miniMEDS

Short GMW Signatures from Isometry Hints

Dinand Blom[1], Giuseppe Lamorgese[1] Ruben Niederhagen[2,3], Lars Ran[1], and Simona Samardjiska[1]

[1] Radboud University, Nijmegen, The Netherlands
[2] Academia Sinica, Taipei, Taiwan
[3] University of Southern Denmark, Odense, Denmark
{dinand.blom,giuseppe.lamorgese,lars.ran,simona.samardjiska}@ru.nl, ruben@polycephaly.org

Abstract. Hints are a concept from side-channel analysis referring to small amounts of information that enable an adversary, via a potentially very efficient procedure, to extract the secret. We take this cryptanalytic concept and give it a constructive make-over.
We formalize a short Σ-protocol based on the famous Goldreich, Micali, Wigderson protocol, in which the responses are not of the full witness size, but are only small hints. The hints can be used to extract the full response. We show that under our formalization, these protocols preserve the security properties of Σ-protocols.
We instantiate a practical protocol using a group action from tensor isomorphism and an efficient extracting procedure borrowed from an attack by Narayanan et al. on the MEDS signature scheme. We derive a signature scheme, called miniMEDS, which demonstrates practically the benefits of our hint-based construction, allowing for 22%-24% reduction in signature size compared to the latest updated parameters of MEDS.

1 Introduction

Recently, due to major advancement in quantum computing, the projected timeline for the arrival of the quantum era has been shortened to less than 10 years. Luckily we have already started the transition to post-quantum cryptography, with the first standards for digital signatures and key encapsulation mechanisms being released by NIST in 2024. While this clearly sets in motion the migration process towards fully quantum-resistant digital society, the standardization process for basic cryptographic primitives is far from over, with standards still being prepared, the additional round for signatures still ongoing, and some countries setting up their own standardization processes. In addition, the field is maturing, and more attention gets devoted to designing efficient post-quantum advanced primitives such as threshold and ring signatures.

A common strategy for designing post-quantum signatures is from the famous Goldreich, Micali, Wigderson Σ-protocol [22] via the Fiat-Shamir transform. This approach was used in several proposals in the additional signature

round - MEDS [8], LESS [2], and ALTEQ [36]. Although none of these are moving into the third round announced recently, they are still very much interesting as basis for signatures with more advanced functionalities. The nice thing about all of these is that they are all based on hard isomorphism problems, or put differently, on the hardness of the group action inversion problem for non-commutative group actions. Therefore, analysis, design and optimization strategies follow similar patterns. For example, one can reduce signature size by using generic trade-offs between signature size, public key size, and computation time, such as replacing the response to challenge 0 by a seed and further creating more involved seed trees, introducing multiple public keys and fixed-weight challenges. Still, even with these trade-offs, the response for challenge 1 is still an isomorphism, and depending on the problem, the bitsize of an isomorphism can be huge compared to the input size (for example cubic for MEDS and ALTEQ). Therefore, in recent years a lot of effort has been put into reducing the size or representation of the used isomorphisms.

In [30], Persichetti and Santini provide an alternative formulation for the Linear Equivalence Problem (LEP) to reduce the size of the signature of LESS [2]. An improvement of this work [10] introduces the concept of canonical forms of an isometry and demonstrates how to use it to reduce the witness size, i.e. the response for challenge 1. In [9], the authors propose a technique that reduces the size of the witness in MEDS from three matrices to two vectors which in essence are compact representations of codewords. Hence, only these vectors need to be sent as a response, and later be used to uniquely expand the full isometry on the verifier side. This is already quite a big reduction, but raises the question of whether we can reduce the signature size of MEDS even more. This paper answers the question positively – we go a step further from previous results and show that we only need a single vector.

Our contribution In [27], Narayanan et al. proposed an algorithm for solving MCE exploiting the fact that one can attach a unique label that is invariant under the secret isometry to each codeword of corank 1. Thus, it can be used to match codewords that get mapped one to another via the isometry. The paper further explains how this isometry can be found from the matching pair.

Abstracting away from the enumeration needed for the algorithm, we can see that if one is given two codewords of corank 1, such that one is mapped to the other via the isometry, this isometry can be completely determined. So we can think of this pair as a *hint* for the isometry. Hint here has the exact same meaning as in the context of partial key exposure [14,25,19,26] – it is a small amount of information that helps to efficiently recover the secret.

We take this concept of *hints* that is widely exploited in side-channel analysis and leakage evaluation, and we put it into a different constructive context – as means for reducing the signature size, generically in GMW-based signatures.

We start by defining precisely what isometry hints are, and what sort of algorithms we attach to them. We then define two different protocols that are more efficient versions of the GMW Σ-protocol in terms of communication cost. Both are based on using hints, and we prove that both preserve the properties of

Σ-protocols, under conditions that we carefully state. In a natural instantiation of the first one, the hints are pairs of input points. By formalizing a special type of canonical form, in the second protocol, we can reduce the size of the hint to a single point.

We then instantiate the second, more efficient protocol using the hints that arise from Narayanan et al.'s attack. We show that the hints fit our framework, and thus by extension, that the obtained protocol is a Σ-protocol. Based on this protocol, we define a shorter version of MEDS, that we call miniMEDS. The sizes of miniMEDS are smaller than the most-optimized signatures from [9], even without any standard trade-offs optimizing for signature size. When employing these standard trade-off techniques, we observe a reduction of up to 24% compared to [9], achieved for NIST Level 5. Interestingly, in the prefered version without standard trade-offs, besides the reduction in signature size, we also gain in signing and verification time. The procedure for recovering the isometry is intrinsically faster than in [9], and we employ specific algorithmic optimizations that we show (under some mild assumptions) that do not affect the security of the scheme.

We provide a complete, constant-time implementation of miniMEDS that we appropriately benchmark and compare to other signature schemes. The implementation is available at https://github.com/IIS-ACE-lab/MiniMEDS.

2 Preliminaries

2.1 Basic notation

Let $\mathbb{F}_q$ be the finite field of order q. We use bold letters for vectors $\mathbf{a}, \mathbf{c}, \mathbf{x}, \ldots$, and matrices $\mathbf{A}, \mathbf{B}, \ldots$. The entries of a vector $\mathbf{a}$ are denoted by a_i, and we write $\mathbf{a} = (a_1, \ldots, a_n)$ for a (column) vector of dimension n. Similarly, the entries of a matrix $\mathbf{A}$ are denoted by a_{ij}. We use $\mathcal{M}_{n\times m}(q)$ to denote the space of $n \times m$ matrices over the field $\mathbb{F}_q$. We denote by $GL_n(q)$ the general linear group of degree n over $\mathbb{F}_q$. Random sampling from a set S is denoted by $a \xleftarrow{\$} S$.

2.2 Cryptographic Group Actions

Definition 1. *Let X be a set and $(G, \cdot)$ be a group. A left (resp. right) group action is a mapping*

$$\begin{aligned} \star : G \times X &\to X \\ (g, x) &\mapsto g \star x \end{aligned}$$

such that, for all $x \in X$ and $g_1, g_2 \in G$, it holds that

$$g_2 \star (g_1 \star x) = (g_2 \cdot g_1) \star x \qquad (\textit{resp. } g_2 \star (g_1 \star x) = (g_1 \cdot g_2) \star x).$$

A group action can have a number of mathematically desirable properties. For example, we say that a group action is:

- *Commutative*: for any $g_1, g_2 \in G$, we have $g_2 \star (g_1 \star x) = g_1 \star (g_2 \star x)$.
- *Transitive*: given $x_1, x_2 \in X$, there is some $g \in G$ such that $g \star x_1 = x_2$.
- *Free*: if $g \star x = x$, then g is the identity.

A *cryptographic* group action is a group action with some additional properties that are useful for cryptographic applications. First of all, the following procedures should be efficient:

- *Evaluation*: given x and g, compute $g \star x$.
- *Sampling*: sample uniformly at random from G.
- *Membership testing*: verify that $x \in X$.

Finally, cryptographic group actions should come with security guarantees; for instance, the *group action inversion problem* should be hard:

*Problem 1 (**Group Action Inversion Problem (GAIP)**).*
Given: The pair $x_1, x_2 \in X$.
Goal: Find, if any, $g \in G$ such that $g \star x_1 = x_2$.

Early constructions using group actions were based on the action of finite groups of prime order, for which the GAIP problem is given by the discrete logarithm problem. Later, the isogeny-based construction was introduced by Couveignes in [12] and independently by Rostovtsev and Stolbunov [33]. It uses the action of the *class group* on the set of isomorphism classes of ordinary elliptic curves over finite fields. This scheme was not very efficient and it was subsequently improved by adapting the group action to supersingular elliptic curves over prime fields - allowing faster computations - resulting in the well-known CSIDH [7] key exchange algorithm. The latter group action has been generalized introducing orientations of elliptic curves in the OSIDH [11] key exchange algorithm, later further generalized to SCALLOP [21], PEARL-SCALLOP [1], and group actions on higher-dimensional isogenies, like KLaPoTi [29] and qt-Pegasis [15].

Nowadays, the landscape of cryptographic group actions has also been enriched with new constructions from hard isomorphism problems, which in general ask for the isometry given two isomorphic objects.

In code-based cryptography, we have the code equivalence problem which can be instantiated differently depending on the nature of the codes and the metric considered. For linear codes in the Hamming metric we have the *linear code equivalence problem* that given two codes, asks for a monomial transformation sending one to the other. This problem underlies the security of LESS [2]. In the case of matrix codes using the rank metric, we have the matrix code equivalence problem: given two matrix codes $\mathcal{C}$ and $\mathcal{D}$, find two invertible matrices $\mathbf{A}$ and $\mathbf{B}$ such that $\mathcal{D} = \mathbf{A}\mathcal{C}\mathbf{B}$. The security of MEDS [8] relies on the hardness of this problem. A mathematically interesting observation is that matrix codes can be seen as trilinear forms, and therefore the matrix code equivalence problem can be reformulated as the trilinear form equivalence problem. When the trilinear forms are alternating, then we can build a group action on them based on the

alternating trilinear form equivalence problem: given two alternating trilinear forms ϕ, ϕ', find an invertible matrix $\mathbf{A}$ such that $\phi(\mathbf{A}(), \mathbf{A}(), \mathbf{A}()) = \phi'$. This group action is used to build the ALTEQ [35] signature scheme.

In lattice-based cryptography we have the *lattice isomorphism problem* (LIP): given two lattices, find an orthonormal tranformation sending one to the other. Based on this hard problem, Ducas and van Woerden constructed in [17] an identification scheme, a key encapsulation mechanism and a digital signature. Subsequently, the digital signature HAWK [16] has been proposed, which hinges on LIP using module lattices. From LIP, it has been constructed an action of the invertible integer matrices on the quadratic forms on $\mathbb{R}^n$, whose properties are showed in [5]. This group action has been instantiated quite recently for the first time in [28] to build blind signatures.

2.3 The GMW Σ-protocol

We give the definitions for the specific types of protocols we are interested in.

Definition 2. *A* Σ-protocol *for a binary relation* $\mathcal{R}$ *is a three-pass interactive protocol between two parties: a prover* $\mathcal{P}$ *and a verifier* $\mathcal{V}$*. The protocol is composed of the following procedures:*

I. *Keygen: on input some public data (including system parameters), output a public key* pk *(the instance) and the corresponding secret key* sk *(the witness), i.e.,* $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{R}$*. Give* sk *to the prover;* pk *is distributed publicly and is available to all parties*

II. *Commit: on input the public key* pk*, the prover* $\mathcal{P}$ *outputs a public commitment* cmt *and sends it to the verifier.*

III. *Challenge: on input the public key* pk *and the commitment* cmt*,* $\mathcal{V}$ *samples uniformly at random a challenge* ch *from the challenge space* C *and sends it to the prover.*

IV. *Response: on input the secret key* sk*, the public key* pk*, the commitment* cmt *and the challenge* ch*,* $\mathcal{P}$ *outputs a response* resp *and sends it to the verifier.*

V. *Verify: on input a public key* pk*, the commitment* cmt*, the challenge* ch*, and the response* resp*,* $\mathcal{V}$ *outputs either 1 (accept) if the* transcript $(\mathsf{cmt}, \mathsf{ch}, \mathsf{resp})$ *is valid, or 0 (reject) otherwise.*

We require that a Σ-protocol satisfies the following properties:

- **Completeness**: for all $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{R}$, when $\mathcal{P}(\mathsf{pk}, \mathsf{sk})$ and $\mathcal{V}(\mathsf{pk})$ interact with each other, $\mathcal{V}(\mathsf{pk})$ always outputs "accept".
- **Special Soundness**: There exists an efficient extractor that given two accepting transcripts $(\mathsf{pk}, \mathsf{cmt}, \mathsf{ch}, \mathsf{resp})$ and $(\mathsf{pk}, \mathsf{cmt}, \mathsf{ch}', \mathsf{resp}')$ with $\mathsf{ch} \neq \mathsf{ch}'$, extracts a witness sk for the public key pk.
- **Honest-Verifier Zero-Knowledge**: there exists an efficient probabilistic algorithm Sim (called *simulator*) that takes as input an instance pk and a challenge ch, and satisfies the following properties:

Algorithm 1 GMW protocol

I. Commit(pk)

1. $\psi_0 \xleftarrow{\$} \mathfrak{I}$.
2. $\mathcal{O}' \leftarrow \psi_0 \star \mathcal{O}_0$
3. $\mathsf{cmt} \leftarrow \mathcal{O}'$
4. Return cmt

II. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0,1\}$.
2. Return ch.

