectively to the public key components $(\mathsf{pk}_{0,i}, \mathsf{pk}_{1,i})$ and $(\mathsf{pk}_{0,j}, \mathsf{pk}_{1,j})$. The corresponding bit b can be extracted by comparing the order in which these equalities hold.

$\underline{\mathsf{KeyGen}(\lambda)}$**:** Sample $\mathsf{b} = (\mathsf{b}_1, \ldots, \mathsf{b}_\kappa) \leftarrow\!\$\ \{0,1\}^\kappa$ and $(g_{0,1}, \ldots, g_{0,\kappa}), (g_{1,1}, \ldots, g_{1,\kappa}) \leftarrow\!\$\ G^\kappa$. Generate $\iota \leftarrow\!\$\ G$ and set $x_1 \leftarrow \iota \star x_0$. Set the public key $\mathsf{pk} = ((\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa}))$ with $\mathsf{pk}_{0,i} = g_{0,i} \star x_{\mathsf{b}_i}$, $\mathsf{pk}_{1,i} = g_{1,i} \star x_{1-\mathsf{b}_i}$. Output $\mathsf{sk} = \big(\iota, (g_{0,1}, \ldots, g_{0,\kappa}), (g_{1,1}, \ldots, g_{1,\kappa}), \mathsf{b}\big), \mathsf{pk}$.

$\underline{\mathsf{Signer}_1(\mathsf{sk})}$**:** Compute $\mathsf{pk}', \mathbf{t}', \sigma' \leftarrow \mathsf{KeyRand}(\mathsf{pk})$. For $i \in [\kappa]$ set $(g'_{0,i}, g'_{1,i}) = (t'_{0,\sigma'(i)} \cdot g_{0,\sigma'(i)}, t'_{1,\sigma'(i)} \cdot g_{1,\sigma'(i)})$. Sample $\mathbf{h}_0, \mathbf{h}_1 \leftarrow\!\$\ G^\mu$. For each $i \in [\mu]$ compute

$$(y_{0,i}, y_{1,i}) \leftarrow (h_{0,i}, h_{1,i}) \star (x_0, x_1),$$

and apply $\mathsf{rswap}(y_{0,i}, y_{1,i})$. Forward $\mathsf{pk}', \mathbf{y} := \big((y_{0,1}, y_{1,1}), \ldots, (y_{0,\mu}, y_{1,\mu})\big)$ to the user.

$\underline{\mathsf{User}_1(\mathsf{pk}', \mathbf{y}, \mathsf{M})}$**:** Compute $\mathsf{pk}'', \mathbf{t}'', \sigma'' \leftarrow \mathsf{KeyRand}(\mathsf{pk}')$. Sample $\mathbf{r}_0, \mathbf{r}_1 \leftarrow\!\$\ G^\mu$ and $\pi \leftarrow\!\$\ \mathcal{S}_\mu$. Compute

$$(z_{0,i}, z_{1,i}) := \big(r_{0,i} \star y_{0,\pi(i)},\ r_{1,i} \star y_{1,\pi(i)}\big) \quad \forall i \in [\mu]$$

then apply $\mathsf{rswap}(z_{0,i}, z_{1,i})$ and set $\mathbf{z} = ((z_{0,1}, z_{1,1}), \ldots, (z_{0,\mu}, z_{1,\mu}))$. Compute $c \leftarrow \mathsf{H}(\mathbf{z}, \mathsf{pk}'', \mathsf{M})$, set $c' := (\sigma''(c_{\pi^{-1}(1)}), \ldots, \sigma''(c_{\pi^{-1}(\mu)}))$, and send c' to the signer.

$\underline{\mathsf{Signer}_2(\mathsf{sk}, c')}$**:** For each $i \in [\mu]$, using b, compute

$$(s'_{0,i}, s'_{1,i}) \leftarrow \big(h_{\mathsf{b}_{\sigma'(c'_i)}, i} \cdot (g'_{0,\,c'_i})^{-1},\ \ h_{1-\mathsf{b}_{\sigma'(c'_i)}, i} \cdot (g'_{1,\,c'_i})^{-1}\big).$$

Send $\mathbf{s}' := \big((s'_{0,1}, s'_{1,1}), \ldots, (s'_{0,\mu}, s'_{1,\mu})\big), \mathbf{t}'$ and σ' to the user.

$\underline{\mathsf{User}_2(\mathsf{pk}, \mathbf{y}, \pi, \mathbf{r}_0, \mathbf{r}_1, c', \mathbf{s}')}$**:** Verify that pk' was computed using t' and σ' and check, for all $i \in [\mu]$,

$$(s'_{0,i}, s'_{1,i}) \star \big(\mathsf{pk}'_{0,c'_i}, \mathsf{pk}'_{1,c'_i}\big) \ \in\ \big\{(y_{0,i}, y_{1,i}), (y_{1,i}, y_{0,i})\big\}.$$

If any check fails, output $\perp$ and abort. Otherwise, compute $\mathbf{s} = ((s_{0,1}, s_{1,1}), \ldots, (s_{0,\mu}, s_{1,\mu}))$ where

$$(s_{0,i}, s_{1,i}) \leftarrow \big(r_{d_i,i} \cdot s'_{0,\pi(i)} \cdot t''^{-1}_{0,c'_{\pi(i)}},\ r_{1-d_i,i} \cdot s'_{1,\pi(i)} \cdot t''^{-1}_{1,c'_{\pi(i)}}\big).$$

The choice bit d_i is computed as follows: if $(s'_{0,\pi(i)}, s'_{1,\pi(i)}) \star \big(\mathsf{pk}'_{0,c'_{\pi(i)}}, \mathsf{pk}'_{1,c'_{\pi(i)}}\big) = (y_{0,\pi(i)}, y_{1,\pi(i)})$ then set $d_i = 0$, otherwise $d_i = 1$. Output the signature $\mathsf{sig} = \big(\mathbf{z},\ \mathbf{s},\ \mathbf{t},\ \sigma\big)$ where $\mathbf{t} = ((t''_{0,\sigma''(1)} \cdot t'_{0,\sigma'(\sigma''(1))}, \ldots, t''_{0,\sigma''(\kappa)} \cdot t'_{0,\sigma'(\sigma''(\kappa))}), (t''_{1,\sigma''(1)} \cdot t'_{1,\sigma'(\sigma''(1))}, \ldots, t''_{1,\sigma''(\kappa)} \cdot t'_{1,\sigma'(\sigma''(\kappa))}))$ and $\sigma = \sigma' \circ \sigma''$.

$\underline{\mathsf{Verify}(\mathsf{pk}, \mathsf{M}, \mathsf{sig})}$**:** Parse $\mathsf{sig} = (\mathbf{z},\ \mathbf{s},\ \mathbf{t},\ \sigma)$ where $\mathbf{z} = \big((z_{0,1}, z_{1,1}), \ldots, (z_{0,\mu}, z_{1,\mu})\big)$, $\mathbf{s} = \big((s_{0,1}, s_{1,1}), \ldots, (s_{0,\mu}, s_{1,\mu})\big)$ and $\mathbf{t} = ((t_{0,1}, \ldots, t_{0,\kappa}), (t_{1,1}, \ldots, t_{1,\kappa}))$. Set $\overline{\mathsf{pk}} = ((t_{0,1} \star \mathsf{pk}_{0,\sigma(1)}, \ldots, t_{0,\kappa} \star \mathsf{pk}_{0,\sigma(\kappa)}), (t_{1,1} \star \mathsf{pk}_{1,\sigma(1)}, \ldots, t_{1,\kappa} \star \mathsf{pk}_{1,\sigma(\kappa)}))$. Compute $c^\star \leftarrow \mathsf{H}(\mathbf{z}, \overline{\mathsf{pk}}, \mathsf{M})$. Accept iff for all $i \in [\mu]$ the following holds and reject otherwise

$$(s_{0,i}, s_{1,i}) \star \big(\overline{\mathsf{pk}}_{0,c_i}, \overline{\mathsf{pk}}_{1,c_i}\big) \ \in\ \big\{(z_{0,i}, z_{1,i}), (z_{1,i}, z_{0,i})\big\}.$$

Scheme 1: Wombat: Concurrent Secure BS based on the Protocol from Fig. 3.

HVZK. A simulator $\mathsf{ZK.Sim}(\mathsf{pk}, c)$ proceeds as follows:

1. Sample $\mathbf{s} \leftarrow\!\!\$\ G^{2\kappa}$ independently.
2. For each $i \in [\kappa]$, compute $(y_{0,i}, y_{1,i}) := (s_{0,i}, s_{1,i}) \star (\mathsf{pk}_{0,c(i)}, \mathsf{pk}_{1,c(i)})$, then apply rswap to the pair.
3. Output the transcript $(\mathbf{y}, c, \mathbf{s})$.

For any fixed challenge c, the distribution of the simulated transcript matches that of a real execution: the vector $\mathbf{s}$ is uniformly sampled from $G^{2\kappa}$, and for each i, the verification equation holds in one of the two possible orders with equal probability, independent of the secret bits b. Hence, the simulation is perfect.

4 Concurrent BS from Group Action Inverse Problem

Now we are ready to introduce our new blind signature framework basing on Scheme 1. Formally, we require the following ingredients and the subroutine:

- $(x_0, X, G, \star)$ be a cryptographic group action.
- Parameters $\kappa = \kappa(\lambda) \geq 2, \alpha = \alpha(\lambda) \geq 2$ and $\mu = \alpha \cdot \kappa$ be functions in λ.
- $\mathsf{H} : \{0,1\}^* \to [\kappa]^\mu$ be a hash function (modeled as a random oracle) with the following property: for any string M and $i, j \in [\kappa]$ the number of occurrences of i in $\mathsf{H}(\mathsf{M})$ is equal to those of j.
- $\mathsf{KeyRand}(\mathsf{pk})$: on input a public key $\mathsf{pk} = ((\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa}))$ sample a vector $\mathbf{t} = ((t_{0,1}, \ldots, t_{0,\kappa}), (t_{1,1}, \ldots, t_{1,\kappa})) \leftarrow\!\!\$\ G^{2\kappa}$ and permutation $\sigma \leftarrow\!\!\$\ \mathcal{S}_\kappa$, randomize pk by computing $\mathsf{pk}' = ((t_{0,\sigma(1)} \star \mathsf{pk}_{0,\sigma(1)}, \ldots, t_{0,\sigma(\kappa)} \star \mathsf{pk}_{0,\sigma(\kappa)}), (t_{1,\sigma(1)} \star \mathsf{pk}_{1,\sigma(1)}, \ldots, t_{1,\sigma(\kappa)} \star \mathsf{pk}_{1,\sigma(\kappa)}))$ and output $(\mathsf{pk}', \mathbf{t}, \sigma)$.

Here, we let ι to be a part of the sk in replace of x_1, which reduces the secret key size in general.

Theorem 4.1. *The blind signature presented in Scheme 1 is correct.*

Proof. Let $\mathsf{sig} = (\mathbf{z}, \mathbf{s}, \mathbf{t}, \sigma)$ be the output of User_2 and M be the message. We show that

$$(s_{0,i}, s_{1,i}) \star (\overline{\mathsf{pk}}_{0,c_i}, \overline{\mathsf{pk}}_{1,c_i}) \in \{(z_{0,i}, z_{1,i}), (z_{1,i}, z_{0,i})\}$$

where $\overline{\mathsf{pk}} = ((t_{0,1} \star \mathsf{pk}_{0,\sigma(1)}, \ldots, t_{0,\kappa} \star \mathsf{pk}_{0,\sigma(\kappa)}), (t_{1,1} \star \mathsf{pk}_{1,\sigma(1)}, \ldots, t_{1,\kappa} \star \mathsf{pk}_{1,\sigma(\kappa)}))$ and $c = (c_1, \ldots, c_\mu) = \mathsf{H}(\mathbf{z}, \overline{\mathsf{pk}}, \mathsf{M})$ is the challenge. By the definition of $t_{b,i}$ at the end of User_2, we have, for all $i \in [\kappa]$ and $b \in \{0,1\}$,

$$\overline{\mathsf{pk}}_{b,i} = t''_{b,\sigma''(i)} \cdot t'_{b,\sigma'(\sigma''(i))} \star \mathsf{pk}_{b,\sigma(i)} = \mathsf{pk}''_{b,i}.$$

It follows that $\mathsf{H}(\mathbf{z}, \overline{\mathsf{pk}}, \mathsf{M}) = \mathsf{H}(\mathbf{z}, \mathsf{pk}'', \mathsf{M})$. From the definition of $s_{b,i}$ in User_2, we have:

$$\begin{aligned}
s_{0,i} &= r_{d_i,i} \cdot {s'}_{0,\pi(i)} \cdot {t''}^{-1}_{0,c'_{\pi(i)}} \\
&= r_{d_i,i} \cdot h_{\mathsf{b}_{\sigma'(c'_{\pi(i)})},\pi(i)} \cdot {g'}^{-1}_{0,c'_{\pi(i)}} \cdot {t''}^{-1}_{0,c'_{\pi(i)}} \\
&= r_{d_i,i} \cdot h_{\mathsf{b}_{\sigma'(\sigma''(c_i))},\pi(i)} \cdot {g'}^{-1}_{0,\sigma''(c_i)} \cdot {t''}^{-1}_{0,\sigma''(c_i)} \\
&= r_{d_i,i} \cdot h_{\mathsf{b}_{\sigma'(\sigma''(c_i))},\pi(i)} \cdot (t'_{0,\sigma'(\sigma''(c_i))} \cdot g_{0,\sigma'(\sigma''(c_i))})^{-1} \cdot {t''}^{-1}_{0,\sigma''(c_i)} \\
&= r_{d_i,i} \cdot h_{\mathsf{b}_{\sigma'(\sigma''(c_i))},\pi(i)} \cdot ({t''}_{0,\sigma''(c_i)} \cdot t'_{0,\sigma'(\sigma''(c_i))} \cdot g_{0,\sigma'(\sigma''(c_i))})^{-1}
\end{aligned}$$

and similarly

$$s_{1,i} = r_{1-d_i,i} \cdot h_{1-\mathsf{b}_{\sigma'(\sigma''(c_i))},\pi(i)} \cdot ({t''}_{1,\sigma''(c_i)} \cdot t'_{1,\sigma'(\sigma''(c_i))} \cdot g_{1,\sigma'(\sigma''(c_i))})^{-1}.$$

Therefore, we have:

$$(s_{0,i}, s_{1,i}) \star (\overline{\mathsf{pk}}_{0,c_i}, \overline{\mathsf{pk}}_{1,c_i}) = (r_{d_i,i} \star y_{\mathsf{b}_{\sigma'(\sigma''(c_i))},\pi(i)}, r_{1-d_i,i} \star y_{1-\mathsf{b}_{\sigma'(\sigma''(c_i))},\pi(i)})$$

before rswap in Sign_1. By construction of $(z_{0,i}, z_{1,i})$ in User_1, we can conclude that $(s_{0,i}, s_{1,i}) \star (\overline{\mathsf{pk}}_{0,c_i}, \overline{\mathsf{pk}}_{1,c_i}) \in \{(z_{0,i}, z_{1,i}), (z_{1,i}, z_{0,i})\}$ as desired.