III. Response(sk, pk, cmt, ch)

1. If $\mathsf{ch} = 1$, set $\psi_1 \leftarrow \psi_0 \circ \phi^{-1}$.
2. $\mathsf{resp} \leftarrow \psi_{\mathsf{ch}}$
   Return resp.

IV. Verify(pk, cmt, ch, resp)

1. $\mathsf{cmt}' \leftarrow \psi_{\mathsf{ch}} \star \mathcal{O}_{\mathsf{ch}}$.
   If $\mathsf{cmt}' = \perp$, return "reject".
2. If $Equal(\mathsf{cmt}', \mathsf{cmt})$, return "accept".
   Otherwise, return "reject".

i. for all inputs (pk, ch), the algorithm Sim always outputs a pair (com, resp) such that (pk, com, ch, resp) is a valid transcript;

ii. for all $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{R}$ and for every challenge ch, if we compute

$$(\mathsf{com}, \mathsf{resp}) \leftarrow \mathsf{Sim}(\mathsf{pk}, \mathsf{ch}),$$

then (pk, com, ch, resp) has the same distribution as that of a transcript of a conversation between $\mathcal{P}(\mathsf{pk}, \mathsf{sk})$ and $\mathcal{V}(\mathsf{pk})$.

Schnorr's protocol [34] is likely the most well-known example of a Σ-protocol based the discrete logarithm problem. Here we will be interested in the GMW protocol, by Goldreich, Micali, and Wigderson [23]. Initially presented for the graph isomorphism problem, it can be instantiated for any hard isomorphism problem. Since these can be viewed as the GAIP problem for cryptographic group actions, we give a general description of the GMW protocol in terms of cryptographic group action in Algorithm 1.

Let $\mathfrak{O}$ be a set of objects and $\mathfrak{I}$ be the group of isomorphisms acting on $\mathfrak{O}$. Let $\phi \in \mathfrak{I}$ be an isomorphism s.t. $\mathcal{O}_1 = \phi \star \mathcal{O}_0$, with $\mathcal{O}_0, \mathcal{O}_1 \in \mathfrak{O}$. Given $\mathcal{O}_0$ and $\mathcal{O}_1$, the prover $\mathcal{P}$ wants to prove to the verifier $\mathcal{V}$ knowledge of ϕ without revealing any information about it.

3 GMW from isometry hints

The witness in the GMW protocol is typically an isometry between two objects, e.g. two tensors, two codes, two lattices, etc. This means that the ephemeral secret and the responses to the two challenges are also isometries. In the initial proposals of MEDS, ALTEQ or LESS, these were represented simply as the matrices or permutations that define the isometry, which results in a large communication cost. For example, in MEDS, the isometry is defined as $\phi = (\mathbf{A}, \mathbf{B}, \mathbf{C})$, where $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are $n \times n$ matrices, which results in a communication cost of $3n^2$ field elements for the response to each challenge.

However, we already know from cryptanalytical results on tensor isomorphism that knowing $\mathbf{A}$ is sufficient to compute $\mathbf{B}$ and $\mathbf{C}$, and thus the entire isometry ϕ [13]. The specs of MEDS [8] further show that, depending on the parameters, one needs even less information. For example, in the case of $n \times n \times n$ tensors, one needs only three rows of $\mathbf{A}$ to compute the entire isometry. This corresponds to knowing three (full-rank) pairs of matching codewords if we view the tensors as matrix codes along one of the axis. This was improved in [27] to knowing a single pair of matching corank-1 points (or codewords). Notably, this result was known empirically for a long time, in the context of Isomorphism of polynomials [20], but to the best of our knowledge no proof exists to date. Very recently [4,6], it was shown that a similar result holds for isometries between codes in the Hamming metric i.e. two matching codewords, reveal completely the isometry.

We can summarize all of these results as a single observation: the isometry ϕ between tensors, codes, trilinear forms, or sets of polynomials, can be represented by a small amount of information, which we call an *isometry hint*, from which the entire isometry can be efficiently computed.

Isometry hints. We proceed next to formalizing this notion of isometry hints and the properties that it should satisfy to be useful in the context of GMW protocols.

Definition 3. *Let $\mathfrak{O}$ be a set and $\mathfrak{I}$ be the group of all isometries on $\mathfrak{O}$ such that*

$$\begin{aligned} \star : \mathfrak{I} \times \mathfrak{O} &\to \quad \mathfrak{O} \\ (\phi, \mathcal{O}) &\mapsto \phi \star \mathcal{O} \end{aligned}$$

is a cryptographic group action. We say that a set Φ is an isometry hint *for an isometry $\phi \in \mathfrak{I}$ if:*

- *the bit-size of Φ is sublinear in the size of the isometry ϕ.*
- *It is easy to sample an isometry hint for the isometry ϕ, given $\mathcal{O}$ and ϕ. That is, there exists an efficient probabilistic sampler, indistinguishable from a random sampler, that we denote by SI and write $\Phi \leftarrow SI(\phi, \mathcal{O}, \phi \star \mathcal{O})$.*
- *The group action above extends to the isometry hint Φ. That is, the properties of Φ are preserved under the action of $\mathfrak{I}$, i.e. if Φ is an isometry hint for ϕ, then $\psi \star \Phi$ is an isometry hint for $\phi \circ \psi$ with $\psi \in \mathfrak{I}$.*
- *There exists a PPT algorithm $\mathcal{E}$ called* isometry extractor*, that given Φ and $\mathcal{O}_0, \mathcal{O}_1 \in \mathfrak{O}$ extracts an isometry between $\mathcal{O}_0, \mathcal{O}_1$, if it exists (i.e. a ϕ such that $\phi \star \mathcal{O}_0 = \mathcal{O}_1$), or $\perp$ otherwise. We will write $\phi \leftarrow \mathcal{E}(\Phi, \mathcal{O}_0, \mathcal{O}_1)$.*

For many interesting objects and isometries, we already know (see previous section) that isometry hints exist. For example, for tensors of dimension $n \times n \times n$, we can use two pairs of matching points (corresponding to codewords) as an isometry hint, as demontrated in [9]. A recent paper by Bardet et al. [4] shows that for code equivalence, a single pair of matching points is sufficient to extract the isometry and can thus be used as an isometry hint. We will later show a similar statement for $n \times n \times n$ tensors.

Algorithm 2 GMW Σ-protocol using isometry hints

I. Keygen()

1. $\mathcal{O}_0 \xleftarrow{\$} \mathfrak{O}$.
2. Set $\phi \xleftarrow{\$} \mathfrak{I}$.
3. Set $\mathcal{O}_1 \leftarrow \phi(\mathcal{O}_0)$.
4. Set $\mathsf{sk} \leftarrow \phi$, $\mathsf{pk} = (\mathcal{O}_0, \mathcal{O}_1)$.
5. Return $(\mathsf{sk}, \mathsf{pk})$.

II. Commit(pk)

1. Set $\psi_0 \xleftarrow{\$} \mathfrak{I}$.
2. Set $\mathcal{O}' \leftarrow \psi_0(\mathcal{O}_0)$
3. $\mathsf{cmt} \leftarrow \mathcal{O}'$
4. Return cmt

III. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0, 1\}$.
2. Return ch.

IV. Response(sk, pk, cmt, ch)

1. If $\mathsf{ch} = 1$, set $\psi_1 \leftarrow \psi_0 \circ \mathsf{sk}^{-1}$.
2. Generate $\Phi_{\mathsf{ch}} \leftarrow SI(\psi_{\mathsf{ch}}, \mathcal{O}_{\mathsf{ch}}, \mathcal{O}')$
3. $\mathsf{resp} \leftarrow \Phi_{\mathsf{ch}}$
   Return resp.

V. Verify(pk, cmt, ch, resp)

1. Extract $\psi' \leftarrow \mathcal{E}(\Phi_{\mathsf{ch}}, \mathcal{O}_{\mathsf{ch}}, \mathcal{O}')$
2. If $\psi' = \perp$ return "reject".
3. $\mathsf{cmt}' \leftarrow \psi'(\mathcal{O}_{\mathsf{ch}})$.
   If $\mathsf{cmt}' = \perp$, return "reject".
4. If $Equal(\mathsf{cmt}', \mathsf{cmt})$, return "accept".
   Otherwise, return "reject".

Example 1. In this example we show that the construction from [9] fits the definition of isometry hints. Let $\mathfrak{O}$ be the set of all $n \times n \times n$ tensors over $\mathbb{F}_q$ and let $\mathfrak{I}$ be the group of isometries $\mathrm{GL}_n(q)^3$. Then, an isometry hint for an isometry ϕ between two tensors $\mathcal{T}_0, \mathcal{T}_1 \in \mathfrak{O}$ can be defined as two pairs of matching points $(\mathbf{u}_i, \mathbf{v}_i)$, $i = 1, 2$, where $\mathbf{u}_i$ and $\mathbf{v}_i$ are points in $\mathbb{F}_q^n$ such that $\mathbf{u}_i = \mathbf{A}\mathbf{v}_i$, where $\mathbf{A}$ is the first component of the isometry ϕ. Furthermore, we require $\mathcal{T}_0(\mathbf{u}_i, -, -)$ to be full-rank matrices.

Then the size of the hint $2n\log(q)$ is indeed sublinear in the size of the isometry $3n^2\log(q)$. Furthermore, it is easy to sample 2 points $\mathbf{u}_i$ such that the corresponding matrices are full rank and construct $\mathbf{v}_i$ from these and $\mathbf{A}$. By defining the action of $\mathrm{GL}_n(q)^3$ on a hint $(\mathbf{u}_i, \mathbf{v}_i)$ to be $(\tilde{\mathbf{A}}^{-1}\mathbf{u}_i, \tilde{\mathbf{A}}^{-1}\mathbf{v}_i)$, we see that this hint respects the group action on $\mathfrak{O}$. Finally, as used constructively in [9], there is indeed an extractor that computes $\mathbf{B}, \mathbf{C}$ from the collision and then solves for $\mathbf{A}$.

We are now ready to define the GMW protocol in terms of isometry hints.

Theorem 1. *The protocol in Algorithm 2 is a Σ-protocol for the GAIP relation, where the $Equal()$ is a function that efficiently compares the two objects.*

Proof. In order to show that the protocol is a Σ-protocol we need to prove the properties of completeness, special soundness and honest-verifier zero-knowledge.

Completeness. If the prover and the verifier follow the protocol, then the response resp generated by the prover will be a valid isometry hint for the isometry

ψ_{ch} between $\mathcal{O}_{\mathsf{ch}}$ and $\mathcal{O}'$. The verifier then uses the isometry extractor to extract an isometry ψ' from the response. This isometry might not be the same as the one generated by the honest prover, but if everything is done honestly, it will still map $\mathcal{O}_{\mathsf{ch}}$ to $\mathcal{O}'$. In this case, it might happen that the representation of $\mathcal{O}'$ obtained by applying ψ' to $\mathcal{O}_{\mathsf{ch}}$ is different from the one generated by the honest prover. This is not a problem since $\mathit{Equal}(\mathsf{cmt}', \mathsf{cmt})$ compares the objects and not their representations. See Remark 1 for more details on this point.

Honest-Verifier Zero-Knowledge. Given $\mathsf{pk} = (\mathcal{O}_0, \mathcal{O}_1)$, if the challenge ch is equal to 0, then the simulator proceeds exactly as the prover does, namely running $\mathsf{cmt}_0^* \leftarrow \mathbf{Commit}(\mathsf{pk})$ and $\mathsf{rsp}_0^* \leftarrow \mathbf{Response}(\mathsf{sk}^*, \mathsf{pk}, \mathsf{cmt}_0^*, 0)$, where sk^* is a dummy argument that is not used. Hence, in this case zero-knowledge trivially follows.