Remark 4.2 (Recoverable commitments.). We remark that to simplify the proof in the algebraic group action model, we include the set elements in the signature. In fact, in Scheme 1, we can modify the User_2 algorithm such that the user returns $(c^\star, \mathbf{s}, \mathbf{t}, \sigma)$ as the signature. This modification allows any public verifier to compute $\mathbf{z}$ using $\mathbf{s}, \mathbf{t}, \sigma$ and pk and the user uses lexicographic ordering for $(z_{0,i}, z_{1,i})$ or their hashes to replace the random swapping and skip the trial in the verification. In this case, the verification process merely involves verifying the hash output $c^\star$. This approach results in a more compact signature. We utilize this alternative in our instantiation.

4.1 Security Proofs

We first show the blindness. On high-level, we show that each view out of the two executions for the adversary has the same probability generating the same signature.

Theorem 4.3. *The blind signature presented in Scheme 1 has perfect blindness.*

Proof. We show that any signature can be linked to any transcript of a non-aborting execution of the protocol of the same probability conditioned on the signer's view. In particular, we show that for any pair message/signature M, sig and any transcript $[(\mathsf{pk}', \mathbf{y}, (c'_1, \ldots, c'_\mu), (\mathbf{s}', \mathbf{t}', \sigma')]$ of an execution of the scheme,

there exist permutations π, σ'' and group elements $((r_{0,1}, \ldots, r_{0,\mu}), (r_{1,1}, \ldots, r_{1,\mu}))$ and $((t''_{0,1}, \ldots, t''_{0,\kappa}), (t''_{1,1}, \ldots, t''_{1,\kappa}))$ such that

$$\begin{aligned}\mathsf{sig} =&(((r_{0,\pi(i)} \star y_{0,\pi(i)}, r_{1,\pi(i)} \star y_{1,\pi(i)}))_{i\in[\mu]},\\ &((r_{d_i,\pi(i)} \cdot s'_{0,\pi(i)} \cdot {t''}^{-1}_{0,c'_{\pi(i)}}, r_{1-d_i,\pi(i)} \cdot s'_{1,\pi(i)} \cdot {t''}^{-1}_{1,c'_{\pi(i)}}))_{i\in[\mu]},\\ &((t''_{0,\sigma''(i)} \cdot t'_{0,\sigma'(\sigma''(i))})_{i\in[\kappa]}, (t''_{1,\sigma''(i)} \cdot t'_{1,\sigma'(\sigma''(i))})_{i\in[\kappa]}), \sigma' \circ \sigma'').\end{aligned}$$

Let $\mathsf{sig} = (((z_{0,i}, z_{1,i}))_i, ((s_{0,i}, s_{1,i}))_i, ((t_{0,i})_i, (t_{1,i})_i), \sigma)$ be the signature and $\overline{\mathsf{pk}}$ the corresponding public key. We compute $\sigma'' = (\sigma')^{-1} \circ \sigma$. From the permutations σ'', σ', $((t_{0,1}, \ldots, t_{0,\kappa}), (t_{1,1}, \ldots, t_{1,\kappa}))$ and $((t'_{0,1}, \ldots, t'_{0,\kappa}), (t'_{1,1}, \ldots, t'_{1,\kappa}))$ we compute $((t''_{0,1}, \ldots, t''_{0,\kappa}), (t''_{1,1}, \ldots, t''_{1,\kappa}))$ as follows:

$$t''_{b,i} = t_{b,\sigma''^{-1}(i)} \cdot t'^{-1}_{b,\sigma'(i)}.$$

We compute the challenge

$$(c_1, \ldots, c_\mu) = \mathsf{H}(((z_{0,i}, z_{1,i}))_i, \overline{\mathsf{pk}}, \mathsf{M})$$

and compare it with $(c'_1, \ldots, c'_\mu)$. From the knowledge of σ'' and from the fact that the challenge is balanced, there exists a permutation π such that

$$(c'_1, \ldots, c'_\mu) = (\sigma''(c_{\pi^{-1}(1)}), \ldots, \sigma''(c_{\pi^{-1}(\mu)})).$$

We recover $((r_{0,1}, \ldots, r_{0,\mu}), (r_{1,1}, \ldots, r_{1,\mu}))$ from the following equalities $s_{0,i} = r_{d_i,\pi(i)} \cdot s'_{0,\pi(i)} \cdot {t''}^{-1}_{0,c'_{\pi(i)}}$ and $s_{1,i} = r_{1-d_i,\pi(i)} \cdot s'_{1,\pi(i)} \cdot {t''}^{-1}_{1,c'_{\pi(i)}}$, where the bits d_i can be computed according to the verification of the signer's response. We note that π is not unique: in fact, there are $\alpha!^\kappa$ distinct permutations that work. This is not a problem for the proof since the number of possible permutations is independent of the signature and the transcript.

Theorem 4.4. *Let* BS *be Scheme 1. For any p.p.t. algebraic adversary $\mathcal{A}$, there exists a p.p.t. adversary $\mathcal{B}$ and a negligible function $\varepsilon = \varepsilon(\lambda)$ such that*

$$\mathsf{Adv}^{\mathsf{cOMUF}}_{\mathcal{A},\ell}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{GAIP}}_{\mathcal{B}}(\lambda) + \varepsilon(\lambda).$$

Proof. Let $\mathcal{A}$ be an adversary as in the statement. We assume that $\mathcal{A}$ makes, at most, a polynomial number of queries $Q = Q(\lambda)$. We denote by $(\mathsf{M}_1, \mathsf{sig}_1), \ldots, (\mathsf{M}_{\ell+1}, \mathsf{sig}_{\ell+1})$ the pairs message/signature output by $\mathcal{A}$ on input the public key $\mathsf{pk} = ((\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa}))$. More precisely, for $j \in [\ell + 1]$, $\mathsf{sig}_j = (\mathbf{z}_j, \mathbf{s}_j, \mathbf{t}_j, \sigma_j)$ where:

$$\begin{aligned}&\mathbf{z}_j = ((z^j_{0,1}, z^j_{1,1}), \ldots, (z^j_{0,\mu}, z^j_{1,\mu})),\ \mathbf{s}_j = ((s^j_{0,1}, s^j_{1,1}), \ldots, (s^j_{0,\mu}, s^j_{1,\mu}))\\ &\mathbf{t}_j = ((t^j_{0,1}, \ldots, t^j_{0,\kappa}), (t^j_{1,1}, \ldots, t^j_{1,\kappa}))\end{aligned}$$

Since the adversary is algebraic, it will also describe each set element $z^j_{b,i}$ in terms of the previously seen set elements. That is, for all b, i, j it will give us $\alpha^j_{b,i} \in G$

and a set element $x^j_{b,i} \in X$ seen during its execution, such that $z^j_{b,i} = \alpha^j_{b,i} \star x^j_{b,i}$. We will also assume that all the forgeries output by $\mathcal{A}$ are valid signatures, that is for all i, j

$$(s^j_{0,i}, s^j_{1,i}) \star \left(\overline{\mathsf{pk}}^j_{0,c_{i,j}}, \overline{\mathsf{pk}}^j_{1,c_{i,j}}\right) \ \in \ \left\{(z^j_{0,i}, z^j_{1,i}), (z^j_{1,i}, z^j_{0,i})\right\} \tag{1}$$

where $\overline{\mathsf{pk}}^j = ((t^j_{0,1} \star \mathsf{pk}_{0,\sigma(1)}, \ldots, t^j_{0,\kappa} \star \mathsf{pk}_{0,\sigma(\kappa)}), (t^j_{1,1} \star \mathsf{pk}_{1,\sigma(1)}, \ldots, t^j_{1,\kappa} \star \mathsf{pk}_{1,\sigma(\kappa)}))$ and $(c^j_1, \ldots, c^j_\mu) = \mathsf{H}(\mathbf{z}_j, \overline{\mathsf{pk}}^j, \mathsf{M}_j)$.

For $j \in [\ell + 1]$ we define A_j and B_j as follows. A_j is the set of commitments $(z^j_{0,i}, z^j_{1,i})$ that are *simulated*, that is the triplet of elements (from the signature) $((z^j_{0,i}, z^j_{1,i}), c^j_i, (s^j_{0,i}, s^j_{1,i}))$ is a *simulated* transcript of the underlined sigma protocol, which means, without loss of generality, that $(z^j_{0,i}, z^j_{1,i}) = (s^j_{0,i}, s^j_{1,i}) \star (\overline{\mathsf{pk}}^j_{0,c_{i,j}}, \overline{\mathsf{pk}}^j_{1,c_{i,j}})$. The set A_j is well defined because the adversary is algebraic: a commitment belongs to this set, wlog, if and only if $z^j_{b,i}$ is described by the adversary as $s^j_{b,i} \star \overline{\mathsf{pk}}^j_{b,c_{i,j}}$. We define B_j as the set of commitments $(z^j_{0,i}, z^j_{1,i}) = (\alpha^j_{0,i} \star x^j_{0,i}, \alpha^j_{1,i} \star x^j_{1,i})$ where $x^j_{0,i}$ or $x^j_{1,i}$ is a signer commitment from a closed session (at the moment of the corresponding call to H) with the same challenge public key: namely, c^j_i coincides with the ith challenge forwarded from User_1 to the signer for $x^j_{0,i}$ or $x^j_{1,i}$ for the same session-specific (ephemeral) key. We define $X_j = A_j \cup B_j$. Roughly speaking, for this set of commitments, the adversary is guessing with respect to the hash queries: since those commitments are simulated or come from closed sessions, a valid signature requires the hash output to match a predetermined value.

For $j \in [\ell + 1]$ let us denote with Open_j the set of indices corresponding to the sessions that are open when $\mathcal{A}$ is querying the hash of the j-th forgery. For $k \in \mathsf{Open}_j$ we define the set Ω^k_j as the collection of commitments in $\mathbf{z}_j$ of the form $h \star y$ where y is a signer's commitment of session k, whose corresponding challenge matches the corresponding signer's response of session k (they are both linked to the same public key in the end). Roughly speaking, for this set of commitments, the adversary is guessing with respect to the signer's hidden randomness generating the ephemeral key. For $j \in [\ell + 1]$ we also define the number ω_j as follows:

$$\omega_j := |\{k \in \mathsf{Open}_j \mid \Omega^k_j \neq \emptyset\}| - 1.$$

Now, let us consider the following games.

Game $\mathbf{G_0}$: The concurrent one-more unforgeability experiment $\mathsf{ExpOUnf}(1^\lambda)$.

Game $\mathbf{G_1}$: This is G_0 with the following modification. When the adversary $\mathcal{A}$ outputs the signatures $(\mathsf{M}_1, \mathsf{sig}_1), \ldots, (\mathsf{M}_{\ell+1}, \mathsf{sig}_{\ell+1})$, if there is a j such that $\mathcal{A}$ never queries $\mathsf{H}((\mathbf{z}_j, -, \mathsf{M}_j))$ we emulate a query via lazy sample, otherwise we do nothing. Then, we abort the experiment if at least one of the two conditions is satisfied.

1. There exists $j \in [\ell + 1]$ such that $|X_j| \geq \mu^\delta$.
2. $\sum_{j=1}^{\ell+1} \omega_j \geq \mu^{1-\delta} - 2$

Where δ is a real number in the interval $(0, 1)$.

Roughly speaking, the first abort condition captures that the adversary cannot produce a forgery containing many commitments that are all tied to a predetermined challenge. The intuition behind the second condition is that each session is bound to a specific public key; therefore, commitments originating from different sessions cannot be freely combined within a single forgery.

Lemma 4.5. *Let E be the event "$\mathcal{A}$ aborts in G_1". Then we have*

$$\Pr[E] \leq Q \cdot p_{\mu^\delta} + \left(\frac{1}{\kappa}\right)^{\mu^{1-\delta}-2}.$$

Proof. Let E_1 and E_2 be the events corresponding, respectively, to the first and the second abort condition. For a given commitment $\overrightarrow{\mathsf{com}}$, set elements $\overline{\mathsf{pk}}$, message M and challenges $c_1, \ldots, c_T$ where $\mathsf{H}(\overrightarrow{\mathsf{com}}, \overline{\mathsf{pk}}, \mathsf{M})$ has never been queried before, we have $\mathsf{H}(\overrightarrow{\mathsf{com}}, \overline{\mathsf{pk}}, m)$ is of a given form $(c_1, \ldots, c_T, *, \ldots, *)$ with the probability

$$p_T = \frac{\alpha!^\kappa}{\mu!} \cdot \frac{(\mu - T)!}{\prod_{i=1}^{\kappa}(\alpha - t_i)!} \leq \frac{\alpha!^\kappa}{\mu!} \cdot \frac{(\mu - \kappa\lfloor T/\kappa \rfloor)!}{(\alpha - \lfloor T/\kappa \rfloor)!^\kappa}.$$

where t_j is the number of occurrences of j in the given challenge. It follows that $\Pr[E_1] \leq Q \cdot p_{\mu^\delta}$.

We now estimate $\Pr[E_2]$. Since the attacker doesn't know the k-th signer's permutation (applied to the public key) until they query Sign_2 (and hence close the session), and since the final public key is part of the hash, each session is independent from the others. Moreover, for a given j and $k_0 \in \mathsf{Open}_j$, the challenges corresponding to the Ω_j^k for $k \neq k_0$, must agree with the unknown k-th signer's permutation. It follows that

$$\Pr[E_2] \leq \prod_{j=1}^{\ell+1} \left(\frac{1}{\kappa}\right)^{\omega_j} = \left(\frac{1}{\kappa}\right)^{\sum_{j=1}^{\ell+1}\omega_j} \leq \left(\frac{1}{\kappa}\right)^{\mu^{1-\delta}-2}.$$

We conclude by using the union bound.

We now proceed with the reduction to GAIP. Let $\mathcal{A}$ be a p.p.t. adversary playing G_1. We construct an adversary $\mathcal{B}$ playing $\mathrm{G} = \mathsf{GAIP}$ as follows. Let $\Pi = (x, X, G, \star)$ be as in our G.

1. Let Y be the challenge set element given from G. Set $x_0 = x$ and $x_1 = Y$. Run $\mathsf{KeyGen}(\lambda)$ to obtain pk. In particular, sample $\mathsf{b} = (\mathsf{b}_1, \ldots, \mathsf{b}_\kappa) \leftarrow\!\!\$\ \{0,1\}^\kappa$ and $(g_{0,1}, \ldots, g_{0,\kappa}), (g_{1,1}, \ldots, g_{1,\kappa}) \leftarrow\!\!\$\ G^\kappa$. Then the public key is

$$\mathsf{pk} = ((g_{0,1} \star x_{\mathsf{b}_1}, \ldots, g_{0,\kappa} \star x_{\mathsf{b}_\kappa}), (g_{1,1} \star x_{1-\mathsf{b}_1}, \ldots, g_{1,\kappa} \star x_{1-\mathsf{b}_\kappa})).$$

2. Run $\mathcal{A}$ on input $(x_0, x_1, X, G, \star)$ and pk.
3. Simulate the queries to H in the standard way and the queries to $\mathsf{Sign}_1()$ and $\mathsf{Sign}_2()$ as in the blind signature scheme.
   - j-th Sign_1: compute $\mathsf{pk}'_j, \mathbf{t}'_j, \sigma'_j \leftarrow \mathsf{KeyRand}(\mathsf{pk})$ and set $(g'^j_{0,i}, g'^j_{1,i}) = (t'^j_{0,\sigma'(i)} \cdot g_{0,\sigma'(i)}, t'^j_{1,\sigma'(i)} \cdot g_{1,\sigma'(i)})$. Sample $\mathbf{h}^j_0, \mathbf{h}^j_1 \leftarrow\!\$\, G^\mu$. For each $i \in [\mu]$ compute
   $$(y^j_{0,i}, y^j_{1,i}) \leftarrow (h^j_{0,i}, h^j_{1,i}) \star (x_0, x_1),$$
   then apply $\mathsf{rswap}(y^j_{0,i}, y^j_{1,i})$. Respond with pk'_j and $\mathbf{y}_j := ((y^j_{0,i}, y^j_{1,i}))_{i\in[\mu]}$.
   - j-th Sign_2: on input the challenge $(c'^j_1, \ldots, c'^j_\mu)$, for $i \in [\mu]$, using b_j, compute $(s'^j_{0,i}, s'^j_{1,i}) \leftarrow \left(h^j_{\mathsf{b}_{\sigma'(c'^j_i)},i} \cdot (g'^j_{0,\,c'^j_i})^{-1},\ \ h_{1-\mathsf{b}_{\sigma'(c'^j_i)},i} \cdot (g'^j_{1,\,c'^j_i})^{-1}\right)$.
   Respond with $\mathbf{s}_j = ((s'^j_{0,1}, s'^j_{1,1}), \ldots, (s'^j_{0,\mu}, s'^j_{1,\mu}))$.

   Correctness of the simulation follows from correctness of the scheme.
4. We receive from $\mathcal{A}$ the forgeries $(\mathsf{M}_1, \mathsf{sig}_1), \ldots (\mathsf{M}_{\ell+1}, \mathsf{sig}_{\ell+1})$ where $\mathsf{sig}_j = (\mathbf{z}_j, \mathbf{s}_j, \mathbf{t}_j, \sigma_j)$ with
   $$\begin{aligned}\mathbf{z}_j &= ((z^j_{0,1}, z^j_{1,1}), \ldots, (z^j_{0,\mu}, z^j_{1,\mu}))\\ \mathbf{s}_j &= ((s^j_{0,1}, s^j_{1,1}), \ldots, (s^j_{0,\mu}, s^j_{1,\mu}))\\ t_j &= (t^j_1, \ldots, t^j_\kappa)\end{aligned}$$
   together with $\alpha^j_{b,i} \in G$ and $x^j_{b,i} \in X$ such that $z^j_{b,i} = \alpha^j_{b,i} \star x^j_{b,i}$, where $x^j_{b,i}$ is a set element saw during its execution.
5. We extract $g \in G$ such that $g \star x_0 = x_1$. We note that we can extract the challenge Dlog if one of the following conditions are satisfied:
   (a) There exists b, i, j such that $x^j_{b,i} = x_0$ or $x^j_{b,i} = x_1$. Since the forgeries are valid signatures, from Eq. (1) we have $s^j_{b',i} \star \overline{\mathsf{pk}}^j_{b',c^j_i} = z^j_{b,i} = \alpha^j_{b,i} \star x^j_{b,i}$. Let us assume $x^j_{b,i} = x_0$. By construction, $\overline{\mathsf{pk}}^j_{b',c^j_i} = \gamma \star x_0$ or $\overline{\mathsf{pk}}^j_{b',c^j_i} = \gamma \star x_1$ for some $\gamma \in G$ depending on the choice b. Therefore we can extract $g = (s^j_{b',i} \cdot \gamma)^{-1} \cdot \alpha^j_{b,i}$ if $\overline{\mathsf{pk}}^j_{b',c^j_i} = \gamma \star x_1$. Since the choice of b is random and unknown to $\mathcal{A}$, this happens with probability $1/2$. The situation is the same if $x^j_{b,i} = x_1$.

   (b) There exists b, i, j such that $x^j_{b,i}$ is (a component of) a public key different from $\gamma \star \mathsf{pk}_{0,\sigma(c^j_i)}$ or $\gamma \star \mathsf{pk}_{1,\sigma(c^j_i)}$ for some $\gamma \in G$. In this case, the extraction can be done as the previous one.

   (c) There exists b, i, j such that $x^j_{b,i}$ is a signer's commitment from a session whose corresponding challenge differs from the one asked in the corresponding Sign_2 query. In this case, there exists I, J such that $x^j_{b,i} = y^J_{b',I}$ and $c^j_i \neq c'^J_I$. On one hand (by correctness of the simulation) we have $y^J_{b',I} = s'^J_{b'',I} \star \mathsf{pk}'_{b'',c'^J_I}$, on the other, applying Eq. (1), $y^J_{b',I} = {\alpha^j_{b,i}}^{-1} \star z^j_{b,i} = ({\alpha^j_{b,i}}^{-1} \cdot s^j_{b''',i}) \star \overline{\mathsf{pk}}_{b''',c^j_i}$. We thus obtain the relation $s'^J_{b'',I} \star \mathsf{pk}'_{b'',c'^J_I} = ({\alpha^j_{b,i}}^{-1} \cdot s^j_{b''',i}) \star \overline{\mathsf{pk}}_{b''',c^j_i}$ from which we can extract if

at the right-hand side we have x_0 and x_1 at the left-hand side (or vice versa). Since b is random and unknown to $\mathcal{A}$, and rswap is applied, this happens with probability $1/2$.

We show that at least one among the three above conditions must hold. Let us assume by contradiction that (a), (b), and (c) do not hold. We show that it contradicts the abort conditions in G_1. Since the adversary is algebraic and the above three conditions do not hold, then each forgery is made with only commitments from the sets X_j and Ω_j^i. It follows that

$$\sum_{j=1}^{\ell+1} |X_j| + \sum_{i,j} |\Omega_j^i| = (\ell+1)\cdot\mu.$$

By the first abort condition, we have $|X_j| < \mu^\delta$. Moreover, since each of the ℓ open session comes with μ commitments, we have $\sum_{i,j} |\Omega_j^i| \leq \ell \cdot \mu$. Therefore, we must have that $\ell > \mu^{1-\delta} - 1 := \ell_0$. By the definition, ω_j counts the number of distinct open sessions that contribute at least one commitment to the j-th forgery minus 1. Since there must be at least ℓ_0 open sessions in total and the commitments of the forgeries belong only to X_j and Ω_j^i, it follows that $\sum_j^{\ell+1} \omega_j > \ell_0 - 1 = \mu^{1-\delta} - 2$, which contradicts the second abort condition.

We have shown that from the forgeries returned by $\mathcal{A}$ we can extract g such that $Y = x_1 = g \star x_0 = g \star x$ with probability $1/2$ due to witness indistinguishability (Proposition 2.9). It follows that

$$\mathsf{Adv}^{\mathsf{GAIP}}_{\mathcal{B}}(\lambda) = 1/2 \cdot \mathsf{Adv}^{\mathrm{G}_1}_{\mathcal{A}}(\lambda) \text{ and therefore,}$$

$$\mathsf{Adv}^{\mathsf{cOMUF}}_{\mathcal{A},\ell}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{GAIP}}_{\mathcal{B}}(\lambda) + \Pr[E] \leq 2 \cdot \mathsf{Adv}^{\mathsf{GAIP}}_{\mathcal{B}}(\lambda) + Q \cdot p_{\mu^\delta} + \left(\frac{1}{\kappa}\right)^{\mu^{1-\delta}-2}.$$

In conclusion, for $\delta = 1/2$ we have

$$\mathsf{Adv}^{\mathsf{cOMUF}}_{\mathcal{A},\ell}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{GAIP}}_{\mathcal{B}}(\lambda) + \varepsilon$$

where $\varepsilon = Q \cdot p_{\mu^{1/2}} + \left(\frac{1}{\kappa}\right)^{\mu^{1/2}-2}$ is negligible.

Remark 4.6 (Comparison with Tanuki's result). In [HLM+25, Thm. 10], it is shown that their framework is concurrently secure with the following bound on the adversary's advantage:

$$\mathsf{Adv}^{\mathsf{cOMUF}}_{\mathcal{A},\ell}(\lambda) \leq \mathsf{Adv}^{\mathsf{OMVP}}_{\mathcal{B}}(\lambda) + \varepsilon,$$

where $\varepsilon = Q\binom{\mu}{\alpha,\ldots,\alpha}^{-1} + (\ell+1)\frac{(\kappa/2)!}{\kappa!} + Q^2 p_{\mu/2}\frac{(\kappa/2)!}{\kappa!}$ and for $T \in [\mu]$, $p_T = \frac{\alpha!^\kappa}{\mu!} \cdot \frac{(\mu-\kappa\lfloor T/\kappa\rfloor)!}{(\alpha-\lfloor T/\kappa\rfloor)!^\kappa}$. Looking at ε, we note that the term $(\ell+1)\frac{(\kappa/2)!}{\kappa!}$ only depends on κ (and not μ). Therefore, ε is negligible only if $\kappa = \mathsf{poly}(\lambda)$ is a polynomial

function in λ, making the bound applicable only to the special case $\kappa = \mu$ and $\kappa = \mu/2$. Our result, Theorem 4.4, gives the following bound instead:

$$\mathsf{Adv}^{\mathsf{cOMUF}}_{\mathcal{A},\ell}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{GAIP}}_{\mathcal{B}}(\lambda) + \varepsilon$$

where $\varepsilon = Q \cdot p_{\mu^{1/2}} + \left(\frac{1}{\kappa}\right)^{\mu^{1/2}-2}$, which is negligible for $\mu = \mathsf{poly}(\lambda)$, without imposing any restrictions κ.

In conclusion, this proof is more general and allows for greater flexibility in the choice of κ and μ.

5 Instantiations

We consider the following two post-quantum group actions for our instantiations from isogenies and codes respectively. Due to the distinct sampling algorithm feature, the applicability of modulo lattice isomorphism [DPPvW22] problem remains an interesting open problem.

CSIDH/CSI-FiSh. CSI-FiSh [BKV19], improved upon the isogeny-based key exchange CSIDH [CLM+18], uses the action of the ideal class group G of the quadratic order $\mathbb{Z}[\sqrt{-p}]$ on the set X of supersingular elliptic curves defined over a prime field $\mathbb{F}_p$. To overcome the scalability issue (i.e. the cryptographic group action considered in our work) has been an active line of research [DFK+23, CLP24, DEF+25]. The elements of X can be represented as an element in $\mathbb{F}_p$ and the group element G can be represented as an element in $\mathbb{Z}/N\mathbb{Z}$ where $N \approx \sqrt{p}$. For our instantiation, we take the CSIFiSh parameter $p = 512$ such that it takes 64 and 32 B to represent a set element and a group element respectively.

LESS. LESS [BMPS20], a code-based signature scheme submitted to the NIST call, is based on the hardness of the *linear equivalence problem*. This problem naturally fits the framework of cryptographic group actions. The set X consists of k-dimensional linear codes of length n over $\mathbb{F}_q$; each code can be represented by a generator matrix in row reduced echelon form $\mathsf{RREF}(\cdot)$ in $\mathbb{F}_q^{k\times n}$. The group G is taken to be the monomial group $\mathcal{M}_n$, consisting of matrices that are the product of a permutation matrix and a non-singular diagonal matrix. The group action is defined as: $\star\colon G \times X \to X, \quad (M, x) \mapsto \mathsf{RREF}(xM)$.