If $\mathsf{ch} = 1$, let $\mathbf{Commit}_{\mathcal{O}_1}(\mathsf{pk})$ be $\mathbf{Commit}(\mathsf{pk})$ where we substitute $\mathcal{O}_0$ with $\mathcal{O}_1$. Then the simulator produces the commitment-response pair $(\mathsf{cmt}_1^*, \mathsf{rsp}_1^*)$, where $\mathsf{cmt}_1^* \leftarrow \mathbf{Commit}_{\mathcal{O}_1}(\mathsf{pk})$ and $\mathsf{rsp}_1^* \leftarrow \mathbf{Response}(\mathsf{sk}^*, \mathsf{pk}, \mathsf{cmt}_1^*, 0)$, with sk^* being a dummy argument that is not used. Then we have $(\mathsf{cmt}_1^*, \mathsf{rsp}_1^*) = (\mathcal{O}^*, \mathit{SI}(\psi_1^*, \mathcal{O}_1, \mathcal{O}^*))$, where ψ_1^* is the isometry from $\mathcal{O}_1$ generated at random in $\mathbf{Commit}_{\mathcal{O}_1}(\mathsf{pk})$. The transcript $(\mathsf{pk}, \mathsf{cmt}_1^*, 1, \mathsf{rsp}_1^*)$ is accepted by the verifier, as the latter runs $\mathcal{E}(\mathit{SI}(\psi_1^*, \mathcal{O}_1, \mathcal{O}^*), \mathcal{O}_1, \mathcal{O}^*)$ extracting an isometry ψ' from $\mathcal{O}_1$ to $\mathcal{O}^*$, and therefore we have that $\mathsf{cmt}' = \psi' \star \mathcal{O}_1$, $\mathcal{O}^* = \mathsf{cmt}_1^*$, from which it follows that $\mathit{Equal}(\mathsf{cmt}', \mathsf{cmt}_1^*)$ is true. Next, let us consider an honestly generated commitment-response pair $(\mathsf{cmt}_1, \mathsf{rsp}_1)$. We want to prove that $(\mathsf{cmt}_1, \mathsf{rsp}_1)$ has the same distribution as $(\mathsf{cmt}_1^*, \mathsf{rsp}_1^*)$. Now, since ψ_0 and ψ_1^* are sampled at random from $\mathfrak{I}$, then also ψ_1 appears as a random isometry, and therefore it follows that rsp_1 has the same distribution as rsp_1^*. Finally, the two commitments cmt_1 and cmt_1^* are derived by applying the random isometries ψ_1 and ψ_1^* respectively to $\mathcal{O}_1$. Hence, cmt_1 and cmt_1^* have the same distribution.

Special Soundness. Let $(\mathsf{pk}, \mathsf{cmt}, 0, \mathsf{rsp}_0)$ and $(\mathsf{pk}, \mathsf{cmt}, 1, \mathsf{rsp}_1)$ be two valid transcripts. Then we can run

$$\psi_0 \leftarrow \mathcal{E}(\mathsf{rsp}_0, \mathcal{O}_0, \mathsf{cmt}),$$
$$\psi_1 \leftarrow \mathcal{E}(\mathsf{rsp}_1, \mathcal{O}_1, \mathsf{cmt})$$

from which we obtain the isometry $\psi_1^{-1} \circ \psi_0$ such that $(\psi_1^{-1} \circ \psi_0) \star \mathcal{O}_0 = \mathcal{O}_1$ which is a witness for our relation.

Remark 1. It may happen that the verifier extracts a different isometry than the one generated by the honest prover. This may happen if the automorphism group of the object is not trivial. One way to take care of the problem is to make the assumption that the automorphism group is trivial. This will imply that if there are two isometries that map $\mathcal{O}_{\mathsf{ch}}$ to $\mathcal{O}'$, then they must be the same isometry. The trivial automorphism assumption is not really a strong assumption, since for most of the objects we are interested in, the automorphism group is indeed trivial with very high probability. For example, this is the case for random codes, random lattices and random tensors. It has also been used regularly in the literature [32,8,27].

However, the trivial automorphism assumption is not really necessary for showing the completeness of the protocol, at least not for the type of objects we are used to. The difference is that if the automorphism group is not trivial, the verifier should verify that cmt and cmt' are actually the same object, but with a different representation. This could be done using for example some unique systematic representation of the objects, or using an efficient equality checking procedure, both of which exist for the usual suspects of codes, lattices and tensors.

3.1 A step further: Canonical Forms for more efficient communication

Previously, in Example 1 we defined the set Φ as a set of pairs of matching points. The first point corresponds to an element (let's say a codeword) in $\mathcal{O}_0$ and the second point corresponds to the image of that element (codeword) under the isometry ϕ in $\mathcal{O}_1$. It seems natural that we require points in both objects, i.e. pairs of points, because they define which point gets mapped to which.

A natural question is whether we can avoid using these pairs, and instead, work only with points/elements in one of the objects. For this we need the concept of canonical forms.

Definition 4. *A canonical form of $\mathcal{O} \in \mathfrak{O}$ with respect to a set Φ of hints is a unique element $\mathcal{O}_\Phi$ in the orbit of $\mathcal{O}$ such that the following is satisfied:*

- *$\Phi \mapsto \mathcal{O}_\Phi$ is an injection.*

If the automorphism group of $\mathcal{O}$ is trivial, Φ defines a unique isometry ϕ such that $\mathcal{O}_\Phi = \phi \star \mathcal{O}$.

Example 2. We will use the construction in [9] as our running example. In Example 1 we used hints to compactify elements of $\mathrm{GL}_n(q)^3$. In this example we will show that we can extend this to be a canonical form hint. Given a tensor $\mathcal{T}$ we call a pair of full-rank points $(\mathbf{u}_1, \mathbf{u}_2)$ a canonical form hint.

Then, given such a hint, we can construct $\mathbf{B}, \mathbf{C}$ in such a way that they map $\mathcal{T}(\mathbf{u}_1, -, -)$ to the identity matrix $\mathbf{I}_n$ and $\mathcal{T}(\mathbf{u}_2, -, -)$ to the identity matrix shifted up by one $\mathbf{J}_n$. This uniquely determines $\mathbf{B}, \mathbf{C}$. Then, we can compute $(\mathbf{I}_n, \mathbf{B}, \mathbf{C}) \star \mathcal{T}$ and echolonize it with respect to the first axis. This gives us a transformation $(\mathbf{A}, \mathbf{I}_n, \mathbf{I}_n)$. Composing these gives us a unique canonical form for $(\mathcal{T}, (\mathbf{u}_1, \mathbf{u}_2))$. One can check that this is indeed equivariant under the group action.

Remark 2. Note that, we could define a different type of canonical form that would be independent of any type of representation of the object, or any type of hint. Such canonical forms are not uncommon in the literature. Think of for example of the well known (and useful) Gauss-Jordan form of a matrix, or the Jordan canonical form. However, in our context, such a representation-independent unique form is not useful, and does not help in our overall framework.

Using canonical forms we can improve the communication cost of the GMW protocol given in Algorithm 2. Instead of committing to a random object isometric to $\mathcal{O}_0$, the prover can commit to the canonical form of $\mathcal{O}_0$ with respect to a randomly chosen set Φ. Similarly, as previously, we require the same properties for the isometry hint, and the existence of two algorithms: 1) an isometry hint sampler and 2) an extractor. The difference now is that the extractor is an efficient algorithm that given an object $\mathcal{O}$ and a set Φ of isometry hints computes the canonical form of $\mathcal{O}$ with respect to Φ;

Definition 5. *Let $(\mathfrak{I}, \mathfrak{O}, \star)$ be be the cryptographic group action as in Definition 3. We say that a set Φ_C is a* canonical-form hint *for $\mathcal{O} \in \mathfrak{O}$ if:*

- *the bit-size of Φ_C is sublinear in the size of the isometries from $\mathfrak{I}$.*
- *It is easy to sample a canonical-form hint given $\mathcal{O}$. That is, there exists an efficient probabilistic sampler, indistinguishable from a random sampler, that we denote by SI_C and write $\Phi_C \xleftarrow{\$} SI_C(\mathcal{O})$.*
- *There exists an efficient algorithm $\mathcal{E}$ called* canonical-form extractor*, that given Φ_C and $\mathcal{O} \in \mathfrak{O}$ computes the canonical form $\mathcal{O}_{\Phi_C}$ of $\mathcal{O}$ with respect to Φ_C, and an isometry ϕ_C such that $\mathcal{O}_{\Phi_C} = \phi_C \star \mathcal{O}$. If the automorphism group of $\mathcal{O}$ is trivial, this isometry is unique and the extractor is deterministic. We will write $(\phi_C, \mathcal{O}_{\Phi_C}) \leftarrow \mathcal{E}_{canon}(\mathcal{O}, \Phi_C)$.*
- *The group action $(\mathfrak{I}, \mathfrak{O}, \star)$ extends to the canonical-form hint Φ_C in the same way as in Definition 3. That is, the properties of Φ_C are preserved under the action of elements of $\mathfrak{I}$, i.e. if Φ_C is a canonical-form hint for $\mathcal{O}$, then $\psi \star \Phi_C$ is a canonical-form hint for $\psi^{-1} \star \mathcal{O}$ with $\psi \in \mathfrak{I}$, and*

$$\mathcal{E}_{canon}(\mathcal{O}, \Phi_C)[2] = \mathcal{E}_{canon}(\psi^{-1} \star \mathcal{O}, \psi \star \Phi_C)[2]$$

i.e. the two obtained canonical forms are equal.

The updated version of the GMW protocol using canonical forms is given in Algorithm 3. We can state the following theorem.

Theorem 2. *The protocol in Algorithm 3 is a Σ-protocol for the GAIP relation, under the conditions stated in Definition 5 and assuming the trivial automorphism assumption holds for elements in $\mathfrak{O}$.*

Proof. Again, we prove the properties of completeness, special soundness and honest-verifier zero-knowledge assuming the conditions from Definition 5.

Completeness. We consider an honestly generated commitment-response pair $(\mathsf{cmt}_{\mathsf{ch}}, \mathsf{rsp}_{\mathsf{ch}})$ related to the challenge ch. If $\mathsf{ch} = 0$, then prover and verifier perform the same operations on the same inputs, therefore completeness in this case is trivial. If $\mathsf{ch} = 1$, then on the prover's side $\mathcal{E}_{canon}(\mathcal{O}_0, \Psi_0)$ is computed, while the verifier computes $\mathcal{E}_{canon}(\mathcal{O}_1, \Psi_1)$. By definition is equal to $\mathcal{E}_{canon}(\mathcal{O}_1, \Psi_1)[2] = \mathcal{E}_{canon}(\phi \star \mathcal{O}_0, \phi^{-1} \star \Psi_0)[2]$. But since by hypothesis

$$\mathcal{E}_{canon}(\mathcal{O}_0, \Psi_0)[2] = \mathcal{E}_{canon}(\phi(\mathcal{O}_0), \phi^{-1} \star \Psi_0)[2],$$

the verifier accepts.

Algorithm 3 Canonical form GMW Σ-protocol

I. Keygen()

1. $\mathcal{O}_0 \xleftarrow{\$} \mathfrak{O}$
2. Set $\phi \xleftarrow{\$} \mathfrak{I}$
3. Set $\mathcal{O}_1 \leftarrow \phi(\mathcal{O}_0)$
4. Set $\mathsf{sk} \leftarrow \phi$, $\mathsf{pk} = (\mathcal{O}_0, \mathcal{O}_1)$
5. Return $(\mathsf{sk}, \mathsf{pk})$

II. Commit(pk)

1. Set $\Psi_0 \xleftarrow{\$} SI_C(\mathcal{O}_0)$
2. Extract $(\psi_0, \mathcal{O}_{\Psi_0}) \leftarrow \mathcal{E}_{canon}(\mathcal{O}_0, \Psi_0)$
3. $\mathsf{cmt} \leftarrow \mathcal{O}_{\Psi_0}$
4. Return cmt

III. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0,1\}$.
2. Return ch.

IV. Response(sk, pk, cmt, ch)

1. If $\mathsf{ch} = 1$, generate $\Psi_1 \leftarrow \phi^{-1} \star \Psi_0$
2. $\mathsf{resp} \leftarrow \Psi_{\mathsf{ch}}$
   Return resp.

V. Verify(pk, cmt, ch, resp)

1. Extract $(\psi'_{\mathsf{ch}}, \mathcal{O}'_{\Psi_{\mathsf{ch}}}) \leftarrow \mathcal{E}_{canon}(\mathcal{O}_{\mathsf{ch}}, \mathsf{resp})$
2. If $\psi' = \perp$ return "reject".
3. $\mathsf{cmt}' \leftarrow \mathcal{O}'_{\Psi_{\mathsf{ch}}}$.
   If $\mathsf{cmt}' = \perp$, return "reject".
4. If $\mathsf{cmt}' = \mathsf{cmt}$, return "accept". Otherwise, return "reject".

Honest-Verifier Zero-Knowledge. Given $\mathsf{pk} = (\mathcal{O}_0, \mathcal{O}_1)$, if the challenge ch is equal to 0, then the simulator proceeds exactly as the prover does, and therefore zero-knowledge is trivial in this case.

For $\mathsf{ch} = 1$, let $\mathbf{Commit}_{\mathcal{O}_1}(\mathsf{pk})$ be the function $\mathbf{Commit}(\mathsf{pk})$ where we substitute $\mathcal{O}_0$ with $\mathcal{O}_1$. Then the simulator produces the commitment-response pair $(\mathsf{cmt}_1^*, \mathsf{rsp}_1^*)$, where $\mathsf{cmt}_1^* \leftarrow \mathbf{Commit}_{\mathcal{O}_1}(\mathsf{pk})$ and $\mathsf{rsp}_1^* \leftarrow \mathbf{Response}(\mathsf{sk}^*, \mathsf{pk}, \mathsf{cmt}_1^*, 0)$, with sk^* being a dummy argument that is not used. Then the transcript $(\mathsf{pk}, \mathsf{cmt}_1^*, 1, \mathsf{rsp}_1^*)$ is accepted by the verifier as prover and verifier run $\mathcal{E}_{canon}$ on the same inputs.

What remains is to prove that an honestly generated commitment-response pair $(\mathsf{cmt}_1, \mathsf{rsp}_1) = (\mathcal{O}', \phi^{-1} \star \Psi_0)$ has the same distribution as $(\mathsf{cmt}_1^*, \mathsf{rsp}_1^*) = (\mathcal{O}^*, \Psi_1^*)$. We can notice that Ψ_0 and Ψ_1^* are both sampled using the sampler SI which is indistinguishable from a random sampler that samples uniformly at random from the set of all allowed hints for $\mathcal{O}_0$ and $\mathcal{O}_1$. (For the purpose of this proof we call them $\mathfrak{P}_0$ and $\mathfrak{P}_1$ respectively). Therefore for the purpose of the proof, we can replace SI by a random sampler. Since ϕ^{-1} is an isometry, $\phi^{-1} \star \Psi_0$ is uniformly distributed in $\mathfrak{P}_1$, therefore rsp_1 has the same distribution as rsp_1^*.

Finally, we show that we can deterministically derive cmt_1 and cmt_1^* from $(\mathcal{O}_1, \mathsf{rsp}_1)$ and $(\mathcal{O}_1, \mathsf{rsp}_1^*)$ respectively:

$$\mathcal{E}_{canon}(\mathcal{O}_1, \mathsf{rsp}_1)[2] = \mathcal{E}_{canon}(\phi \star \mathcal{O}_0, \phi^{-1} \star \psi_0)[2] = \mathcal{E}_{canon}(\mathcal{O}_0, \psi_0)[2] = \mathsf{cmt}_1,$$
$$\mathcal{E}_{canon}(\mathcal{O}_1, \mathsf{rsp}_1^*)[2] = \mathsf{cmt}_1^*.$$

Hence, cmt_1 and cmt_1^* have the same distribution.

Special Soundness. Same as proof of Theorem 1.

4 Instantiation for tensor isomorphism

In the previous section, we presented two optimizations for the GMW protocol, assuming the existence of some specific types of isometry hints. We carefully showed the exact properties that need to be satisfied by these hints in order to preserve the security claims of GMW protocol. We now turn to a specific, practical instantiation of the second, more efficient Canonical-form GMW protocol from Algorithm 3. The one in Algorithm 2 can analogously be instantiated using the exact same primitives.

As our instantiation is based on the Tensor Isomorphism problem, we start by recalling some relevant notions about tensors and the basic GMW protocol from tensor isomorphism.

Tensors. A tensor (or more precisely 3-tensor) $\mathcal{T} \in \mathbb{F}_q^{n \times n \times n}$ is an $n \times n \times n$-dimensional object with entries from $\mathbb{F}_q$. We can view $\mathcal{T}$ as an ordered list of n matrices from $\mathcal{M}_{n \times n}(\mathbb{F}_q)$, which directly corresponds to viewing the tensor as a matrix code [8]. Here we adopt the representation as a trilinear map $\mathcal{T} : \mathbb{F}_q^{n \times n \times n} \to \mathbb{F}_q$, defined by

$$\mathcal{T}(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{i,j,k} T_{ijk} x_i y_j z_k$$

For a fixed field $\mathbb{F}_q$, we denote the set of all $n \times n \times n$ tensors by $\mathfrak{T}_n$. Note that one could define $n \times m \times k$ more generally, but we will only make use of $n \times n \times n$ tensors in the following.

Given a tensor $\mathcal{T}_0$, and a triplet of matrices $(\mathbf{A}, \mathbf{B}, \mathbf{C}) \in GL_n(q)$, we can construct a new tensor $\mathcal{T}_1$ as follows:

$$\mathcal{T}_1(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \mathcal{T}_0(\mathbf{A}x, \mathbf{B}y, \mathbf{C}z) \tag{1}$$

Since the transformation $\phi = (\mathbf{A}, \mathbf{B}, \mathbf{C})$ preserves the rank when we view the tensors as matrix codes, we will refer to ϕ as the isometry between $\mathcal{T}_0$ and $\mathcal{T}_1$.

Definition 6. *(3-Tensor-Isomorphism (3-TI) problem) Given tensors $\mathcal{T}_0$ and $\mathcal{T}_1$, find, if it exists, an isometry $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ such that*

$$\mathcal{T}_1(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \mathcal{T}_0(\mathbf{A}x, \mathbf{B}y, \mathbf{C}z) \tag{2}$$

for all vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}$.

Algorithm 4 Basic MEDS Σ-protocol

Public Data

$q, n \in \mathbb{N}$.
$\mathbf{A}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$,
$\mathbf{B}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$,
$\mathbf{C}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$.

I. Keygen()

1. $\mathcal{T}_0 \xleftarrow{\$} \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n$.
2. Set $(\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1) \xleftarrow{\$} \mathrm{GL}_n(q) \times \mathrm{GL}_n(q) \times \mathrm{GL}_n(q)$.
3. Set $\mathcal{T}_1 \leftarrow (\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1) \star \mathcal{T}_0$.
4. Set $\mathsf{sk} \leftarrow (\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1)$, $\mathsf{pk} = (\mathcal{T}_0, \mathcal{T}_1)$.
5. Return $(\mathsf{sk}, \mathsf{pk})$.

II. Commit(pk)

1. Set $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}}) \xleftarrow{\$} \mathrm{GL}_n(q) \times \mathrm{GL}_n(q) \times \mathrm{GL}_n(q)$.
2. Set $\tilde{\mathcal{T}} \leftarrow (\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}}) \star \mathcal{T}$
3. $\mathsf{cmt} \leftarrow \tilde{\mathcal{T}}$
4. Return cmt

III. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0, 1\}$.
2. Return ch.

IV. Response(sk, pk, cmt, ch)

1. $\mathsf{resp} \leftarrow (\mathbf{A}_{\mathsf{ch}}^{-1} \cdot \tilde{\mathbf{A}}, \mathbf{B}_{\mathsf{ch}}^{-1} \cdot \tilde{\mathbf{B}}, \mathbf{C}_{\mathsf{ch}}^{-1} \cdot \tilde{\mathbf{C}})$. Return resp.

V. Verify(pk, cmt, ch, resp)

1. Parse resp into $(\mathbf{A}', \mathbf{B}', \mathbf{C}')$. If $\mathbf{A}' \notin \mathrm{GL}_n(q)$ or $\mathbf{B}' \notin \mathrm{GL}_n(q)$, or $\mathbf{C}' \notin \mathrm{GL}_n(q)$, return "reject".
2. $\mathsf{cmt}' \leftarrow (\mathbf{A}', \mathbf{B}', \mathbf{C}') \star \mathcal{T}_{\mathsf{ch}}$. If $\mathsf{cmt}' = \perp$, return "reject".
3. If $\mathsf{cmt}' = \mathsf{cmt}$, return "accept". Otherwise, return "reject".

Fixing one axis of $\mathcal{T}$, using some point $\mathbf{u} \in \mathbb{F}_q^n$, for ex. as $\mathcal{T}(\mathbf{u}, -, -)$, we obtain a bilinear form that can be viewed as a matrix. We will be interested when this matrix has a specific rank. In particular, if the rank of this matrix is $n - 1$ we will refer to the point as corank-1 point.

The MEDS Σ-protocol. It is well known from the literature (for ex. [8]) that the triplet $(\mathrm{GL}_n(q) \times \mathrm{GL}_n(q) \times \mathrm{GL}_n(q), \mathfrak{T}_n, \star)$ where $\star$ is defined by:

$$(\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}_0 := \mathcal{T}_1$$

with $\mathcal{T}_1$ obtained using equation (1) is a right cryptographic group action due to the 3-TI problem being considered hard on average. The GMW protocol instantiated with this group action is given in Algorithm 4. It served as the basis for the MEDS signature scheme [8], therefore we refer to it as the MEDS Σ-protocol. Note that in [8], further optimizations are applied, but here we present the simplest variant for better readability.

4.1 Narayanan's distinguishing invariant

In [27], Narayanan et al. observe that one can create a global invariant function for $n \times n \times n$ tensors that taking a tensor as input, maps each corank-1 projective point from $\mathbb{P}(\mathbb{F}_q^n)$ to a unique value. They call such a function a *distinguishing invariant*.

They show that such a function exists for the set of projective corank-1 points under the trivial automorphism group assumption, nowadays widely accepted as a mild in the literature. This function F that they construct actually maps the tensor $\mathcal{T}$ to an isomorphic one that is uniquely (up to a negligible probability) tied to the projective point we started from. To emphasize this connection to the starting point let us denote it by $\mathcal{T}_{\hat{\mathbf{u}}}$. In addition, the procedure actually constructs the isomophism $\phi_{\hat{\mathbf{u}}}$ between the original tensor and the new one.

This property makes it possible to use the invariant to solve the tensor isomorphism problem, notably breaking the proposed parameters of the NIST submission MEDS. On a high level, since this is a distinguishing property, the problem of finding the isomorphism between two tensors $\mathcal{T}_0$ and $\mathcal{T}_1$ can be reduced to finding projective corank-1 points $\hat{\mathbf{u}}_0$ and $\hat{\mathbf{u}}_1$ such that $F(\mathcal{T}_0, \hat{\mathbf{u}}_0) = F(\mathcal{T}_1, \hat{\mathbf{u}}_1)$. Moreover, since the procedures for finding $F(\mathcal{T}_0, \hat{\mathbf{u}}_0)$ and $F(\mathcal{T}_1, \hat{\mathbf{u}}_1)$ construct the isomorphisms $\phi_{\hat{\mathbf{u}}_0}$ and $\phi_{\hat{\mathbf{u}}_1}$, the isomorphism between $\mathcal{T}_0$ and $\mathcal{T}_1$ can be obtained as $\phi_{\hat{\mathbf{u}}_1}^{-1} \circ \phi_{\hat{\mathbf{u}}_0}$.

In the next subsection we show that we can use Narayanan's distinguishing invariant to construct a canonical form extractor that needs a *single* corank-1 point as a hint.

4.2 Constructing a canonical form extractor from Narayanan's invariant

Algorithms 5, 6 and 7 are adapted from Narayanan et al. [27].

Theorem 3. *Let $\mathbb{F}_q$ be a field of polynomial size in n. Let $\mathbf{u}$ be a corank-1 point for a tensor $\mathcal{T}$ in $\mathbb{F}_q^{n \times n \times n}$. Then, with high probability, $\mathbf{u}$ is a canonical-form hint with an efficient canonical form extractor $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$ for the TI relation, where $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$ is defined by Algorithm 5.*

Proof. We want to show that the $\mathbf{u}$ satisfies the conditions from Definition 5.

- Since the bitsize of the hint $\mathbf{u}$ is $n \log_2 q$, it is clear that it is sublinear compared to the size of the isometry which is $3n^2$.
- A corank-1 point for $\mathcal{T}$ can be sampled using a simple rejection sampling technique as follows.
  Sample a point $\mathbf{u} \in \mathbb{F}_q^n$, evaluate to obtain the bilinear form $\mathcal{T}(\mathbf{u}, -, -)$, check the rank of the obtained form. If it is $n-1$, we have found a corank-1 point. If not repeat the procedure.
  Since approximately $1/q$ of all n-dimensional points are corank-1 points, we expect the procedure to find a corank-1 point in at most q steps.
  The total runtime of the sampler is then $O(qn^3)$. Since rejection sampling is statistically indistinguishable from random sampling, and q is polynomial in n we have our efficient sampler.

Algorithm 5 $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$

Require: 3-tensor $\mathcal{T}$, and a corank-1 point $\mathbf{u}$

Construct $(\mathbf{L}_u, \mathbf{L}_v, \mathbf{L}_w) \leftarrow \mathsf{Corank1Walk}(\mathcal{T}, \mathbf{u})$.
Compute $\tilde{\mathcal{T}}_{\mathbf{u}} \leftarrow (\mathbf{L}_u, \mathbf{L}_v, \mathbf{L}_w) \star \mathcal{T}$
Construct $(\mathbf{F}, \mathbf{G}, \mathbf{H}) \leftarrow \mathsf{Scale}(\tilde{\mathcal{T}}_{\mathbf{u}})$.
Compute $\mathcal{T}_{\mathbf{u}} \leftarrow (\mathbf{F}, \mathbf{G}, \mathbf{H}) \star \tilde{\mathcal{T}}_{\mathbf{u}}$
return $\mathcal{T}_{\mathbf{u}}$.