In LESS, public keys, elements in X, are chosen to be in systematic form, enabling storage as the $k \times (n-k)$ suffix of the generator matrix. For the parameter set $(n, k, q) = (252, 126, 127)$ recommended in [BBB+24] for the NIST-1 security level. Hence, it takes 13.6 KB and 472.5 B to represent a set element and a group element respectively.

5.1 Improvement on the Code-based Instantiation

In this subsection, we compress the signature size of our code-based instantiations. For reference, a direct LESS-based instantiation is given in Scheme 3

and Scheme 4, covering $\{\mathsf{KeyGen}, \mathsf{Signer}\}$ and $\{\mathsf{User}, \mathsf{Verify}\}$ respectively. Our improved construction is in Scheme 2; the modification is entirely on the user side, while key generation and the signer remain unchanged. We also show that the same technique applies to Tanuki's LESS instantiations; see Appendix B.

To compress the signature size of the instantiations, we use one of the canonical forms introduced in [CPS23]. Roughly speaking, their work establishes the class group and the new equivalence classes of the codes accordingly with efficient mapping in between the codes and their canonical representatives. We summarize the basic propositions as follows.

Background.

Proposition 5.1 ([CPS23]). *Let $F = \mathcal{M}_k \oplus \mathcal{M}_{n-k} \leq \mathcal{M}_n$ be the subgroup of monomial matrices acting on the rows and columns of the non-systematic part of a generator matrix. Then there exists a canonical form function CF (implicitly parameterized by F) such that, on input a k-dimensional systematic code, outputs either another k-dimensional systematic code or $\perp$. Concretely, $\mathsf{CF} : \mathbb{F}^{k\times(n-k)} \to \mathbb{F}^{k\times(n-k)} \cup \{\perp\}$, and the following hold:*

1. *CF runs in time polynomial in n and $\log_2(q)$;*
2. *The codes generated by $(\mathbf{I}_k \mid \mathbf{A})$ and $(\mathbf{I}_k \mid \mathbf{A}')$ are F-equivalent if and only if there exist $\mathbf{M}_1 \in \mathcal{M}_k$ and $\mathbf{M}_2 \in \mathcal{M}_{n-k}$ such that $\mathbf{A} = \mathbf{M}_1\mathbf{A}'\mathbf{M}_2$;*
3. *For any $\mathbf{A}$ with $\mathsf{CF}(\mathbf{A}) \neq \perp$, the codes generated by $(\mathbf{I}_k \mid \mathsf{CF}(\mathbf{A}))$ and $(\mathbf{I}_k \mid \mathbf{A})$ are F-equivalent;*
4. *For any two F-equivalent codes generated by $(\mathbf{I}_k \mid \mathbf{A})$ and $(\mathbf{I}_k \mid \mathbf{A}')$, we have $\mathsf{CF}(\mathbf{A}) = \mathsf{CF}(\mathbf{A}')$.*

For convenience, we may abuse notation and write $\mathsf{CF}(\mathbf{G}) := (\mathbf{I}_k \mid \mathsf{CF}(\mathbf{A}))$ for a systematic generator matrix $\mathbf{G} = (\mathbf{I}_k \mid \mathbf{A})$ whenever $\mathsf{CF}(\mathbf{A})$ is defined, and $\mathsf{CF}(\mathbf{G}) := \perp$ otherwise. In fact, several choices for the subgroup F are possible [CPS23]. Following [BBB+24], we set $F = \mathcal{M}_k \oplus \mathcal{M}_{n-k}$, which guarantees that the probability of CF outputting $\perp$ is negligible. With this choice, we obtain the following result.

Proposition 5.2 (Thm 8, [CPS23]). *There exists an efficient compression function $\mathsf{CosetRep} : \mathcal{M}_n \to S_{k,n}$ mapping any $\mu \in \mathcal{M}_n$ to the unique ordered size-k subset $J \subseteq \{1, \ldots, n\}$ such that μ admits a decomposition $\mu = \phi \circ \chi$ with $\phi \in F$ and $\chi \in S_{k,n}$, where J records the positions mapped to the first k coordinates by χ.*

Moreover, $\mathsf{CosetRep}$ is constant on cosets: for any $\mu_0, \mu_1 \in \mathcal{M}_n$ with $\mu_0 F = \mu_1 F$, we have $\mathsf{CosetRep}(\mu_0) = \mathsf{CosetRep}(\mu_1)$.

Hence, one can represent the output of $\mathsf{CosetRep}$ as an n-bit string representing choosing an entry or not. For our case where $n = 2k$, the encoding is almost optimal compared to $\log_2 \binom{n}{k}$.

Subroutines. We first introduce RREF^*, an extension of the standard RREF. Given a generator matrix $\mathbf{G}$, the deterministic algorithm RREF^* outputs a systematic form $\mathbf{G}' \in \mathbb{F}_q^{k\times n}$ together with a permutation ρ satisfying $\mathbf{G}' = \mathsf{RREF}(\mathbf{G}\rho)$. We also introduce CF^*, which is the same as CF, except that it additionally provides a monomial map μ explaining how the input code $(\mathbf{I}|\mathbf{A})$ is mapped to the code generated $(\mathbf{I}|\mathsf{CF}(\mathbf{A}))$. While RREF^* and CF^* look similar at first glance, they serve different roles in our instantiation.

A plain call to RREF on a random generator matrix returns a *systematic* form with a good chance, in which case the pivot set is $\{1,\ldots,k\}$. However, with non-negligible probability the row-reduced form is *non-systematic*. A naïve fix is to abort and resample, which is discouraged in blind-signature designs (see the issue pointed out in [HKLN20]). Instead, we introduce RREF^* which enable the user to *adapt* the randomness so the output is always systematic. This halves the description cost of a map: we move from specifying two index sets to specifying just one.

In the concurrent flow, the signer first randomizes pk to pk' (for concurrent security), and the user then randomizes pk' to pk'' (for blindness). The signer reveals the randomness in Signer_2, after which the user must provide (i) a map from pk to pk'' and (ii) a map from pk'' to the commitment codes com. To compress these maps, both pk'' and com must be presented in canonical form. The subtlety is that, even if the user knows the concrete monomial maps $\mathsf{pk}\to\mathsf{pk}''\to\mathsf{com}$, the *compressed* map of the latter one is not the one between the canonical forms of pk and com, because F-equivalence is not preserved under any monomial maps. We therefore introduce CF^*, which records the *translation monomial* sending a code to its canonical representative, which enables the user to compute the "calibrated" compressed map between pk'' and com. Both RREF^* and CF^* require adjusting randomness; we prove these adjustments preserve blindness.

Formally, we introduce three following subroutines in our improvement.

- $\underline{\mathsf{RREF}^*(\mathbf{G})}$: On input a full-rank generator matrix $\mathbf{G} \in \mathbb{F}_q^{k\times n}$, return a pair $(\mathbf{G}', \rho)$ with a permutation $\rho \in S_n$ and $\mathbf{G}' \in \mathbb{F}_q^{k\times n}$ such that
$$\mathbf{G}' \;=\; \mathsf{RREF}(\mathbf{G}\,\rho) \;=\; (\mathbf{I} \mid \mathbf{A}^\star)$$
for some $\mathbf{A}^\star \in \mathbb{F}_q^{k\times(n-k)}$.
- $\underline{\mathsf{CF}^*(\mathbf{A})}$: On input a $k\times(n-k)$ matrix $\mathbf{A}$ or say a systematic generator matrix $(\mathbf{I} \mid \mathbf{A})$, return $\mathsf{CF}(\mathbf{A})$ and a monomial $\mu \in F \subseteq \mathcal{M}_n$ such that
$$\mathsf{RREF}\big((\mathbf{I} \mid \mathbf{A})\mu\big) = (\mathbf{I} \mid \mathsf{CF}(\mathbf{A})).$$
- $\underline{\mathsf{KeyRand}^*(\mathsf{pk})}$: Given $\mathsf{pk} = (\mathsf{pk}_1,\ldots,\mathsf{pk}_\kappa)$, sample $\mathbf{T}_1,\ldots,\mathbf{T}_\kappa \leftarrow\!\!\$\; G$ and a permutation $\sigma \leftarrow\!\!\$\; S_\kappa$. For each $i \in [\kappa]$, set
$$\mathsf{pk}'_i, \rho_i \leftarrow \mathsf{RREF}^*(\mathsf{pk}_{\sigma(i)}\mathbf{T}_{\sigma(i)}),$$
update $\mathbf{T}_{\sigma(i)} \leftarrow \rho_i \circ \mathbf{T}_{\sigma(i)}$, and then set $\mathsf{pk}'_i, \mu \leftarrow \mathsf{CF}^*(\mathsf{pk}'_i)$ and update $\mathbf{T}_{\sigma(i)} \leftarrow \mu \circ \mathbf{T}_{\sigma(i)}$. Output $(\mathsf{pk}', (\mathbf{T}_1,\ldots,\mathbf{T}_\kappa), \sigma)$, where $\mathsf{pk}' = (\mathsf{pk}'_1,\ldots,\mathsf{pk}'_\kappa)$.

Remark 5.3. CF^* is computationally feasible in our setting. Indeed, one can apply the polynomial-time reduction from Proposition 5.7 for F-equivalence to recover the corresponding monomial map. A more efficient method is to trace the execution of CF directly. For the case $F = \mathcal{M}_k \oplus \mathcal{M}_{n-k}$, the computation consists of a sequence of column- and row-sorting and scaling operations [CPS23], which are straightforward to track. In this setting, the augmented version CF^* introduces less than 1% overhead compared to CF in our experiment.

Security. To show the improvement does not compromise security, we require the following results.

Lemma 5.4 (Lemma 1, [LP25]). *Let $\mathcal{C}_0, \mathcal{C}_1$ be two linear codes where $\mathcal{C}_0$ is generated by $\mathbf{G}_0$ and $\mathcal{C}_1$ is generated by a systematic matrix $\mathbf{G}_1 = (\mathbf{I}_k \mid \mathbf{A}_1)$. Suppose $\mu \in \mathcal{M}_n$ satisfies $\mu(\mathcal{C}_0) = \mathcal{C}_1$, and let $J \leftarrow \mathsf{CosetRep}(\mu)$ Then we have*

$$\mathsf{CF}\left(\left((\mathbf{G}_0)_J\right)^{-1} \cdot (\mathbf{G}_0)_{\bar{J}}\right) = \mathsf{CF}(\mathbf{A}_1).$$

Proof. Since $\mu(\mathcal{C}_0) = \mathcal{C}_1$, there exists $\mathbf{S} \in \mathrm{GL}_k(q)$ and a monomial matrix $Q \in \mathcal{M}_n$ such that $\mathbf{G}_1 = \mathbf{S} \cdot \mathbf{G}_0 \cdot \mathbf{Q}$. Recall that $J \leftarrow \mathsf{CosetRep}(\mu)$ represents the size-k subset corresponding to the information set mapped to the first k coordinates by $\mathbf{Q}$. Hence, $(\mathbf{G}_0)_J$ is invertible. By following the decomposition through J and the standard linear algebra, we have

$$\mathbf{G}_1 = (\,\mathbf{I}_k \mid \mathbf{A}_1\,) = \mathbf{S} \cdot (\,(\mathbf{G}_0)_J \mid (\mathbf{G}_0)_{\bar{J}}\,) \cdot \begin{pmatrix} \mathbf{Q}'_r & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}'_c \end{pmatrix}.$$

Thus, the non-systematic part of $\mathbf{G}_1$ can be expressed as

$$\mathbf{A}_1 = \mathbf{Q}'^{-1}_r \cdot (\mathbf{G}_0)_J^{-1} (\mathbf{G}_0)_{\bar{J}} \cdot \mathbf{Q}'_c,$$

where $\mathbf{Q}'^{-1}_r \in \mathcal{M}_k$ and $\mathbf{Q}'_c \in \mathcal{M}_{n-k}$. By Proposition 5.1, we have

$$\mathsf{CF}\left((\mathbf{G}_0)_J^{-1} (\mathbf{G}_0)_{\bar{J}}\right) = \mathsf{CF}(\mathbf{A}_1).$$

Theorem 5.5 (Theorem 3, [LP25]). *For any three $\mathcal{M}_n$-equivalent codes generated by $\mathbf{G}$, $\mathbf{G}_0 = (\mathbf{I}_k \mid \mathbf{A}_0)$ and $\mathbf{G}_1 = (\mathbf{I}_k \mid \mathbf{A}_1)$ where $\mathsf{CF}(\mathbf{G}_0) = \mathbf{G}_0$ and $\mathsf{CF}(\mathbf{G}_1) = \mathbf{G}_1$, the following two sets have the same cardinality:*

$$S_0 = \left\{\mathbf{Q} \in \mathcal{M}_n : \mathsf{CF}(\mathbf{G}') = \mathbf{G}_0 \text{ where } (\mathbf{G}', \rho) \leftarrow \mathsf{RREF}^*(\mathbf{GQ})\right\}$$
$$S_1 = \left\{\mathbf{Q} \in \mathcal{M}_n : \mathsf{CF}(\mathbf{G}') = \mathbf{G}_1 \text{ where } (\mathbf{G}', \rho) \leftarrow \mathsf{RREF}^*(\mathbf{GQ})\right\}$$

Proof. Due to the $\mathcal{M}_n$-equivalence, there exists $\mathbf{M} \in \mathcal{M}_n$ such that $\mathbf{G}_1 = \mathsf{RREF}(\mathbf{G}_0\mathbf{M})$. For each $\mathbf{Q} \in S_1$ together with the corresponding $(\mathbf{G}', \rho) \leftarrow \mathsf{RREF}^*(\mathbf{GQ})$, we claim that the mapping $\mathbf{Q} \mapsto \mathbf{Q}\rho\mathbf{M}\rho^{-1}$ is a bijection onto S_1. First, the monomial $\mathbf{Q}\rho\mathbf{M}\rho^{-1}$ is in S_1 because

$$\mathsf{RREF}(\mathbf{G}(\mathbf{Q}\rho\mathbf{M}\rho^{-1})\rho, \rho) \leftarrow \mathsf{RREF}^*(\mathbf{Q}\rho\mathbf{M}\rho^{-1}).$$

Since $\mathsf{RREF}(\mathbf{GQ}\rho) = \mathbf{G}'$, and $\mathbf{GQ}\rho$ is F-equivalent to $\mathbf{G}_0$. Hence,

$$\mathbf{GQ}\rho\mathbf{M} = \mathbf{G}(\mathbf{Q}\rho\mathbf{M}\rho^{-1})\rho$$

is F-equivalent to $\mathbf{G}_0\mathbf{M}$.

Given $\mathbf{Q}_1 \in S_1$ with ρ be the corresponding auxiliary outputs via $\mathsf{RREF}^*(\mathbf{GQ}_1)$, take $\mathbf{Q}_0 = \mathbf{Q}_1\rho\mathbf{M}^{-1}\rho^{-1}$. Since $\mathsf{RREF}(\mathbf{G}_1\mathbf{M}^{-1})$ is systematic, ρ is the auxiliary outputs of $\mathsf{RREF}^*(\mathbf{GQ}_0)$ such that $\mathsf{RREF}(\mathbf{GQ}_1\rho\mathbf{M}^{-1})$ is systematic. Hence, $\mathbf{Q}_0 \in S_1$, and due to the finiteness of S_0, S_1 the mapping is bijection.

We can have an immediate corollary as follows.

Corollary 5.6. *Let $\mathbf{G}_0$ and $\mathbf{G}_1$ be two generator matrices of $\mathcal{M}_n$-equivalent codes. Define the following two random variables:*

$$\begin{aligned} Y_0 &:= (\mathsf{CF}(\mathbf{G}_0'), \mathbf{Q}\rho_0) \quad \textit{where } \mathbf{Q} \leftarrow\!\!\$\ \mathcal{M}_n,\ (\mathbf{G}_0', \rho_0) \leftarrow \mathsf{RREF}^*(\mathbf{G}_0\mathbf{Q}); \\ Y_1 &:= (\mathsf{CF}(\mathbf{G}_1'), \mathbf{Q}\rho_1) \quad \textit{where } \mathbf{Q} \leftarrow\!\!\$\ \mathcal{M}_n,\ (\mathbf{G}_1', \rho_1) \leftarrow \mathsf{RREF}^*(\mathbf{G}_1\mathbf{Q}). \end{aligned}$$

Then Y_0 and Y_1 are identically distributed as random variables over $\mathbb{F}_q^{k\times n} \times \mathcal{M}_n$.

Proposition 5.7 (Prop 10, [CPS23]). *If $(\mathcal{C}, \mathcal{C}')$ admits a solution for CF-LEP, then a solution for LEP on input $\mathcal{C}, \mathcal{C}'$ can be found in polynomial time.*

Concretely, say the two codes are generated by matrices $\mathbf{G}, \mathbf{G}' \in \mathbb{F}_q^{k\times n}$. If there exist size-$k$ sets $J, J' \subseteq \{1, \ldots, n\}$ such that

$$\mathsf{CF}\left(\mathbf{G}_J^{-1} \cdot \mathbf{G}_{\bar{J}}\right) = \mathsf{CF}\left((\mathbf{G}')_{J'}^{-1} \cdot (\mathbf{G}')_{\bar{J}'}\right),$$

then one can efficiently find a monomial matrix $\mu \in \mathcal{M}_n$ such that $\mathcal{C}' = \mu(\mathcal{C})$.

Let $G = \mathcal{M}_n$ be the group of monomial matrices. Fix two RREF generator matrices $\mathbf{X}_0, \mathbf{X}_1 \in \mathbb{F}_q^{k\times n}$. Let κ, α, μ and $\mathsf{H} : \{0,1\}^* \to [\kappa]^\mu$ be defined the same as in Scheme 1.

$\underline{\mathsf{User}_1(\mathsf{pk}', \mathbf{y}, \mathsf{M})}$**:** Compute $\mathsf{pk}'', \mathbf{t}'', \sigma'' \leftarrow \mathsf{KeyRand}^*(\mathsf{pk}')$.
Sample $(\mathbf{r}_{0,1}, \ldots, \mathbf{r}_{0,\mu}), (\mathbf{r}_{1,1}, \ldots, \mathbf{r}_{1,\mu}) \leftarrow\!\!\$\ G^\mu$ and $\pi \leftarrow\!\!\$\ S_\mu$. For $i \in [\mu]$ compute

$$\mathbf{Z}_{0,i}, \rho_0 \leftarrow \mathsf{RREF}^*(\mathbf{Y}_{0,\pi(i)}\,\mathbf{r}_{0,i}), \qquad \mathbf{Z}_{1,i}, \rho_1 \leftarrow \mathsf{RREF}^*(\mathbf{Y}_{1,\pi(i)}\,\mathbf{r}_{1,i}),$$

$$\mathbf{Z}_{0,i} \leftarrow (\mathbf{I}|\mathsf{CF}(\mathbf{Z}_{0,i})), \qquad \mathbf{Z}_{1,i} \leftarrow (\mathbf{I}|\mathsf{CF}(\mathbf{Z}_{1,i})),$$

$$\mathbf{r}_{0,i}\cdot\rho_0 \leftarrow \mathbf{r}_{0,i}, \qquad \mathbf{r}_{1,i}\cdot\rho_1 \leftarrow \mathbf{r}_{1,i},$$

then apply $\mathsf{rswap}(\mathbf{Z}_{0,i}, \mathbf{Z}_{1,i})$. Let $\mathbf{z} = ((\mathbf{Z}_{0,1}, \mathbf{Z}_{1,1}), \ldots, (\mathbf{Z}_{0,\mu}, \mathbf{Z}_{1,\mu}))$. Compute $c \leftarrow \mathsf{H}(\mathbf{z}, \mathsf{pk}'', \mathsf{M})$ and send $c' = (\sigma''(c_{\pi^{-1}(1)}), \ldots, \sigma''(c_{\pi^{-1}(\mu)}))$.

$\underline{\mathsf{User}_2(\mathsf{pk}, \mathbf{y}, \pi, \mathbf{r}_0, \mathbf{r}_1, c', \mathbf{s}')}$**:** Verify $\mathsf{KeyRand}(\mathsf{pk}; \mathbf{t}', \sigma') \stackrel{?}{=} \mathsf{pk}'$. For all $i \in [\mu]$,

$$\big(\mathsf{RREF}(\mathsf{pk}'_{0,c'_i}\,\mathbf{s}'_{0,i}),\ \mathsf{RREF}(\mathsf{pk}'_{1,c'_i}\,\mathbf{s}'_{1,i})\big)\ \in\ \{(\mathbf{Y}_{0,i}, \mathbf{Y}_{1,i}), (\mathbf{Y}_{1,i}, \mathbf{Y}_{0,i})\};$$

abort if not. Let $d_i \in \{0,1\}$ be the bit such that

$$\big(\mathsf{RREF}(\mathsf{pk}'_{0,c'_{\pi(i)}}\,\mathbf{s}'_{0,\pi(i)}),\ \mathsf{RREF}(\mathsf{pk}'_{1,c'_{\pi(i)}}\,\mathbf{s}'_{1,\pi(i)})\big) = (\mathbf{Y}_{d_i,\pi(i)},\ \mathbf{Y}_{1-d_i,\pi(i)}).$$

Set $T = \Big(\big(\,\mathsf{CosetRep}(\mathbf{t}''_{0,\sigma''(i)}\,\mathbf{t}'_{0,\sigma'(\sigma''(i))})\,\big)_{i=1}^{\kappa},\ \big(\,\mathsf{CosetRep}(\mathbf{t}''_{1,\sigma''(i)}\,\mathbf{t}'_{1,\sigma'(\sigma''(i))})\,\big)_{i=1}^{\kappa}\Big)$. For each $i \in [\mu]$ set

$$S_{0,i} = \mathsf{CosetRep}((\mathbf{t}''_{0,c'_{\pi(i)}})^{-1}\,\mathbf{s}'_{0,\pi(i)}\,\mathbf{r}_{d_i,i}),\ \ S_{1,i} = \mathsf{CosetRep}((\mathbf{t}''_{1,c'_{\pi(i)}})^{-1}\,\mathbf{s}'_{1,\pi(i)}\,\mathbf{r}_{1-d_i,i}).$$

Output $\mathsf{sig} = (\mathbf{z}, S, T, \sigma)$ with $S = ((S_{0,1}, S_{1,1}), \ldots, (S_{0,\mu}, S_{1,\mu}))$, $\sigma = \sigma' \circ \sigma''$.

$\underline{\mathsf{Verify}(\mathsf{pk}, \mathsf{M}, \mathsf{sig})}$**:** Parse $\mathsf{sig} = (\mathbf{z}, S, T, \sigma)$ with $\mathbf{z} = ((\mathbf{Z}_{0,1}, \mathbf{Z}_{1,1}), \ldots, (\mathbf{Z}_{0,\mu}, \mathbf{Z}_{1,\mu}))$, $S = ((S_{0,1}, S_{1,1}), \ldots, (S_{0,\mu}, S_{1,\mu}))$ and $T = ((T_{0,1}, \ldots, T_{0,\kappa}), (T_{1,1}, \ldots, T_{1,\kappa}))$. Form

$$\overline{\mathsf{pk}} = \Big(\big(\,(\mathbf{I}|\mathsf{CF}((\mathsf{pk}_{0,\sigma(i)})^{-1}_{T_{0,i}}(\mathsf{pk}_{0,\sigma(i)})_{\bar{T}_{0,i}})\big)_{i=1}^{\kappa},\ \big(\,(\mathbf{I}|\mathsf{CF}((\mathsf{pk}_{1,\sigma(i)})^{-1}_{T_{1,i}}(\mathsf{pk}_{1,\sigma(i)})_{\bar{T}_{1,i}})\big)_{i=1}^{\kappa}\Big).$$

Compute $c^\star \leftarrow \mathsf{H}(\mathbf{z}, \overline{\mathsf{pk}}, \mathsf{M})$ and accept iff for all $i \in [\mu]$,

$$\Big((\mathbf{I}|(\overline{\mathsf{pk}}_{0,c_i})^{-1}_{S_{0,i}}(\overline{\mathsf{pk}}_{0,c_i})_{\bar{S}_{0,i}}), (\mathbf{I}|(\overline{\mathsf{pk}}_{1,c_i})^{-1}_{S_{1,i}}(\overline{\mathsf{pk}}_{1,c_i})_{\bar{S}_{1,i}})\Big)\ \in\ \{(\mathbf{Z}_{0,i}, \mathbf{Z}_{1,i}), (\mathbf{Z}_{1,i}, \mathbf{Z}_{0,i})\};$$

Scheme 2: The improvement upon the LESS instantiation for User and Verify.

Now we are ready to prove the security of our improved scheme.

Theorem 5.8 (Blindness). *The blind signature {Schemes 2 and 3} has blindness under malicious key.*

Proof. To show the blindness, we have to show that for arbitrary two transcripts

$$\mathsf{sig} = (\overline{\mathsf{pk}}, \mathbf{z}, (c_1, \ldots, c_\mu), S, T, \sigma),\ \ \mathsf{sig}' = (\overline{\mathsf{pk}}', \mathbf{z}, (c'_1, \ldots, c'_\mu), S, T, \sigma'),$$

corresponding to Signer's first messages

$$((\widetilde{\mathsf{pk}}, (\mathbf{Y}_{0,1}, \mathbf{Y}_{1,1}), \ldots, (\mathbf{Y}_{0,\mu}, \mathbf{Y}_{1,\mu})), (\widetilde{\mathsf{pk}}', ((\mathbf{Y}'_{0,1}, \mathbf{Y}'_{1,1}), \ldots, (\mathbf{Y}'_{0,\mu}, \mathbf{Y}'_{1,\mu}))).$$

Here while $\overline{\mathsf{pk}}, \overline{\mathsf{pk}}', c, c'$ are not in the signature, they will be determined by the signatures. We include them in the transcripts for clarity in explanation.

The adversary cannot distinguish the order of the executions with advantage.

For the session specific key $\widetilde{\mathsf{pk}}$, by Corollary 5.6, the probabilities of resulting in $\overline{\mathsf{pk}}, \overline{\mathsf{pk}}'$ are the same regardless of the update made in CF^* in $\mathsf{KeyRand}^*$. The same applies for $\widetilde{\mathsf{pk}}'$.

Under the malicious key setting, though the generator matrices are not necessarily $\mathcal{M}_n$-equivalent, the $\mathcal{M}_n$-equivalence between entries within $\mathbf{z}_i$, $\mathbf{z}'_{i'}$ and $\overline{pk}_{c_i}$ can be guaranteed when $c_i = c'_{i'}$ by the validness of the signatures. Hence, this ensures the existence of the user's randomness such that the signer's first message $((\mathbf{Y}_{0,1}, \mathbf{Y}_{1,1}), \ldots, (\mathbf{Y}_{0,\mu}, \mathbf{Y}_{1,\mu}))$ leads to sig'. For any first message $\tilde{C}$ of signer, we have $\Pr_\pi[\pi(\tilde{C}) = (c_1, \ldots, c_\mu)] = \Pr_\pi[\pi(\tilde{C}) = (c'_1, \ldots, c'_\mu)]$ where π is drawn uniformly at random.

Without loss of generality, it suffices to show that the probability generating $\mathsf{sig}, \mathsf{sig}'$ are the same. This is a direct implication of Corollary 5.6, and we thereby have perfect blindness.