Algorithm 6 $\mathsf{Corank1Walk}(\mathcal{T}, \mathbf{u})$

Require: 3-tensor $\mathcal{T}$, and a corank-1 point $\mathbf{u}$

$\mathbf{u}_1 \leftarrow \mathbf{u}$
$L_U \leftarrow (\mathbf{u}_1)$, $L_V \leftarrow ()$, $L_W \leftarrow ()$
for $i = 1$ to n **do**
    "Find a vector $\mathbf{v}_i$ in the right kernel of $\mathcal{T}(\mathbf{u}_i, -, -)$"
    if $\mathcal{T}(-, \mathbf{v}_i, -)$ is not corank-1 or $\mathbf{v}_i \in \mathrm{Span}(L_V)$ **then return** "fail"
    $L_V \leftarrow (L_V, \mathbf{v}_i)$
    "Find a vector $\mathbf{w}_i$ that is in the right kernel of $\mathcal{T}(-, \mathbf{v}_i, -)$"
    if $\mathcal{T}(-, -, \mathbf{w}_i)$ is not corank-1 or $\mathbf{w}_i \in \mathrm{span}(L_W)$ **then return** "fail"
    $L_W \leftarrow (L_W, \mathbf{w}_i)$
    if i == n **then**
        break
    "Find a vector $\mathbf{u}_{i+1}$ that is in the right kernel of $\mathcal{T}(-, -, \mathbf{w}_i)$"
    if $\mathcal{T}(\mathbf{u}_{i+1}, -, -)$ is not corank-1 or $\mathbf{u}_{i+1} \in \mathrm{span}(L_U)$ **then return** "fail"
    $L_U \leftarrow (L_U, \mathbf{u}_{i+1})$
Define $\mathbf{L}_U = [\mathbf{u}_1, .., \mathbf{u}_n], \mathbf{L}_V = [\mathbf{v}_1, .., \mathbf{v}_n], \mathbf{L}_W = [\mathbf{w}_1, .., \mathbf{w}_n]$.
return $\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W$

Algorithm 7 $\mathsf{Scale}()$

Require: 3-tensor $\mathcal{T}$

Define, for $i, j, k \geq 3$:

$$a_i \leftarrow \mathcal{T}(\mathbf{e}_i, \mathbf{e}_2, \mathbf{e}_1),\ b_j \leftarrow \mathcal{T}(\mathbf{e}_1, \mathbf{e}_j, \mathbf{e}_1),\ c_k \leftarrow \mathcal{T}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_k),$$
$$d_1 \leftarrow \mathcal{T}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_1),\ d_2 \leftarrow \mathcal{T}(\mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5),\ d_3 \leftarrow \mathcal{T}(\mathbf{e}_1, \mathbf{e}_3, \mathbf{e}_2),\ d_4 \leftarrow \mathcal{T}(\mathbf{e}_2, \mathbf{e}_1, \mathbf{e}_2).$$

Define

$$h_1 \leftarrow 1, h_2 \leftarrow b_3/d_3,$$
$$f_1 \leftarrow 1/d_1, \quad f_2 \leftarrow (d_1^2 \cdot 1 \cdot d_2)^{-1} b_3 c_5,$$
$$g_1 \leftarrow 1/(f_2 h_2 d_4), g_2 \leftarrow 1,$$

For $3 \leq i, j, k \leq n$, define

$$f_i = d_1 f_1 / a_i, \quad g_j = d_1 g_2 / b_j, \quad h_k = d_1 h_1 / c_k$$

Define $\mathbf{F} = \mathrm{diag}(f_1, \ldots, f_n),\ \mathbf{G} = \mathrm{diag}(g_1, \ldots, g_n),\ \mathbf{H} = \mathrm{diag}(h_1, \ldots, h_n)$.
return $\mathbf{F}, \mathbf{G}, \mathbf{H}$

– The extractor $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$ defined by Algorithm 5 is directly adapted from the distinguishing function F of Narayanan et al. from [27]. There they argue that F is distinguishing with high probability if the automorphism group of $\mathcal{T}$ is trivial, meaning 1) $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$ finishes with high probability (meaning the the sampled corank-1 point is a canonical-form hint with high probability) and 2) if the automorphism group is trivial, no two points map to the same tensor in the orbit of $\mathcal{T}$. Translated to our language, we can claim that with high probability $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$ outputs a canonical form.
– The group action $(\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}$ extends to the set of hints as follows: $(\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathbf{u} = \mathbf{A}^{-1}\mathbf{u}$. If $\mathbf{u}$ is a canonical form hint for $\mathcal{T}$, then it is corank-1 and the extractor finds a canonical form. Now $(\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}(\mathbf{A}^{-1}\mathbf{u}, -, -)$ is still of rank $n-1$, so $\mathbf{A}^{-1}\mathbf{u}$ is a corank-1 point for $(\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}$. What remains to show is that $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$ and $\mathcal{E}_{canon}((\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}, \mathbf{A}^{-1}\mathbf{u})$ output the same canonical form.
Now, let's denote by $\mathbf{L}_u, \mathbf{L}_v, \mathbf{L}_w$ and $\mathbf{F}, \mathbf{G}, \mathbf{H}$ the matrices obtained when running $\mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$, and by $\mathbf{L}'_u, \mathbf{L}'_v, \mathbf{L}'_w$ and $\mathbf{F}', \mathbf{G}', \mathbf{H}'$ the matrices obtained when running $\mathcal{E}_{canon}((\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}, \mathbf{A}^{-1}\mathbf{u})$. We have

$$((\mathbf{L}_u\mathbf{F}, \mathbf{L}_v\mathbf{G}, \mathbf{L}_w\mathbf{H}), \mathcal{T}_C) \leftarrow \mathcal{E}_{canon}(\mathcal{T}, \mathbf{u})$$
$$((\mathbf{L}'_u\mathbf{F}', \mathbf{L}'_v\mathbf{G}', \mathbf{L}'_w\mathbf{H}'), \mathcal{T}'_C) \leftarrow \mathcal{E}_{canon}((\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}, \mathbf{A}^{-1}\mathbf{u})$$

Let $\mathbf{u}_0 = \mathbf{u}$. Since $(\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}(\mathbf{A}^{-1}\mathbf{u}_0, -, -) = \mathcal{T}(\mathbf{u}_0, \mathbf{B}(-), \mathbf{C}(-))$ is of rank $n-1$, there exists a unique $\hat{\mathbf{v}}'_0 \in \mathbb{P}(\mathbb{F}_q^n)$ such that identically

$$\mathcal{T}(\mathbf{u}_0, \mathbf{B}\mathbf{v}'_0, \mathbf{C}(-)) = 0.$$

Similarly there exists a unique $\hat{\mathbf{v}}_0 \in \mathbb{P}(\mathbb{F}_q^n)$ such that identically

$$\mathcal{T}(\mathbf{u}_0, \mathbf{v}_0, -) = 0.$$

By uniqueness, we have $\hat{\mathbf{v}}_0 = \hat{\mathbf{B}\mathbf{v}'_0}$.
Similarly, we find that there exist unique $\hat{\mathbf{w}}_0, \hat{\mathbf{w}}'_0 \in \mathbb{P}(\mathbb{F}_q^n)$ such that

$$\mathcal{T}(-, \mathbf{B}\mathbf{v}'_0, \mathbf{C}\mathbf{w}'_0) = 0, \text{ and}$$

$$\mathcal{T}(-, \mathbf{v}_0, \mathbf{w}_0) = 0.$$

Thus, as before, we conclude $\hat{\mathbf{w}}_0 = \hat{\mathbf{C}\mathbf{w}'_0}$.
Proceeding in this way, we can form matrices $\hat{\mathbf{L}}_u, \hat{\mathbf{L}}_v, \hat{\mathbf{L}}_w$ and $\hat{\mathbf{L}}'_u, \hat{\mathbf{L}}'_v, \hat{\mathbf{L}}'_w$ such that

$$\hat{\mathbf{L}}'_u = \mathbf{A}^{-1}\hat{\mathbf{L}}_u, \ \hat{\mathbf{L}}'_v = \mathbf{B}^{-1}\hat{\mathbf{L}}_v, \ \hat{\mathbf{L}}'_w = \mathbf{C}^{-1}\hat{\mathbf{L}}_w \tag{3}$$

where we consider the columns of the matrices as points in $\mathbb{P}(\mathbb{F}_q^n)$.
Now, the goal of the diagonal matrices $\mathbf{F}, \mathbf{G}, \mathbf{H}$ and $\mathbf{F}', \mathbf{G}', \mathbf{H}'$ is to scale, in order to remove the ambiguity given rise by the different representatives of the columns in the matrices $\hat{\mathbf{L}}_u, \hat{\mathbf{L}}_v, \hat{\mathbf{L}}_w$ and $\hat{\mathbf{L}}'_u, \hat{\mathbf{L}}'_v, \hat{\mathbf{L}}'_w$.

This is done by making sure that certain entries of the resulting tensor (corresponding to the first assignment in Scale()) are set to 1, regardless of the tensor (for proof see [27]). Thus we can continue in terms of projective points, assuming the scaling is done right:

$$\begin{aligned}\mathcal{T}'_C &= (\hat{\mathbf{L}}'_u, \hat{\mathbf{L}}'_v, \hat{\mathbf{L}}'_w)((\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}) \\ &= (\mathbf{A}^{-1}\hat{\mathbf{L}}_u, \mathbf{B}^{-1}\hat{\mathbf{L}}_v, \mathbf{C}^{-1}\hat{\mathbf{L}}_w)((\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}) = (\hat{\mathbf{L}}_u, \hat{\mathbf{L}}_v, \hat{\mathbf{L}}_w) \star \mathcal{T} = \mathcal{T}'_C\end{aligned}$$

Hence, the canonical forms are the same.

5 The miniMEDS Σ-protocol

Theorem 3 implies that we can use Narayanan's invariant to instantiate the Canonical Form GMW protocol for the TI relation. For completeness we state the following theorem.

Theorem 4. *The protocol from Algorithm 8 is a Σ-protocol for the TI relation.*

From here, we can construct a signature scheme using the Fiat-Shamir transform. We call the new signature scheme miniMEDS, and we refer to Algorithm 8 as the miniMEDS basic Σ-protocol.

The reason why we say the *basic* protocol, is that this protocol can be further optimized for signature size or computation time. Certainly, there exist a number of generic trade-offs between signature size, public key size, and computation time for the GMW protocol widely used in known proposals, and they can also be applied here. However, our goal for the remaining of the paper is to investigate to which extent we can reduce the runtime of Algorithm 8. The two main procedures that visibly add to the runtime are SampleCorank1 and Corank1Walk. The Scale procedure is not a bottleneck, so we will omit it from our consideration.. We discuss these in more details next.

5.1 Cutting short the walk

The first we will consider is reducing the cost of Corank1Walk. The walk consist of $3n-1$ steps in each of which we have to solve a linear system. The result is a full isometry to a tensor in $\mathfrak{T}_n$ while in essence, we only have to prove knowledge of $\mathbf{u}$ and $\mathbf{A}^{-1}\mathbf{u}$. It turns out that we can still prove this knowledge by doing only 3ℓ steps with $\ell \leq n$ and committing a tensor in $\mathfrak{T}_\ell$.

Definition 7. *Given integers q, n, ℓ we define*

$$\mathsf{CF}_\ell : \mathfrak{T}_n \times \mathbb{P}(\mathbb{F}_q^n) \to \mathfrak{T}_\ell / D_\ell(q)^3 \cup \{\perp\}$$

where $D_\ell(q)$ is the group of invertible diagonal $\ell \times \ell$ matrices. Given a tensor $\mathcal{T}$ and projective point $\mathbf{u}_1$, lift $\mathbf{u}_1$ to a representative and extend it to a corank-1

Algorithm 8 miniMEDS Σ-protocol

Public Data

$q, n \in \mathbb{N}$ $\mathbf{A}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$,
$\mathbf{B}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$,
$\mathbf{C}_0 = \mathbf{I}_n \in \mathrm{GL}_n(q)$.

II. Commit(pk)

1. **Repeat**
   (a) $\mathbf{u}_1 \xleftarrow{\$} \mathsf{SampleCorank1}(\mathcal{T}_0)$
   (b) $(\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W) \leftarrow \mathsf{Corank1Walk}(\mathcal{T}_0, \mathbf{u}_1)$
   (c) $(\mathbf{F}, \mathbf{G}, \mathbf{H}) \leftarrow \mathsf{Scale}((\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W) \star \mathcal{T}_0)$
   Until $(\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W) \neq \perp$ and $(\mathbf{F}, \mathbf{G}, \mathbf{H}) \neq \perp$.
2. Set $\mathcal{T}_C \leftarrow (\mathbf{L}_U \cdot \mathbf{F}, \mathbf{L}_V \cdot \mathbf{G}, \mathbf{L}_W \cdot \mathbf{H}) \star \mathcal{T}_0$
3. $\mathsf{cmt} \leftarrow \mathcal{T}_C$
4. Return cmt

IV. Response(sk, pk, cmt, ch)

1. $\mathsf{resp} \leftarrow \mathbf{A}_{\mathsf{ch}}^{-1} \cdot \mathbf{u}_1$
2. Return resp.