Theorem 5.9. *The blind signature {Schemes 2 and 3} is ℓ-OMUF secure if the code-based blind signature directly instantiated from Scheme 1 is ℓ-OMUF secure. The reduction is polynomial time in λ without loss in advantage.*

Proof. Suppose there exists an adversary $\mathcal{A}$ that breaks the OMUF security of {Schemes 2 and 3}, which we call the improved scheme. We construct an adversary $\mathcal{B}$ that breaks the OMUF security of {Schemes 2 and 3}, called the original scheme, as follows:

1. $\mathcal{B}$ initiates the OMUF experiment for the original scheme with the challenger $\mathcal{C}$, receiving $\mathbf{G}$ and pk.
2. $\mathcal{B}$ runs $\mathcal{A}$ on input $(\mathbf{G}, \mathsf{pk})$.
3. To simulate the signing oracle, whenever $\mathcal{A}$ submits a signing query, $\mathcal{B}$ forwards it to $\mathcal{C}$ and relays the resulting message Y to $\mathcal{A}$.
4. Upon receiving a string c' from $\mathcal{A}$ as User_1's message, $\mathcal{B}$ forwards c' to $\mathcal{C}$ and returns the response $\mathbf{s}'$ to $\mathcal{A}$.

After $\mathcal{A}$ outputs signatures $\{(\mathbf{z}, c_i = (c_{i,1}, \ldots, c_{i,\mu}), S_i, T_i, \sigma_i, m_i)\}_{i\in[\ell+1]}$, where

$$S_i = ((S_{i,0,1}, S_{i,1,1}), \ldots, (S_{i,0,\mu}, S_{i,1,\mu})), \quad T_i = ((T_{i,0,1}, \ldots, T_{i,0,\kappa}), (T_{i,1,1}, \ldots, T_{i,1,\kappa}))$$

$\mathcal{B}$ produces valid forgeries as follows:

1. For each $i \in [\ell+1]$, $b \in \{0,1\}$ and $j \in [\kappa]$,
$$\overline{\mathsf{pk}}_{i,b,j} = \left(\mathbf{I} \mid \mathsf{CF}\left((\mathsf{pk}_{b,\sigma_i(j)})^{-1}_{T_{i,b,j}}(\mathsf{pk}_{b,\sigma_i(j)})_{\bar{T}_{i,b,j}}\right)\right).$$
2. For each $i \in [\ell+1], b \in \{0,1\}$ and each $j \in [\mu]$, apply Proposition 5.7 to solve the linear equivalence problem between $\overline{\mathsf{pk}}_{i,b,c_{i,j}}$ and $(\mathbf{I} \mid \mathbf{Z}_{i,b,j})$ using S, obtaining monomials $\hat{\mathbf{Q}}_{i.b,j}$.

3. For each $i \in [\ell+1], b \in \{0,1\}$ and $j \in [\kappa]$, again, apply Proposition 5.7 to solve the linear equivalence problem between $\overline{\mathsf{pk}}_{i,b,j}$ and $\mathsf{pk}_{b,\sigma_i(j)}$, using T, obtaining monomials $\hat{\mathbf{T}}_{i,b,\tau_i(j)}$.
4. Return the set

$$\left\{(c_i, (\hat{\mathbf{Q}}_{i,b,j}, \hat{\mathbf{T}}_{i,b,j}), m_i)\right\}_{i,b,j \in [\ell+1]\times\{0,1\}\times[\mu]}$$

as forgeries.

The use of Proposition 5.7 gives the corresponding relations needed in the verification the original scheme, and hence gives valid forgeries.

5.2 Performance

	|sk|	|pk|	|sig|	S.OMUF	C.OMUF	Assumption	Genre
Tanuki-3-LESS-(70,70)	2.16 KB	936 KB	64.7 KB	✓	poly(λ)	OM-GAIP	Codes
Tanuki-3-LESS-(54,108)	1.66 KB	719 KB	74.9 KB	✓	poly(λ)	OM-GAIP	Codes
Tanuki-3-CSIDH-(70,70)	2.16 KB	4.4 KB	4.50 KB	✓	poly(λ)	OM-GAIP	Isogenies
Tanuki-3-CSIDH-(54,108)	1.66 KB	3.31 KB	5.18 KB	✓	poly(λ)	OM-GAIP	Isogenies
CSI-Otter [KLLQ23]	16 B	64 B	12* KB	✓	log(λ)	GAIP	Isogenies
CSI-Otter-ζ_4 [KLLQ23]	16 B	512 B	6* KB	✓	log(λ)	ζ_4-rGAIP	Isogenies
HKLN20 [HKLN20]	4.28 MB	444 KB	7.8 MB	N/A	7	RSIS	Lattices
BlindOR [AHJ21]	3.19 KB	34.3 KB	893 KB	N/A	log(λ)	DMLWE&MSIS	Lattices
LEAF-(70,70)	2.16 KB	936 KB	4.42 KB	✓	poly(λ)	OM-GAIP	Codes
LEAF-(54,108)	1.66 KB	719 KB	5.10 KB	✓	poly(λ)	OM-GAIP	Codes
Wombat-LESS-(70,70)	4.41 KB	1.86 MB	8.84 KB	✓	poly(λ)	GAIP	Codes
Wombat-LESS-(49,147)	3.09 KB	1.30 MB	12.3 KB	✓	poly(λ)	GAIP	Codes
Wombat-LESS-(39,195)	2.47 KB	1.03 MB	14.7 KB	✓	poly(λ)	GAIP	Codes
Wombat-CSIDH-(70,70)	4.41 KB	4.38 KB	8.89 KB	✓	poly(λ)	GAIP	Isogenies
Wombat-CSIDH-(49,147)	3.09 KB	3.06 KB	12.54 KB	✓	poly(λ)	GAIP	Isogenies
Wombat-CSIDH-(39,195)	2.47 KB	2.44 KB	14.96 KB	✓	poly(λ)	GAIP	Isogenies

Table 1. Comparison of key and signature sizes of different sigma-protocol-based post-quantum schemes at the 128-bit security level (i.e. $\lambda = 128$) with our Wombat framework shown in Schemes 1 and 2. The columns "S.OMUF & C.OMUF" indicate the sequential security and concurrent security respectively. The former notion means OMUF security by allowing only one execution at the same time, while the latter denotes OMUF allowing multiple concurrent signing sessions. For the performance of CSI-Otter [KLLQ23], we scale their number of repetitions by 1.5 because the original number does not offer sufficient sequential security as mentioned in [DHP24, KLR24]. We name our improved scheme of Tanuki-3-LESS as LEAF; see Appendix A or [LP25] for more details. We give the numbers of the LESS instantiation using the one optimized in Scheme 2.

Size and Security Comparison. Table 1 summarizes key sizes, signature sizes, provable security guarantees, and underlying assumptions for our constructions and the sigma-protocol–based post-quantum blind-signature proposals.

Metric	Our schemes					
	$(\kappa, \mu) = (70, 70)$		$(\kappa, \mu) = (49, 147)$		$(\kappa, \mu) = (39, 195)$	
	CSIDH (s, est.)	LESS (ms, est.)	CSIDH (s, est.)	LESS (ms, est.)	CSIDH (s, est.)	LESS (ms, est.)
KeyGen	3.71	25.8	2.60	18.0	2.07	14.4
Sign	22.60	197.76	31.64	276.94	37.77	330.62
Verify	7.02	95.20	9.83	133.28	11.73	159.12

Table 2. The runtime estimation of our blind signature instantiations for CSIDH-512 (CSIFiSh) (in seconds) and LESS (in milliseconds) group actions on an Intel Core i7-12700H 2.30 GHz.

Compared to CSI-Otter [KLLQ23], both under the same isogeny GAIP assumption, our CSIDH-based instantiation not only achieves smaller signature size by roughly 30% but also provides poly-concurrent OMUF security. Our code-based instantiations, together with the signature-compression technique gives signatures that are approximately 7.2× smaller and rest on the standard code-based assumption compared to Tanuki-LESS. We also show our compression technique applies to Tanuki-LESS, offering a reduction by a factor of 14.5 without compromising the security.

While the public key sizes of code-based instantiations remain relatively large (1-2 MB), they are still fitting well for many blind-signature applications such as anonymous tokens and digital coins, where the key does not always need to be transmitted or stored together with each signature [Mic24, BIS23, App21]. In such settings, the public key can be initialized once and deployed locally at the endpoint, stored in cache and updated occasionally or managed centrally by a certificate authority. In these cases, users only are required to reference the key without specifying it.

Runtime estimation.

We estimate runtime by counting group-action evaluations, which dominate all costs. Concretely, key generation, signing, and verification require $2\kappa, 6(\kappa + \mu), 2(\kappa + \mu)$ group actions, respectively.

By taking the experiment in [HLM+25], the isogeny and the code-based instantiations take 26.4 and 0.19 ms for the group action respectively on an Intel Core i7-12700H 2.30 GHz. For the code-based instantiation the canonical-form and our variant $(\mathsf{CF}, \mathsf{CF}^*)$ computation used for compression is the main extra cost: a canonical-form computation costs on the order of 0.7× a monomial-map evaluation in our experiment in Appendix A. Accounting for all the factors gives the runtime estimates reported for our isogeny and code-based instantiations in Table 2.

Acknowledgments

Yi-Fu Lai is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA -

390781972. Lucjan Hanzlik and Eugenio Paracucchi were funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 545066907.

References

ADMP20. Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 411–439. Springer, Cham, December 2020.

AEB20. Nabil Alkeilani Alkadri, Rachid El Bansarkhani, and Johannes Buchmann. On lattice-based interactive protocols: An approach with less or no aborts. In Joseph K. Liu and Hui Cui, editors, *ACISP 20*, volume 12248 of *LNCS*, pages 41–61. Springer, Cham, November / December 2020.

AHJ21. Nabil Alkeilani Alkadri, Patrick Harasser, and Christian Janson. BlindOR: an efficient lattice-based blind signature scheme from OR-proofs. In Mauro Conti, Marc Stevens, and Stephan Krenn, editors, *CANS 21*, volume 13099 of *LNCS*, pages 95–115. Springer, Cham, December 2021.

AKSY22. Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. Practical, round-optimal lattice-based blind signatures. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 39–53. ACM Press, November 2022.

AO00. Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 271–286. Springer, Berlin, Heidelberg, August 2000.

AOS02. Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n signatures from a variety of keys. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 415–432. Springer, Berlin, Heidelberg, December 2002.

App21. Apple Inc. icloud private relay overview. https://www.apple.com/icloud/docs/iCloud_Private_Relay_Overview_Dec2021.pdf, 2021. Accessed: 2025-08-05.

App24. Apple Inc. Private cloud compute: A new frontier for ai privacy in the cloud. https://security.apple.com/blog/private-cloud-compute/, 2024. Accessed: 2025-08-05.

BBB+23. Marco Baldi, Alessandro Barenghi, Luke Beckwith, Jean-François Biasse, Andre Esser, Kris Gaj, Kamyar Mohajerani, Gerardo Pelosi, Edoardo Persichetti, Markku-Juhani O. Saarinen, Paolo Santini, and Robert Wallace. LESS — Linear Equivalence Signature Scheme. Technical report, National Institute of Standards and Technology, 2023. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.

BBB+24. Marco Baldi, Alessandro Barenghi, Luke Beckwith, Jean-François Biasse, Tung Chou, Andre Esser, Kris Gaj, Patrick Karl, Kamyar Mohajerani, Gerardo Pelosi, Edoardo Persichetti, Markku-Juhani O. Saarinen, Paolo Santini, Robert Wallace, and Floyd Zweydinger. LESS — Linear Equivalence Signature Scheme. Technical report, National Institute of Standards and Technology, 2024. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures.

BBCK24. Benjamin Benčina, Alessandro Budroni, Jesús-Javier Chi-Domínguez, and Mukul Kulkarni. Properties of lattice isomorphism as a cryptographic

group action. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part I*, pages 170–201. Springer, Cham, June 2024.

BBD+25. Michele Battagliola, Giacomo Borin, Giovanni Di Crescenzo, Alessio Meneghetti, and Edoardo Persichetti. Enhancing threshold group action signature schemes: Adaptive security and scalability improvements. Cryptology ePrint Archive, Report 2025/085, 2025.

BBMP24. Michele Battagliola, Giacomo Borin, Alessio Meneghetti, and Edoardo Persichetti. Cutting the GRASS: Threshold GRoup action signature schemes. In Elisabeth Oswald, editor, *CT-RSA 2024*, volume 14643 of *LNCS*, pages 460–489. Springer, Cham, May 2024.

BBN+22. Alessandro Barenghi, Jean-Francois Biasse, Tran Ngo, Edoardo Persichetti, and Paolo Santini. Advanced signature functionalities from the code equivalence problem. Cryptology ePrint Archive, Report 2022/710, 2022.

BCD+24. Alessandro Budroni, Jesús-Javier Chi-Domínguez, Giuseppe D'Alconzo, Antonio J. Di Scala, and Mukul Kulkarni. Don't use it twice! Solving relaxed linear equivalence problems. In *ASIACRYPT 2024, Part VIII*, LNCS, pages 35–65. Springer, Singapore, December 2024.

BCF25. Alessandro Budroni, Jesús-Javier Chi-Domínguez, and Ermes Franch. Don't use it twice: Reloaded! On the lattice isomorphism group action. Cryptology ePrint Archive, Report 2025/516, 2025.

BDK+22. Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 95–126. Springer, Cham, May / June 2022.

BDK+23. Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: generic, simple, and efficient. *DCC*, 91(6):2141–2200, 2023.

BDN+23. Markus Bläser, Dung Hoang Duong, Anand Kumar Narayanan, Thomas Plantard, Youming Qiao, Arnaud Sipasseuth, and Gang Tang. ALTEQ. Technical report, National Institute of Standards and Technology, 2023. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.

BEP+25. Luke Beckwith, Andre Esser, Edoardo Persichetti, Paolo Santini, and Floyd Zweydinger. LESS is even more: Optimizing digital signatures from code equivalence. Cryptology ePrint Archive, Paper 2025/1424, 2025.

BFM+25. Charles Bouillaguet, Thibauld Feneuil, Jules Maire, Matthieu Rivain, Julia Sauvage, and Damien Vergnaud. Blinding post-quantum hash-and-sign signatures. Cryptology ePrint Archive, Paper 2025/895, 2025.