I. Keygen()

1. $\mathcal{T}_0 \xleftarrow{\$} \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n$.
2. $(\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1) \xleftarrow{\$} \mathrm{GL}_n(q) \times \mathrm{GL}_n(q) \times \mathrm{GL}_n(q)$.
3. Set $\mathcal{T}_1 \leftarrow (\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1) \star \mathcal{T}_0$.
4. Set $\mathsf{sk} \leftarrow (\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1)$, $\mathsf{pk} = (\mathcal{T}_0, \mathcal{T}_1)$.
5. Return (sk, pk).

III. Challenge()

1. $\mathsf{ch} \xleftarrow{\$} \{0, 1\}$.
2. Return ch.

V. Verify(pk, cmt, ch, resp)

1. Parse resp into $\mathbf{u}'$.
2. **Repeat**
   (a) $(\mathbf{L}'_U, \mathbf{L}'_V, \mathbf{L}'_W) \leftarrow \mathsf{Corank1Walk}(\mathcal{T}_{\mathsf{ch}}, \mathbf{u}')$
   (b) $(\mathbf{F}', \mathbf{G}', \mathbf{H}') \leftarrow \mathsf{Scale}((\mathbf{L}'_U, \mathbf{L}'_V, \mathbf{L}'_W) \star \mathcal{T}_{\mathsf{ch}})$
   Until $(\mathbf{L}'_U, \mathbf{L}'_V, \mathbf{L}'_W) \neq \perp$ and $(\mathbf{F}', \mathbf{G}', \mathbf{H}') \neq \perp$.
3. Set $\mathsf{cmt}' \leftarrow (\mathbf{L}'_U \cdot \mathbf{F}', \mathbf{L}'_V \cdot \mathbf{G}', \mathbf{L}'_W \cdot \mathbf{H}') \star \mathcal{T}_{\mathsf{ch}}$
   If $\mathsf{cmt}' = \perp$, return "reject".
4. If $\mathsf{cmt}' = \mathsf{cmt}$, return "accept". Otherwise, return "reject".

walk $(\mathbf{u}_1 = \mathbf{u}, \mathbf{v}_1, \mathbf{w}_1, \mathbf{u}_2, \ldots, \mathbf{w}_\ell)$ *using* Corank1Walk *but stopping after* 3ℓ *steps. If* Corank1Walk *fails, return* $\perp$. *Now similar as before, construct the* $n \times \ell$ *matrices*

$$\mathbf{L}_u = \left[\mathbf{u}_1 \cdots \mathbf{u}_\ell\right], \mathbf{L}_v = \left[\mathbf{v}_1 \cdots \mathbf{v}_\ell\right], \mathbf{L}_w = \left[\mathbf{w}_1 \cdots \mathbf{w}_\ell\right].$$

Return $(\mathbf{L}_u, \mathbf{L}_v, \mathbf{L}_w) \star \mathcal{T}$.[4]

In this canonical form definition we forego the scaling problem by quotienting the action of all invertible diagonal matrices on the resulting tensor. One way to (uniquely) pick such representatives is by using the Scale algorithm.

Note that the projective points not leading to $\perp$ are now almost a canonical-form hint for $\mathcal{T} \in \mathfrak{T}_n$. Importantly, note that indeed $\mathsf{CF}_\ell(\mathcal{T}, \mathbf{u}) = \mathsf{CF}_\ell(\phi \star \mathcal{T}, \phi^{-1} \star \mathbf{u})$ for $\phi \in \mathrm{GL}_n(q)$[3]. The only thing that is different is that CF_ℓ does not send $\mathcal{T}$

[4] This is not a group action anymore since $n \times \ell$ matrices are not invertible in general.

to an element in its orbit. Therefore, we cannot immediately apply Theorem 2 and need to show soundness. It turns out that to show soundness, we need the public keys to have an additional property which, depending on the size of ℓ, happens with an overwhelming probability.

Definition 8. *Let $\mathcal{T} \in \mathfrak{T}_n$ be a tensor. We define $\mathcal{T}$ to be* ℓ-collision resistant *if*

$$\mathsf{CF}_\ell(\mathcal{T}, \mathbf{u}) = \mathsf{CF}_\ell(\mathcal{T}, \mathbf{u}') \implies \mathbf{u} = \mathbf{u}' \text{ or } \mathsf{CF}_\ell(\mathcal{T}, \mathbf{u}) = \bot$$

This property is preserved under the group action of $\mathrm{GL}_n(q)^3$.

Theorem 5. *The miniMEDS protocol with parameters $q, n, \ell \geq 3$ (Corank1 walk with ℓ vectors) is special sound for ℓ-collision resistant public keys.*

Proof. Let $\mathcal{T}_0, \mathcal{T}_1 \in \mathbb{F}_q^{n \times n \times n}$ be the public key with private key $\phi \in \mathrm{GL}_n(q)^3$ so that $\phi \star \mathcal{T}_0 = \mathcal{T}_1$. Given two valid related transcripts $(\mathsf{cmt}, 0, \mathbf{u})$ and $(\mathsf{cmt}, 1, \mathbf{u}')$ for the same comitment $\mathsf{cmt} = \hat{\mathcal{T}}$ we have that

$$\mathsf{CF}_\ell(\mathcal{T}_0, \mathbf{u}) = \mathsf{CF}_\ell(\mathcal{T}_1, \mathbf{u}') \neq \bot$$

where $\hat{\mathcal{T}}$ is now simply a representative of this. By $\mathrm{GL}_n(q)^3$-equivariance we find $\mathsf{CF}_\ell(\mathcal{T}_0, \mathbf{u}) = \mathsf{CF}_\ell(\mathcal{T}_0, \phi \star \mathbf{u}')$. Therefore, in the case of ℓ-collision resistance, we find $(\mathbf{u}_1, \ldots, \mathbf{w}_\ell) = (\mathbf{u}'_1, \ldots, \mathbf{w}'_\ell)$. From this, we obtain at least 3 collisions $(\mathbf{u}_1, \ldots, \mathbf{u}_3) \mapsto (\mathbf{u}'_1, \ldots, \mathbf{u}'_3)$ and hence w.h.p. two codeword collisions of full rank. Then, as shown in [9] we can solve MCE and extract the witness in poly-time.

5.2 Weak keys

Since we only prove security for ℓ-collision resistant keys, the ones that are not are considered *weak keys*. To use the scheme in practice, we need to limit the amount of weak keys. Unfortunately, this is a bit too strong, and we are unable to approximate the probability that a tensor is *ℓ-collision resistant*. So instead we use a weaker variant which covers most cases.

Definition 9. *We call a tensor $\mathcal{T} \in \mathfrak{T}_n$* weak ℓ-collision resistant *if*

$$\begin{aligned}\mathsf{CF}_\ell(\mathcal{T}, \mathbf{u}) = \mathsf{CF}_\ell(\mathcal{T}, \mathbf{u}') \implies & U \cap U' \neq \emptyset \text{ or } V \cap V' \neq \emptyset \\ & \text{or } W \cap W' \neq \emptyset \text{ or } \mathsf{CF}_\ell(\mathcal{T}, \mathbf{u}) = \bot\end{aligned}$$

where $U = \langle \mathbf{u}_1, \ldots, \mathbf{u}_\ell \rangle$ and V, W, U', V', W' are defined similarly.

Lemma 1. *Let q, n, ℓ be integers, then we have the following probability*

$$\mathbb{P}_{\mathcal{T} \in \mathfrak{T}}(\mathcal{T} \text{ is not } \textit{weak } \ell\textit{-collision resistant}) \leq q^{2n-4-(\ell^3-3\ell^2+\ell)}.$$

Proof. See Appendix A.

Table 1. Weak key probability for the different parameter sets of miniMEDS. In all cases, $\ell = 6$ was optimal.

SL	q	n	ℓ	weak key prob.
I	4093	25	6	2^{-816}
III	4093	34	6	2^{-600}
V	4093	44	6	2^{-360}

Remark 3. Most pairs of walks are linear independent by dimension analysis, and it seems unlikely that linear dependent walks have a much higher probability of having the same canonical form. However, the problem to get a bound is that the resulting constraints on $\mathcal{T}$ might not be linear independent anymore. Still, we believe that the amount of keys that suffer this is negligible. Therefore, we leave it as a conjecture and open problem that this is indeed the case.

Conjecture 1. Let q, n, ℓ be integers

$$\mathbb{P}_{\mathcal{T}\in\mathfrak{T}}(\mathcal{T} \text{ is a weak key}) \leq q^{2n-4-(\ell^3-3\ell^2+\ell)}.$$

The weak key probability for the different parameter sets are given in Table 1.

Remark 4. In the case of weak ℓ-collision resistance, a forgery only implies that one of the intersections of their supports must be non-empty. We can still solve MCE by constructing the supports (X, Y, Z) and (X', Y', Z') corank-1 walks for x and x' and enumerating all pairs $X \times X' \cup Y \times Y' \cup Z \times Z'$. For the correct pair we can solve MCE by solving 1-collision MCE. However, this approach is exponential in $\ell \approx \mathcal{O}(\sqrt[3]{n})$ and does not give us a polynomial-time reduction.

5.3 Sampling Random Corank-1 Point

There are several approaches to obtain a random corank-1 point of a tensor $\mathcal{T} \in \mathbb{F}_q^{n\times n\times n}$. The simplest approach is to sample a random vector $\mathbf{u} \in \mathbb{F}_q^n$ and check whether it is a corank-1 point, as we did in Theorem 3. This results in a sampler of expected cost of $O(qn^3)$. Since in our parameter regime $q \gg n^2$, this approach results in a computational cost in the order of roughly $O(n^5)$, which is prohibitively expensive for use in practice.

Constructing corank-1 points. To avoid the $O(qn^3)$ cost of repetitive random resampling, we use a constructive approach to compute a corank-1 vector directly: We fix randomly all but one coordinate $\mathbf{u}_{i^\star}$ of $\mathbf{u}$ and compute the coordinate $\mathbf{u}_{i^\star}$ such that $\mathcal{T}(\mathbf{u}, -, -)$ has corank-1. First note that for $\mathbf{B} = \sum_{i\neq i^\star} u_i \mathcal{T}(\mathbf{e}_i, -, -)$. we can write $\mathcal{T}(\mathbf{u}, -, -) = \mathbf{B} + u_{i^\star}\mathcal{T}(\mathbf{e}_{i^\star}, -, -)$. With all but one coordinate $u_{i^\star}$ of u chosen randomly and fixed. We assume that slice $\mathcal{T}(\mathbf{e}_{i^\star}, -, -)$ is invertible and normalize it to the identity matrix. This normalization is part of the public key generation and is used to reduce the key size. Then the matrix $\mathcal{T}(\mathbf{u}, -, -) = \mathbf{B} + u_{i^\star}\mathbf{I}$. has corank 1, and we thus have $\det(\mathbf{B} + u_{i^\star}\mathbf{I}) = 0$.

Algorithm 9 BSGS–FLV sampler for a random corank-1 point

Require: Tensor $\mathcal{T} \in \mathbb{F}_q^{n \times n}$ with $\mathcal{T}(\mathbf{e}_{i^\star}, -, -) = \mathbf{I}$
Ensure: A vector $\mathbf{u} \in \mathbb{F}_q^n$ such that $\mathcal{T}(\mathbf{u}, -, -)$ has corank 1
while true **do**
  Sample $u_i \xleftarrow{\$} \mathbb{F}_q$ for all $i \neq i^\star$
  Compute $\mathbf{B} \leftarrow \sum_{i \neq i^\star} u_i \mathcal{T}(\mathbf{e}_i, -, -)$
  Compute $g(t) = \det(\mathbf{B} + t\mathbf{I}) \in \mathbb{F}_q[t]$ (using BSGS Faddeev–Leverrier algorithm)
  compute the root set $R \leftarrow \{\, t \in \mathbb{F}_q : g(t) = 0 \,\}$
  (using a constant-time exhaustive scan with forward differences)
  compute the valid root set $R' \leftarrow \{\, t \in R : \mathrm{rank}(\mathbf{B} + t\mathbf{I}) = n - 1 \,\}$
  (using constant-time Gaussian elimination)
  if $R' \neq \emptyset$ **then**
    select $t \xleftarrow{\$} R'$ uniformly at random in constant time
    set $\mathbf{u}_{i^\star} \leftarrow t$
    return u

Thus, finding a suitable value for $u_{i^\star}$ reduces to finding a root of the univariate polynomial

$$g(t) = \det(\mathbf{B} + t\mathbf{I}) \in \mathbb{F}_q[t].$$

The polynomial $g(t)$ is (up to a sign change in the variable) the characteristic polynomial of $\mathbf{B}$. The roots of $g(t)$ correspond to the negated eigenvalues of $\mathbf{B}$. In particular, a root t of $g(t)$ corresponds to a value $\mathbf{u}_{i^\star} = t$ such that $\mathbf{B} + t\mathbf{I}$ has corank at least one. A natural approach is therefore to perform a root search over $\mathbb{F}_q$. For each root t, we test whether the matrix $\mathbf{B} + t\mathbf{I}$ has corank exactly one using Gaussian elimination. We then select one of the valid values uniformly at random. Algorithm 9 shows the overall algorithm.

While the naive sampling approach usually fails by yielding a vector $\mathbf{u}$ for which $\mathcal{T}(\mathbf{u}, -, -)$ is full rank and only succeeds with probability about $1/q$ in producing a corank-1 vector, the constructive approach always produces a vector $\mathbf{u}$ such that $\mathcal{T}(\mathbf{u}, -, -)$ has corank at least one and only fails with probability about $1/q$ when the corank exceeds one. Therefore, the success probability is inverted: instead of repeating until a rare success occurs, we only occasionally need to repeat in the case of a rare failure.

More notably, statistically, these sample methods are indistinguishable. The first, rejection sampling method effectively is indistinguishable from sampling uniformly at random from the set of all corank-1 points. This is the same as sampling a linear combination of the slices $\mathcal{T}(\mathbf{e}_i, -, -)$ uniformly from the set of all linear combinations yielding a corank-1 matrix. In turn, this is what we effectively have in the case of the constructive sampler. A difference might occur if the corank is bigger than 1. But this case will be rejected in the constructive sampler, thus effectively yielding the same distribution.

Computing $g(t)$. We consider two representative and practically relevant approaches for computing the polynomial $g(t)$:

Krylov method. One option is to use a Krylov subspace method to compute the minimal polynomial of $\mathbf{B} = \sum_{i \neq i^\star} u_i \mathcal{T}(\mathbf{e}_i, -, -)$, as described in [18, Al-

gorithm 2.1]. This approach has complexity $O(n^3)$ and is deterministic apart from the initial choice of a random vector. However, the degree of the minimal polynomial is not fixed in general. To ensure correctness, one must either restart the computation when the degree is too small or compute the full characteristic polynomial instead. Both options introduce complications: restarts lead to variable running time, while switching to characteristic polynomial introduces control flow that depends on intermediate, data-dependent results.

BSGS Faddeev–Leverrier algorithm. Alternatively, one can compute the characteristic polynomial directly using a baby-step giant-step variant of the Faddeev–Leverrier algorithm (after Urbain Le Verrier), originally due to Preparata and Sarwate [31] in the form described by Johansson [24, Algorithm 2]. This algorithm computes the characteristic polynomial via a structured recurrence based on matrix powers and trace identities. This approach has complexity $O(n^{3.5})$ and deterministically computes the full characteristic polynomial without requiring restarts or degree checks.

Although the Krylov-based method is asymptotically faster, its variable-degree output and the need for restarts or conditional handling make it less suitable for constant-time implementations. In contrast, the BSGS Faddeev–Leverrier algorithm has a fixed control flow and does not depend on data-dependent branching, making it significantly easier to implement in a constant-time manner. For this reason, we use the BSGS Faddeev–Leverrier algorithm.

Root finding via forward differences. Once the polynomial $g(t) \in \mathbb{F}_q[t]$ has been computed, a straightforward approach is to evaluate $g(t)$ sequentially until a root is found. However, this approach would lead to data-dependent running time, since the number and position of roots of g depend on the matrix $\mathbf{B}$ and thus may leak information about the underlying vector $\mathbf{u}$.

To avoid this timing leakage, we evaluate $g(t)$ for every $t \in \mathbb{F}_q$. We then select one of the roots yielding corank-1 uniformly at random in a constant-time manner. This ensures a fixed control flow and constant-time execution of the root-finding procedure. This requires evaluating $g(t)$ at all $t \in \mathbb{F}_q$. One way of doing this is using Horner's rule which requires $O(qn)$ field operations (since $\deg g \leq n$), dominated by multiplications.

To reduce the number of field multiplications, we instead evaluate g over all field elements using forward finite differences: for a polynomial g, define the forward difference operator $\Delta g(t) = g(t+1) - g(t)$. Each application of Δ lowers the degree by one. Hence, if $\deg g \leq n$, then $\Delta^n g(t)$ is constant. Thus, after precomputing the values $\Delta^0 g(0), \Delta^1 g(0), \ldots, \Delta^n g(0)$, the values of $g(t)$ for $t = 0, 1, \ldots, q-1$ can be generated iteratively using only additions via $\Delta^k g(t+1) = \Delta^k g(t) + \Delta^{k+1} g(t)$, for $k = 0, \ldots, n-1$.

This yields an exhaustive root search over $\mathbb{F}_q$ in which each step requires only field additions, replacing most multiplications of Horner evaluation and significantly reducing the cost of root finding in practice. In our implementation, this optimization makes the root-finding step several times faster than Horner evaluation and reduces its contribution to the overall runtime substantially.

Cost of the constructive sampler. The sampler is dominated by the following steps. The computation of the characteristic polynomial of $\mathbf{B}$ using the BSGS Faddeev–Leverrier algorithm requires $O(n^{3.5})$ operations. The subsequent root-finding step for evaluating the polynomial $g(t)$ over all $t \in \mathbb{F}_q$ using forward finite differences costs $O(q)$ field operations. Testing all candidate roots $t \in R$ for the corank-1 condition requires at most n rank computations of cost $O(n^3)$, for a total of $O(n^4)$. Thus, summing the individual contributions, the cost per iteration of the sampler is $O(n^3 + n^{3.5} + q + n^4) = O(n^4 + q)$. One iteration fails only when the resulting matrix has corank greater than one, which occurs with probability about $1/q$. Hence, the expected number of iterations is $\frac{1}{1-1/q} = 1 + O(1/q)$, and the overall expected cost of the constructive approach remains $O(n^4 + q)$. Compared to the naive approach with expected cost $O(qn^3)$, in our parameter regime where $q \gg n^2$, this results in a substantial improvement.

6 Implementation and Performance Evaluation

6.1 Constant-Time Implementation Considerations

As a baseline for a secure implementation, we aim at preventing timing side-channel leakage based on secret-dependent control flow and memory access patterns (other types of leakage are beyond the scope of this work).

All routines are fairly standard to protect against timing leakage, but we need to be careful when performing constant-time Gaussian elimination. This is necessary during the generation of a random corank-1 point using BSGS-FLV and during the corank-1 walk computing $(\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W)$. Gaussian elimination in constant time can be very costly since one needs to hide the pivot positions. This becomes cheaper when one restricts pivot positions to the diagonal. However, restricting pivot positions and aborting otherwise introduces a bias in the distribution of accepted instances. In particular, this restricts the walk to a subset of all valid walks, which may affect the statistical properties of the scheme.

To perform the simple corank-1 checks during BSGS-FLV without introducing a bias, we use a trick from the Wave signature scheme by computing an extended Gaussian elimination [3]: We append a zero row to the matrix and then perform constant-time Gaussian elimination while always performing a swap if the current pivot position is zero. This swaps in the additional last row for a column without pivot so that the following pivots are again on the diagonal. By counting the number of zero pivots on the diagonal, we can detect whether the matrix has corank one.

For the rank check during the corank-1 walk together with kernel computation, we apply a blinding transformation to the public tensor $\mathcal{T}_0$ and the secret vector $\mathbf{u}_i$. This transformation corresponds to a change of basis and preserves the corank-1 structure of the contractions. The blinding ensures that any restrictions imposed on pivot positions do not bias the distribution of the resulting walk in a way that depends on the secret vector $\mathbf{u}_i$. This independence is necessary for the Zero-Knowledge property of the resulting constant-time implementation.

The signing procedure includes re-attempts, e.g., when Algorithm 6 or Algorithm 7 fail. However, these loops depend only on randomness derived from the public tensor $\mathcal{T}_0$ and fresh randomness $\mathbf{u}_i$ but not on secret key material or other long-term secrets. Therefore, a varying number of iterations here does not leak secret information.

The final shape of the three algorithms of miniMEDS is given in Fig. 1. The algorithms incorporate the practical considerations from Section 5.1, Section 5.3 and Section 6.1 on shorter Corank-1 walk, efficient Corank-1 sampler, and blinding for efficient constant-time Gaussian elimination. We do not consider generic trade-offs between signature size, public key size, and computation time known from literature, as our goal is to highlight the benefits of the Canonical-form hinted construction. Instead, we discuss them in the next subsection.

6.2 Generic trade-offs applicable to miniMEDS and the Canonical-form GMW protocol

The presented instantiation prioritizes conceptual simplicity in order to highlight the underlying corank-1 construction. A number of standard techniques from MPC-in-the-head style signatures can be applied to trade signature size, public key size, and computational cost.

For example, using a fixed-weight challenge together with seed compression allows one to store short seeds instead of explicit vectors for the challenge responses, reducing the signature size by approximately 25% at Level 1, at the cost of additional encoding complexity, seed expansion, and sign/verify computation time. Furthermore, one may generalize the binary challenge space to an s-ary challenge, which requires introducing multiple public tensors $\mathcal{T}_0, \ldots, \mathcal{T}_{s-1}$. This reduces the number of repetitions t and hence signature size and sign/verify computation cost, while increasing the public key size linearly in s. Additional techniques such as seed trees combined with low-weight challenges can further reduce communication, at the cost of increased computational effort.

We emphasize that these trade-offs are orthogonal to the proposed construction. Since the main goal of this work is to present and analyze the corank-1 scheme, we leave a systematic exploration of these trade-offs to future work. To show how miniMEDS relates to MEDS in terms of public key and signature size, we provide Table 2 that compares signature sizes to the MEDS variant presented in [9] with the same security parameters using fixed low-weight challenges and seed trees. Note that we do not recommend to use seed trees or low-weight challenged for miniMEDS due to the expected impact on signing and verification time. Also note that the use of a seeds in the signature of miniMEDS requires changes to the sign and verify operations with additional impact on signature size or sign/verification time.

6.3 Evaluation

The complete implementation is available at: `https://github.com/IIS-ACE-lab/MiniMEDS`

Algorithm 10 KeyGen

Require: Security parameters (n, ℓ, t, q)
Ensure: Public key $\mathsf{pk} = (\mathcal{T}_0, \mathcal{T}_1)$, secret key $\mathsf{sk} = \mathbf{A}_1^{-1}$

$\text{seed}_{\mathcal{T}_0} \xleftarrow{\$} \mathbb{F}_2^t$
$\mathcal{T}_0 \xleftarrow{\$} \text{seed}_{\mathcal{T}_0}\mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n$
$(\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1) \xleftarrow{\$} \mathrm{GL}_n(q) \times \mathrm{GL}_n(q) \times \mathrm{GL}_n(q)$
$\mathcal{T}_1 \leftarrow (\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1) \star \mathcal{T}_0$
$\mathsf{pk} = (\text{seed}_{\mathcal{T}_0}, \mathcal{T}_1), \quad \mathsf{sk} \leftarrow \mathbf{A}_1^{-1}$
return $(\mathsf{pk}, \mathsf{sk})$

Algorithm 11 $\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}, m)$

Require: $\mathsf{sk} = \mathbf{A}_1^{-1}$, $\mathsf{pk} = (\text{seed}_{\mathcal{T}_0}, \mathcal{T}_1)$, message m
Ensure: Signature $\sigma = (h, \mu_0, \ldots, \mu_{t-1})$

$\mathcal{T}_0 \xleftarrow{\$} \text{seed}_{\mathcal{T}_0}\mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n$
$(\mathbf{A}, \mathbf{B}, \mathbf{C}) \xleftarrow{\$} \mathrm{GL}_n(q) \times \mathrm{GL}_n(q) \times \mathrm{GL}_n(q)$
$\mathcal{B}_0 \leftarrow (\mathbf{A}, \mathbf{B}, \mathbf{C}) \star \mathcal{T}_0$ ▷ blinded tensor
for $i = 0$ to $t - 1$ **do**
  $\mathbf{u}_i \leftarrow \mathsf{BSGS_FLV}(\mathcal{T}_0)$ ▷ random corank-1 point
  $\tilde{\mathbf{u}}_i = \mathbf{A}^{-1}\mathbf{u}_i$ ▷ map to blinded coordinates
  $(\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W) \leftarrow \mathsf{Corank1Walk}(\mathcal{B}_0, \tilde{\mathbf{u}}_i, \ell)$
  $\mathcal{R}_i \leftarrow (\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W) \star_{n\to\ell} \mathcal{B}_0$ ▷ $\mathrm{O}(n^3 m)$
  $(\mathbf{f}_i, \mathbf{g}_i, \mathbf{h}_i) \leftarrow \mathsf{Scale}(\mathcal{R}_i)$
  $\mathcal{T}_i \leftarrow (\mathrm{diag}(\mathbf{f}_i), \mathrm{diag}(\mathbf{g}_i), \mathrm{diag}(\mathbf{h}_i)) \star_{\ell\to\ell} \mathcal{R}_i$ ▷ $\mathrm{O}(m^3)$
$d \leftarrow \mathsf{Hash}(\mathcal{T}_0 \,|\, \mathcal{T}_1 \,|\, \ldots \,|\, \mathcal{T}_{t-1} \,|\, m)$ ▷ $d = (d_0, \ldots, d_{t-1})$
for $i = 0$ to $t - 1$ **do**
  if $d_i = 0$ **then**
    $\mu_i \leftarrow \mathbf{u}_i$
  else
    $\mu_i \leftarrow \mathbf{A}_1^{-1}\mathbf{u}_i$
return $\sigma = (d, \mu_0, \ldots, \mu_{t-1})$