BGZ23. Dan Boneh, Jiaxin Guan, and Mark Zhandry. A lower bound on the length of signatures based on group actions and generic isogenies. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 507–531. Springer, Cham, April 2023.

BIS23. BIS Innovation Lab. Project tourbillon - exploring privacy, security and scalability for cbdcs - final report. https://www.bis.org/publ/othp80.pdf, 2023.

BKP20. Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices.

In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 464–492. Springer, Cham, December 2020.

BKV19. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Cham, December 2019.

BLNS23. Ward Beullens, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Lattice-based blind signatures: Short, efficient, and round-optimal. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM CCS 2023*, pages 16–29. ACM Press, November 2023.

BLS19. Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 176–202. Springer, Cham, August 2019.

BMPS20. Jean-François Biasse, Giacomo Micheli, Edoardo Persichetti, and Paolo Santini. LESS is more: Code-based signatures without syndromes. In Abderrahmane Nitaj and Amr M. Youssef, editors, *AFRICACRYPT 20*, volume 12174 of *LNCS*, pages 45–65. Springer, Cham, July 2020.

Bol03. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46. Springer, Berlin, Heidelberg, January 2003.

BPS+23. Giacomo Borin, Edoardo Persichetti, Paolo Santini, Federico Pintore, and Krijn Reijnders. A guide to the design of digital signatures based on cryptographic group actions. Cryptology ePrint Archive, Report 2023/718, 2023.

BS24. Katharina Boudgoust and Mark Simkin. The power of NAPs: - compressing OR-proofs via collision-resistant hashing. In *TCC 2024, Part I*, LNCS, pages 35–66. Springer, Cham, November 2024.

CAHL+22. Rutchathon Chairattana-Apirom, Lucjan Hanzlik, Julian Loss, Anna Lysyanskaya, and Benedikt Wagner. PI-cut-choo and friends: Compact blind signatures via parallel instance cut-and-choose and more. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 3–31. Springer, Cham, August 2022.

CDS94. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 174–187. Springer, Berlin, Heidelberg, August 1994.

Cha82. David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982.

CLM+18. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Cham, December 2018.

CLP24. Mingjie Chen, Antonin Leroux, and Lorenz Panny. SCALLOP-HD: Group action from 2-dimensional isogenies. In Qiang Tang and Vanessa Teague, editors, *PKC 2024, Part II*, volume 14603 of *LNCS*, pages 190–216. Springer, Cham, April 2024.

CNP+23. Tung Chou, Ruben Niederhagen, Edoardo Persichetti, Lars Ran, Tovohery Hajatiana Randrianarisoa, Krijn Reijnders, Simona Samardjiska, and Monika Trimoska. MEDS — Matrix Equivalence Digital Signature. Technical report, National Institute of Standards and Technology, 2023. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.

CPS23. Tung Chou, Edoardo Persichetti, and Paolo Santini. On linear equivalence, canonical forms, and digital signatures. Cryptology ePrint Archive, Report 2023/1533, 2023.

DEF+25. Pierrick Dartois, Jonathan Komada Eriksen, Tako Boris Fouotsa, Arthur Herlédan Le Merdy, Riccardo Invernizzi, Damien Robert, Ryan Rueger, Frederik Vercauteren, and Benjamin Wesolowski. PEGASIS: Practical effective class group action using 4-dimensional isogenies. In *CRYPTO 2025, Part I*, LNCS, pages 67–99. Springer, Cham, August 2025.

DFK+23. Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 345–375. Springer, Cham, May 2023.

DGS+18. Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *PoPETs*, 2018(3):164–180, July 2018.

DHK+23. Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Generic models for group actions. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 406–435. Springer, Cham, May 2023.

DHP24. Khue Do, Lucjan Hanzlik, and Eugenio Paracucchi. M&M'S: Mix and match attacks on Schnorr-type blind signatures with repetition. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part VI*, volume 14656 of *LNCS*, pages 363–387. Springer, Cham, May 2024.

DM20. Luca De Feo and Michael Meyer. Threshold schemes from isogeny assumptions. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 187–212. Springer, Cham, May 2020.

dPK22. Rafaël del Pino and Shuichi Katsumata. A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 306–336. Springer, Cham, August 2022.

DPPvW22. Léo Ducas, Eamonn W. Postlethwaite, Ludo N. Pulles, and Wessel P. J. van Woerden. Hawk: Module LIP makes lattice signatures fast, compact and simple. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 65–94. Springer, Cham, December 2022.

FHKS16. Georg Fuchsbauer, Christian Hanser, Chethan Kamath, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model from weaker assumptions. In Vassilis Zikas and Roberto De Prisco, editors, *SCN 16*, volume 9841 of *LNCS*, pages 391–408. Springer, Cham, August / September 2016.

Fis06. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 60–77. Springer, Berlin, Heidelberg, August 2006.

FKL18. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Cham, August 2018.

HAB+17. Ethan Heilman, Leen Alshenibr, Foteini Baldimtsi, Alessandra Scafuro, and Sharon Goldberg. TumbleBit: An untrusted Bitcoin-compatible anonymous payment hub. In *NDSS 2017*. The Internet Society, February / March 2017.

HKL19. Eduard Hauck, Eike Kiltz, and Julian Loss. A modular treatment of blind signatures from identification schemes. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 345–375. Springer, Cham, May 2019.

HKLN20. Eduard Hauck, Eike Kiltz, Julian Loss, and Ngoc Khanh Nguyen. Lattice-based blind signatures, revisited. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 500–529. Springer, Cham, August 2020.

HLM+25. Lucjan Hanzlik, Yi-Fu Lai, Marzio Mula, Eugenio Paracucchi, Daniel Slamanig, and Gang Tang. Tanuki: New frameworks for (concurrently secure) blind signatures from post-quantum group actions. In *ASIACRYPT 2025, Part IV*, LNCS, pages 35–69. Springer, Singapore, December 2025.

HLW23. Lucjan Hanzlik, Julian Loss, and Benedikt Wagner. Rai-choo! Evolving blind signatures to the next level. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 753–783. Springer, Cham, April 2023.

KLLQ23. Shuichi Katsumata, Yi-Fu Lai, Jason T. LeGrow, and Ling Qin. CSI-Otter: Isogeny-based (partially) blind signatures from the class group action with a twist. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 729–761. Springer, Cham, August 2023.

KLLQ24. Shuichi Katsumata, Yi-Fu Lai, Jason T. LeGrow, and Ling Qin. CSI-otter: isogeny-based (partially) blind signatures from the class group action with a twist. *DCC*, 92(11):3587–3643, 2024.

KLR24. Shuichi Katsumata, Yi-Fu Lai, and Michael Reichle. Breaking parallel ROS: Implication for isogeny and lattice-based blind signatures. In Qiang Tang and Vanessa Teague, editors, *PKC 2024, Part I*, volume 14601 of *LNCS*, pages 319–351. Springer, Cham, April 2024.

KLX22. Julia Kastner, Julian Loss, and Jiayu Xu. The Abe-Okamoto partially blind signature scheme revisited. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 279–309. Springer, Cham, December 2022.

Lai24. Yi-Fu Lai. Capybara and tsubaki: Verifiable random functions from group actions and isogenies. *CiC*, 1(3):1, 2024.

LNP22. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 71–101. Springer, Cham, August 2022.

LP25. Yi-Fu Lai and Edoardo Persichetti. LEAF: Compact and efficient blind signature from code-based assumptions. Cryptology ePrint Archive, Paper 2025/1585, 2025.

Mic24. Microsoft. U-prove. https://www.microsoft.com/en-us/research/project/u-prove/, 2024. Accessed: 2025-08-05.

MR24. Jonas Meers and Doreen Riepel. CCA secure updatable encryption from non-mappable group actions. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part I*, pages 137–169. Springer, Cham, June 2024.

PS00. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000.

Supplementary Material

A LESS Instantiation related

Here we translate Scheme 1 from the language of group actions into the language of the linear equivalence for reference. Given the length of the description, we separate it into the signer side and the user sides with the verification.

A.1 Direction Instantiation

Let q be a prime, $G = \mathcal{M}_n$ be the group of $n \times n$ monomial matrices over $\mathbb{F}_q$. Fix two RREF generator matrices $\mathbf{X}_0, \mathbf{X}_1 \in \mathbb{F}_q^{k \times n}$. Let $\kappa = \kappa(\lambda) \geq 2$, $\alpha = \alpha(\lambda) \geq 2$, and $\mu = \alpha\kappa$. Let $\mathsf{H} : \{0,1\}^* \to [\kappa]^\mu$ be balanced.

$\underline{\mathsf{KeyGen}(\lambda)}$**:** Sample $\mathsf{b} = (\mathsf{b}_1, \ldots, \mathsf{b}_\kappa) \leftarrow\!\!\$\ \{0,1\}^\kappa$ and $(\mathbf{g}_{0,1}, \ldots, \mathbf{g}_{0,\kappa}), (\mathbf{g}_{1,1}, \ldots, \mathbf{g}_{1,\kappa}) \leftarrow\!\!\$\ G^\kappa$. Set $\mathsf{pk} = ((\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,\kappa}), (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,\kappa}))$ with $\mathsf{pk}_{0,i} = \mathsf{RREF}(\mathbf{X}_{\mathsf{b}_i}\, \mathbf{g}_{0,i})$ and $\mathsf{pk}_{1,i} = \mathsf{RREF}(\mathbf{X}_{1-\mathsf{b}_i}\, \mathbf{g}_{1,i})$. Output $\mathsf{sk} = ((\mathbf{g}_{0,1}, \ldots, \mathbf{g}_{0,\kappa}), (\mathbf{g}_{1,1}, \ldots, \mathbf{g}_{1,\kappa}), \mathsf{b}),\ \mathsf{pk}$.

$\underline{\mathsf{Signer}_1(\mathsf{sk})}$**:** Compute $\mathsf{pk}', \mathbf{t}', \sigma' \leftarrow \mathsf{KeyRand}(\mathsf{pk})$. For each $i \in [\kappa]$ compute $(\mathbf{g}'_{0,i}, \mathbf{g}'_{1,i}) \leftarrow (\mathbf{g}_{0,\sigma'(i)}\, \mathbf{t}'_{0,\sigma'(i)},\ \mathbf{g}_{1,\sigma'(i)}\, \mathbf{t}'_{1,\sigma'(i)})$. Sample $(\mathbf{h}_{0,1}, \ldots, \mathbf{h}_{0,\mu}), (\mathbf{h}_{1,1}, \ldots, \mathbf{h}_{1,\mu}) \leftarrow\!\!\$\ G^\mu$ and set, for $i \in [\mu]$,

$$\mathbf{Y}_{0,i} = \mathsf{RREF}(\mathbf{X}_0\, \mathbf{h}_{0,i}), \qquad \mathbf{Y}_{1,i} = \mathsf{RREF}(\mathbf{X}_1\, \mathbf{h}_{1,i}),$$

then apply $\mathsf{rswap}(\mathbf{Y}_{0,i}, \mathbf{Y}_{1,i})$. Send pk' and $\mathbf{y} = ((\mathbf{Y}_{0,1}, \mathbf{Y}_{1,1}), \ldots, (\mathbf{Y}_{0,\mu}, \mathbf{Y}_{1,\mu}))$ to the user.

$\underline{\mathsf{Signer}_2(\mathsf{sk}, c')}$**:** For each $i \in [\mu]$, using b, compute

$$\mathbf{s}'_{0,i} = (\mathbf{g}'_{0,\,c'_i})^{-1}\, \mathbf{h}_{\mathsf{b}_{\sigma'(c'_i)},\, i}, \qquad \mathbf{s}'_{1,i} = (\mathbf{g}'_{1,\,c'_i})^{-1}\, \mathbf{h}_{1-\mathsf{b}_{\sigma'(c'_i)},\, i}.$$

Send $\mathbf{s}' = ((\mathbf{s}'_{0,1}, \mathbf{s}'_{1,1}), \ldots, (\mathbf{s}'_{0,\mu}, \mathbf{s}'_{1,\mu}))$, together with $\mathbf{t}'$ and σ'.

Scheme 3: A direct instantiation of Scheme 1 using the Linear Equivalence Problem for Signer, where monomial matrices act on generator matrices by right multiplication.

Let q be a prime, $G = \mathcal{M}_n$ be the group of $n \times n$ monomial matrices over $\mathbb{F}_q$. Fix two RREF generator matrices $\mathbf{X}_0, \mathbf{X}_1 \in \mathbb{F}_q^{k \times n}$. Let $\kappa = \kappa(\lambda) \geq 2$, $\alpha = \alpha(\lambda) \geq 2$, and $\mu = \alpha\kappa$. Let $\mathsf{H} : \{0,1\}^* \to [\kappa]^\mu$ be balanced.

$\underline{\mathsf{User}_1(\mathsf{pk}', \mathbf{y}, \mathsf{M})\textbf{:}}$ Compute $\mathsf{pk}'', \mathbf{t}'', \sigma'' \leftarrow \mathsf{KeyRand}(\mathsf{pk}')$. Sample $(\mathbf{r}_{0,1}, \ldots, \mathbf{r}_{0,\mu}), (\mathbf{r}_{1,1}, \ldots, \mathbf{r}_{1,\mu}) \leftarrow\!\!\$\; G^\mu$ and $\pi \leftarrow\!\!\$\; S_\mu$. For $i \in [\mu]$ compute

$$\mathbf{Z}_{0,i} = \mathsf{RREF}(\mathbf{Y}_{0,\pi(i)}\, \mathbf{r}_{0,i}), \qquad \mathbf{Z}_{1,i} = \mathsf{RREF}(\mathbf{Y}_{1,\pi(i)}\, \mathbf{r}_{1,i}),$$

then apply $\mathsf{rswap}(\mathbf{Z}_{0,i}, \mathbf{Z}_{1,i})$. Let $\mathbf{z} = ((\mathbf{Z}_{0,1}, \mathbf{Z}_{1,1}), \ldots, (\mathbf{Z}_{0,\mu}, \mathbf{Z}_{1,\mu}))$. Compute $c \leftarrow \mathsf{H}(\mathbf{z}, \mathsf{pk}'', \mathsf{M})$ and send $c' = (\sigma''(c_{\pi^{-1}(1)}), \ldots, \sigma''(c_{\pi^{-1}(\mu)}))$.

$\underline{\mathsf{User}_2(\mathsf{pk}, \mathbf{y}, \pi, \mathbf{r}_0, \mathbf{r}_1, c', \mathbf{s}')\textbf{:}}$ Verify $\mathsf{KeyRand}(\mathsf{pk}; \mathbf{t}', \sigma') \stackrel{?}{=} \mathsf{pk}'$. For all $i \in [\mu]$,

$$\big(\mathsf{RREF}(\mathsf{pk}'_{0,c'_i}\, \mathbf{s}'_{0,i}),\ \mathsf{RREF}(\mathsf{pk}'_{1,c'_i}\, \mathbf{s}'_{1,i})\big) \ \in\ \{(\mathbf{Y}_{0,i}, \mathbf{Y}_{1,i}), (\mathbf{Y}_{1,i}, \mathbf{Y}_{0,i})\};$$

abort if not. Let $d_i \in \{0,1\}$ be the bit such that

$$\big(\mathsf{RREF}(\mathsf{pk}'_{0,c'_{\pi(i)}}\, \mathbf{s}'_{0,\pi(i)}),\ \mathsf{RREF}(\mathsf{pk}'_{1,c'_{\pi(i)}}\, \mathbf{s}'_{1,\pi(i)})\big) = \ (\mathbf{Y}_{d_i,\pi(i)},\ \mathbf{Y}_{1-d_i,\pi(i)}).$$

For each $i \in [\mu]$ set $\mathbf{s}_{0,i} = (\mathbf{t}''_{0,c'_{\pi(i)}})^{-1}\, \mathbf{s}'_{0,\pi(i)}\, \mathbf{r}_{d_i,i}$, $\mathbf{s}_{1,i} = (\mathbf{t}''_{1,c'_{\pi(i)}})^{-1}\, \mathbf{s}'_{1,\pi(i)}\, \mathbf{r}_{1-d_i,i}$. Output $\mathsf{sig} = (\mathbf{z},\ \mathbf{s},\ \mathbf{t},\ \sigma)$ with $\mathbf{s} = ((\mathbf{s}_{0,1}, \mathbf{s}_{1,1}), \ldots, (\mathbf{s}_{0,\mu}, \mathbf{s}_{1,\mu}))$, $\sigma = \sigma' \circ \sigma''$, and

$$\mathbf{t} = \Big(\big(\,\mathbf{t}''_{0,\sigma''(i)}\, \mathbf{t}'_{0,\sigma'(\sigma''(i))}\,\big)_{i=1}^{\kappa},\ \big(\,\mathbf{t}''_{1,\sigma''(i)}\, \mathbf{t}'_{1,\sigma'(\sigma''(i))}\,\big)_{i=1}^{\kappa}\Big).$$

$\underline{\mathsf{Verify}(\mathsf{pk}, \mathsf{M}, \mathsf{sig})\textbf{:}}$ Parse $\mathsf{sig} = (\mathbf{z}, \mathbf{s}, \mathbf{t}, \sigma)$ with $\mathbf{z} = ((\mathbf{Z}_{0,1}, \mathbf{Z}_{1,1}), \ldots, (\mathbf{Z}_{0,\mu}, \mathbf{Z}_{1,\mu}))$, $\mathbf{s} = ((\mathbf{s}_{0,1}, \mathbf{s}_{1,1}), \ldots, (\mathbf{s}_{0,\mu}, \mathbf{s}_{1,\mu}))$ and $\mathbf{t} = ((\mathbf{t}_{0,1}, \ldots, \mathbf{t}_{0,\kappa}), (\mathbf{t}_{1,1}, \ldots, \mathbf{t}_{1,\kappa}))$. Form

$$\overline{\mathsf{pk}} = \Big(\big(\,\mathsf{RREF}(\mathsf{pk}_{0,\sigma(i)}\, \mathbf{t}_{0,i})\,\big)_{i=1}^{\kappa},\ \big(\,\mathsf{RREF}(\mathsf{pk}_{1,\sigma(i)}\, \mathbf{t}_{1,i})\,\big)_{i=1}^{\kappa}\Big).$$

Compute $c^\star \leftarrow \mathsf{H}(\mathbf{z}, \overline{\mathsf{pk}}, \mathsf{M})$ and accept iff for all $i \in [\mu]$,

$$\big(\mathsf{RREF}(\overline{\mathsf{pk}}_{0,c_i}\, \mathbf{s}_{0,i}),\ \mathsf{RREF}(\overline{\mathsf{pk}}_{1,c_i}\, \mathbf{s}_{1,i})\big) \ \in\ \{(\mathbf{Z}_{0,i}, \mathbf{Z}_{1,i}), (\mathbf{Z}_{1,i}, \mathbf{Z}_{0,i})\};$$

Scheme 4: A direct instantiation of Scheme 1 using the Linear Equivalence Problem for User and Verify, where monomial matrices act on generator matrices by right multiplication. We may assume the value of d_i in User_2 is unique. Otherwise, the event can be used to break the GAIP problem.

Scheme	$(\kappa = 54,\ \mu = 108)$			$(\kappa = 70,\ \mu = 70)$		
	KeyGen	Sign	Verify	KeyGen	Sign	Verify
Runtime in ms						
Tanuki-3-LESS	9.49	94.8	32.8	12.7	81.0	28.0
Scheme 5	10.3	117	54.8	12.7	99.2	47.5
Runtime in Mcycles						
Tanuki-3-LESS	26.7	252	87.6	35.1	218	75.6
Scheme 5	26.7	316	147	33.9	268	127
Slowdown Factor in Cycles (Scheme 5 / Tanuki)						
	-	1.25×	1.7×	-	1.22×	1.69×

Table 3. Comparison of Tanuki-3-LESS and our Scheme 5 runtimes on an Intel Core i7-12700H @ 2.30 GHz using the same flags on average of 1000 executions respectively. The slowdown ratio shows how many times slower Scheme 5 is compared to LESS. As the key generation algorithms are identical, it is pointless to compare the costs. One can expect a smaller slowdown factor given the faster CF implementation [BEP+25].

B Improving Tanuki's LESS instantiations.

This section gives a brief summary of the preprint [HLM+25]. It applies the tools developed in Section 5 to the LESS instantiation of Tanuki-3 [HLM+25], which we refer to as Tanuki-3-LESS. The resulting scheme achieves a 14.5× reduction in signature size. Our proof-of-concept implementation in C shows only a 1.2× slowdown in signing and a 1.7× slowdown in verification. One can expect a smaller slowdown factor given the faster CF implementation [BEP+25].

It provides a C implementation on an Intel Core i7-12700H @ 2.30 GHz, combining and adapting the AVX2-optimized LESS implementation of CF and RREF functions from [BBB+24] into our specification using the flags `-O3 -mavx2 -mavx -march=native -mtune=native -ftree-vectorize`. Table 1 gives a size comparison with other post-quantum blind signatures, and Table 3 shows the corresponding runtime.

Supplementary Material

Let q be a prime, $G = \mathcal{M}_n$ denote the group of $n \times n$ monomial matrices over $\mathbb{F}_q$. Set polynomially-bounded parameters $\kappa \geq 2$, $\alpha > 0$, and $\mu = \alpha\kappa$ as in the framework. Let $\mathsf{H} : \{0,1\}^* \to [\kappa]^\mu$ be a hash function with balanced output, i.e., each symbol in $[\kappa]$ appears equally often.

$\underline{\mathsf{KeyGen}(\lambda)}$**:** Sample a random full-rank matrix $\mathbf{G} \in \mathbb{F}_q^{k \times n}$. Let $\mathbf{Q}_1 = \mathsf{id}$ and set $\mathsf{pk}_1 = \overline{\mathsf{RREF}(\mathbf{G})}$. For $i = 2$ to κ, sample $\mathbf{Q}_i \leftarrow\!\!\$\ G$ and set $\mathsf{pk}_i = \mathsf{RREF}(\mathbf{G}\mathbf{Q}_i)$. Output $\mathsf{sk} = (\mathbf{Q}_2, \ldots, \mathbf{Q}_\kappa)$ and $\mathsf{pk} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_\kappa)$.

$\underline{\mathsf{Signer}_1(\mathsf{sk}, \mathsf{pk})}$**:** Run $(\mathsf{pk}', T', \sigma') \leftarrow \mathsf{KeyRand}(\mathsf{pk})$ with $T' = (\mathbf{T}'_1, \ldots, \mathbf{T}'_\kappa)$. For each $i \in [\kappa]$, set $\mathbf{Q}'_i = \mathbf{Q}_{\sigma'(i)}\mathbf{T}'_{\sigma'(i)}$. Sample $\tilde{\mathbf{Q}}_1, \ldots, \tilde{\mathbf{Q}}_\mu \leftarrow\!\!\$\ G$ and compute $\mathbf{A}_i = \mathsf{RREF}(\mathbf{G}\tilde{\mathbf{Q}}_i)$ for $i \in [\mu]$. Send $(\mathsf{pk}', (\mathbf{A}_1, \ldots, \mathbf{A}_\mu))$ to the user and keep $\tilde{\mathbf{Q}}_1, \ldots, \tilde{\mathbf{Q}}_\mu$, $\mathbf{Q}'_1, \ldots, \mathbf{Q}'_\kappa$, T', σ' as secret state.

$\underline{\mathsf{User}_1(\mathsf{pk}', (\mathbf{A}_1, \ldots, \mathbf{A}_\mu), m)}$**:** Run $(\mathsf{pk}'', T, \sigma) \leftarrow \mathsf{KeyRand}^*(\mathsf{pk}')$ with $T = (\mathbf{T}_1, \ldots, \mathbf{T}_\kappa)$. Sample $\mathbf{R}_1, \ldots, \mathbf{R}_\mu \leftarrow\!\!\$\ G$ and a random permutation π. Compute $(\mathbf{I} \mid \mathbf{B}_i), \rho_i \leftarrow \mathsf{RREF}^*(\mathbf{A}_i\mathbf{R}_i)$, then $\mathbf{B}_i \leftarrow \mathsf{CF}(\mathbf{B}_i)$ for each $i \in [\mu]$. $C = \mathsf{H}((\mathbf{I} \mid \mathbf{B}_{\pi(1)}), \ldots, (\mathbf{I} \mid \mathbf{B}_{\pi(\mu)}), \mathsf{pk}'', m)$ and parse $C = (c_1, \ldots, c_\mu)$. For each $i \in [\mu]$, set $c'_i = \sigma(c_{\pi^{-1}(i)})$. Send $C' = (c'_1, \ldots, c'_\mu)$ to the signer.

$\underline{\mathsf{Signer}_2(c', \tilde{\mathbf{Q}}_1, \ldots, \tilde{\mathbf{Q}}_\mu, T', \sigma')}$**:** For each $i \in [\mu]$, compute $\mathbf{S}'_i = (\mathbf{Q}'_{c'_i})^{-1}\tilde{\mathbf{Q}}_i$. Return $(S' = (\mathbf{S}'_1, \ldots, \mathbf{S}'_\mu), T', \sigma')$ to the user.

$\underline{\mathsf{User}_2(\mathsf{pk}, S', \pi, (\mathbf{R}_1, \ldots, \mathbf{R}_\mu), T, \sigma, T', \sigma', m)}$**:** Verify if $\mathsf{KeyRand}(\mathsf{pk}; T', \sigma') \stackrel{?}{=} \mathsf{pk}'$; abort if not. For each $i \in [\mu]$, verify that $\mathsf{RREF}\left(\mathsf{pk}'_{c'_i} \cdot \mathbf{S}'_i\right) \stackrel{?}{=} \mathbf{A}_i$; abort if not.

For each $i \in [\mu]$, set $J_i = \mathsf{CosetRep}(\rho_i \circ \mathbf{T}_{c'_i}^{-1}\mathbf{S}'_i\mathbf{R}_i)$. For each $j \in [\kappa]$, compute $I_j = \mathsf{CosetRep}(\mathbf{T}'_{\sigma'(\sigma(j))}\mathbf{T}_{\sigma(j)})$. Let $\tau = \sigma' \circ \sigma$ and return

$$\mathsf{sig} = \big(C,\ (J_{\pi(1)}, \ldots, J_{\pi(\mu)}),\ (I_1, \ldots, I_\kappa),\ \tau\big).$$

$\underline{\mathsf{Verify}(\mathsf{pk}, m, \mathsf{sig})}$**:** Parse $\mathsf{sig} = (C = (c_1, \ldots, c_\mu), (J_1, \ldots, J_\mu), (I_1, \ldots, I_\kappa), \tau)$. For each $i \in [\kappa]$, compute $\overline{\mathsf{pk}}_i = \left(\mathbf{I} \mid \mathsf{CF}\left((\mathsf{pk}_{\tau(i)})_{I_i}^{-1}(\mathsf{pk}_{\tau(i)})_{\bar{I}_i}\right)\right)$. For each $i \in [\mu]$, compute $\mathbf{B}_i = \mathsf{CF}\left((\overline{\mathsf{pk}}_{c_i})_{J_i}^{-1}(\overline{\mathsf{pk}}_{c_i})_{\bar{J}_i}\right)$.

Return 1 if $C \stackrel{?}{=} \mathsf{H}((\mathbf{I} \mid \mathbf{B}_1), \ldots, (\mathbf{I} \mid \mathbf{B}_\mu), \overline{\mathsf{pk}}, m)$ holds; otherwise output 0.

Scheme 5: The improved version of Tanuki-3-LESS. One can remove the identity matrices $\mathbf{I}$ in the hash input. The inclusion here is to make the reduction smoother.