Algorithm 12 $\mathsf{Verify}(\mathsf{pk}, m, \sigma)$

Require: $\mathsf{pk} = (\text{seed}_{\mathcal{T}_0}, \mathcal{T}_1)$, message m, signature $\sigma = (d, \mu_0, \ldots, \mu_{t-1})$
Ensure: Accept or Reject

$\mathcal{T}_0 \xleftarrow{\$} \text{seed}_{\mathcal{T}_0}\mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n$
for $i = 0$ to $t - 1$ **do**
  $\mathcal{T}^{(i)} \leftarrow \mathcal{T}_{d_i}$
  $(\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W) \leftarrow \mathsf{Corank1Walk}(\mathcal{T}^{(i)}, \mu_i)$
  $\mathcal{R}_i \leftarrow (\mathbf{L}_U, \mathbf{L}_V, \mathbf{L}_W) \star_{n\to\ell} \mathcal{T}^{(i)}$ ▷ $\mathrm{O}(n^3 m)$
  $(\mathbf{f}_i, \mathbf{g}_i, \mathbf{h}_i) \leftarrow \mathsf{Scale}(\mathcal{R}_i)$
  $\mathcal{T}_i \leftarrow (\mathrm{diag}(\mathbf{f}_i), \mathrm{diag}(\mathbf{g}_i), \mathrm{diag}(\mathbf{h}_i)) \star_{\ell\to\ell} \mathcal{R}_i$ ▷ $\mathrm{O}(m^3)$
$d' \leftarrow \mathsf{Hash}(\mathcal{C}_0 \,|\, \mathcal{C}_1 \,|\, \ldots \,|\, \mathcal{C}_{t-1} \,|\, m)$
if $d' = d$ **then**
  return Accept
else
  return Reject

Fig. 1. The miniMEDS signature scheme with efficient Corank-1 sampler, blinding for efficient constant-time Gaussian elimination and shorter Corank-1 walk. Standard generic trade-offs for GMW signatures are not included. The added reference to ℓ in Corank1Walk and as a subscript in $\star_{n\to\ell}$ and $\star_{\ell\to\ell}$ is to indicate the shortened walk to ℓ steps.

Table 2. Comparison of miniMEDS to previous MEDS constructions using the same security parameters and the same level of generic trade-offs — pk compression, fixed weight challenges, and seed tree.

NIST	q	n	s	t	w	pk (B)	sig (B)
miniMEDS (no generic trade-offs)							
1	4093	25	2	128	$\mathbb{E}[w] = t/2$	23 630	4 816
3	4093	34	2	192	$\mathbb{E}[w] = t/2$	59 244	9 816
5	4093	44	2	256	$\mathbb{E}[w] = t/2$	128 160	16 928
miniMEDS (fixed-weight challenges, pk compression)							
1	4093	25	2	132	66	22 532	3 531
3	4093	34	2	198	99	57 254	7 425
5	4093	44	2	262	131	124 904	12 838
miniMEDS (seed tree, fixed low-weight challenges, pk compression)							
1	4093	25	2	864	19	22 532	2 514
3	4093	34	2	928	31	57 254	5 389
5	4093	44	2	1008	44	124 904	9 240
MEDS [9] (seed tree, fixed low-weight challenges, pk compression)							
1	4093	25	2	864	19	21 595	3 217
3	4093	34	2	928	31	55 520	6 970
5	4093	44	2	1008	44	122 000	12 144

We evaluate its performance on an **AMD Ryzen Threadripper PRO 9975WX** CPU. All experiments were conducted on a single isolated core using `taskset`, with the CPU frequency governor set to `performance` and turbo boost disabled to ensure stable measurements. All code was compiled using `gcc` with platform-specific optimizations enabled using flags `-O3 -march=native -fno-plt -funroll-loops -flto`.

Cycle and instruction counts were obtained using `perf stat` with user-space events (`cycles:u`, `instructions:u`). Each operation was executed repeatedly within the benchmark binaries to amortize set-up and measurement overheads. Reported cycle counts and runtimes are normalized per operation. For each configuration, we performed multiple independent runs and report the median.

We consider parameter sets corresponding to NIST security levels 1, 3, and 5 and compare our scheme to MEDS, LESS, FAEST, SLH-DSA, and ML-DSA. The performance values in Table 3 are collected from different implementations and benchmark platforms, and should therefore be interpreted as indicative rather than directly comparable. For LESS, we use the values from the LESS submission document v2. The reported performance is obtained from the reference code compiled with `-O3 -march=native -flto -ftree-vectorize -funroll-loops`, similar to our miniMEDS benchmarking setup, on a 13th Gen Intel(R) Core(TM) i7-1355U clocked at 5.0 GHz. For FAEST, the numbers are from the submission website and correspond to the AVX2 implementation on

Table 3. Comparison of miniMEDS, LESS, and FAEST public-key sizes, signature sizes, and performance (in Mcycles rounded to three significant figures).

NIST	Parameter Set	pk (B)	sig (B)	KeyGen (Mcycles)	Sign (Mcycles)	Verify (Mcycles)
			miniMEDS			
1	miniMEDS-L1	23 630	4 816	7.60	387	119
3	miniMEDS-L3	59 244	9 816	25.6	1 550	454
5	miniMEDS-L5	128 160	16 928	121	5 500	1 570
			MEDS			
1	MEDS-level1-fast	21 595	5 456	2.39	251	247
1	MEDS-level1-small	21 595	3 217	2.20	1 410	1 410
3	MEDS-level3-fast	55 520	10 786	6.67	1 010	1 010
3	MEDS-level3-small	55 520	6 970	6.67	4 510	4 560
5	MEDS-level5-fast	122 000	21 052	17.7	3 540	3 600
5	MEDS-level5-small	122 000	12 144	18.1	13 400	13 500
			LESS			
1	LESS-252-192	13 940	2 625 / 2 289	1.3	376	304
1	LESS-252-45	97 484	1 329 / 1 313	8.6	89.5	71.9
3	LESS-400-220	35 074	6 329 / 5 585	4.0	1 090	865
3	LESS-400-102	105 174	4 131 / 3 867	11.5	503	399
5	LESS-548-345	65 793	10 680 / 9 464	10.0	3 720	2 960
5	LESS-548-137	197 315	7 436 / 7 116	29.6	1 480	1 190
			FAEST			
1	FAEST-128s	32	4 506	0.005	12.8	9.78
3	FAEST-192s	48	11 260	0.011	54.7	42.3
5	FAEST-256s	48	20 696	0.013	76.3	74.5
1	FAEST-EM-128s	32	3 906	0.005	9.40	7.40
3	FAEST-EM-192s	48	9 340	0.009	39.3	36.2
5	FAEST-EM-256s	64	17 984	0.013	62.5	59.7
			SLH-DSA / SPHINCS$^+$-SHA2			
1	-128s-simple	32	7 856	85.0	645	0.861
3	-192s-simple	48	16 224	125	1 250	1.44
5	-256s-simple	64	29 792	80.9	1 030	1.99
			ML-DSA / Dilithium			
2	Dilithium2	1 312	2 420	0.124	0.333	0.118
3	Dilithium3	1 952	3 293	0.256	0.529	0.179
5	Dilithium5	2 592	4 595	0.298	0.642	0.280

an x86-64 platform with AVX2 and AES-NI, using an AMD Zen 3 Ryzen 9 5950X processor at 3.4 GHz. For SLH-DSA, we use the SPHINCS$^+$ submission document, version 3.1, reporting the AVX2 implementation compiled with optimization flag `-O3` on a 3.1 GHz Intel Xeon E3-1220 CPU (Haswell). For ML-DSA, we use the Dilithium website results for the AVX2 implementation on an Intel Core i7-6600U CPU (Skylake). For MEDS, we use the most recent parameter update and the reference implementation from [9], benchmarked on an AMD Ryzen 7 PRO 5850U.

To keep the comparison readable, Table 3 reports representative parameter sets rather than the complete trade-off space. For MEDS and LESS, we include variants that expose the main size–time and public-key/signature-size trade-offs. For FAEST we report the smaller-signature "s" variants and the corresponding EM variants. For SLH-DSA/SPHINCS$^+$ we use the SHA2 "s-simple" AVX2 benchmark points, and for ML-DSA we include the standard Dilithium2/3/5 parameter sets.

7 Future Work

Our paper demonstrates that we can reduce the signature size of GMW-based signatures by using some intrinsic properties of isomorphisms and invariants. But, a number of open questions remain that open some future avenues for research. In particular it would be interesting to apply this technique on concrete other hard isomorphism problems, for example on LESS using Bardet et al.'s recent result [4], and to compare to the already available canonical form solution from [10]. Even more unexplored is a construction from lattice isomorphism hints, or whether a similar compression exists for other protocols beyond the GMW protocol. One concrete avenue would be to apply the construction to advanced signatures (threshold, blind, ring signatures) from the MCE problem and evaluate the performance benefits and disadvantages.

References

1. Allombert, B., Biasse, J.F., Eriksen, J.K., Kutas, P., Leonardi, C., Page, A., Scheidler, R., Bagi, M.T.: PEARL-SCALLOP: Parameter extension applicable in real-life SCALLOP. Cryptology ePrint Archive, Paper 2024/1744 (2024), https://eprint.iacr.org/2024/1744
2. Baldi, M., Barenghi, A., Beckwith, L., Biasse, J., Chou, T., Esser, A., Gaj, K., Karl, P., Mohajerani, K., Pelosi, G., Persichetti, E., Saarinen, M.O., Santini, P., Wallace, R., Zweydinger, F.: LESS — Linear Equivalence Signature Scheme (2024), available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures
3. Banegas, G., Carrier, K., Chailloux, A., Couvreur, A., Debris-Alazard, T., Gaborit, P., Karpman, P., Loyer, J., Niederhagen, R., Sendrier, N., Smith, B., Tillich, J.: Wave. Tech. rep., National Institute of Standards and Technology (2023), available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures

4. Bardet, M., Brion, C., Otmani, A., Saeed, M., Sendrier, N.: Solving the linear code equivalence problem from single codeword matching. Cryptology ePrint Archive, Paper 2026/550 (2026), https://eprint.iacr.org/2026/550
5. Benčina, B., Budroni, A., Chi-Domínguez, J.J., Kulkarni, M.: Properties of lattice isomorphism as a cryptographic group action. In: Post-Quantum Cryptography: 15th International Workshop, PQCrypto 2024, Oxford, UK, June 12–14, 2024, Proceedings, Part I. p. 170–201. Springer-Verlag, Berlin, Heidelberg (2024). https://doi.org/10.1007/978-3-031-62743-9_6, https://doi.org/10.1007/978-3-031-62743-9_6
6. Budroni, A., Esser, A.: One pair to rule them all: An optimal algorithm for solving code equivalence via codeword search. Cryptology ePrint Archive, Paper 2026/268 (2026), https://eprint.iacr.org/2026/268
7. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: Csidh: An efficient post-quantum commutative group action. In: Advances in Cryptology – ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III. p. 395–427. Springer-Verlag, Berlin, Heidelberg (2018). https://doi.org/10.1007/978-3-030-03332-3_15, https://doi.org/10.1007/978-3-030-03332-3_15
8. Chou, T., Niederhagen, R., Persichetti, E., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: Take your meds: Digital signatures from matrix code equivalence. In: El Mrabet, N., De Feo, L., Duquesne, S. (eds.) Progress in Cryptology - AFRICACRYPT 2023. pp. 28–52. Springer Nature Switzerland, Cham (2023)
9. Chou, T., Niederhagen, R., Ran, L., Samardjiska, S.: Reducing signature size of matrix-code-based signature schemes. In: Saarinen, M.J., Smith-Tone, D. (eds.) Post-Quantum Cryptography. pp. 107–134. Springer Nature Switzerland, Cham (2024)
10. Chou, T., Persichetti, E., Santini, P.: On linear equivalence, canonical forms, and digital signatures. Designs, Codes and Cryptography **93**(7), 2415–2457 (2025). https://doi.org/10.1007/s10623-025-01576-1
11. Colò, L., Kohel, D.: Orienting supersingular isogeny graphs. Journal of Mathematical Cryptology **14**(1), 414–437 (2020). https://doi.org/doi:10.1515/jmc-2019-0034, https://doi.org/10.1515/jmc-2019-0034
12. Couveignes, J.M.: Hard homogeneous spaces. Cryptology ePrint Archive, Paper 2006/291 (2006), https://eprint.iacr.org/2006/291
13. Couvreur, A., Debris-Alazard, T., Gaborit, P.: On the hardness of code equivalence problems in rank metric (2021), https://arxiv.org/abs/2011.04611
14. Dachman-Soled, D., Ducas, L., Gong, H., Rossi, M.: Lwe with side information: Attacks and concrete security estimation. In: Advances in Cryptology – CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part II. p. 329–358. Springer-Verlag, Berlin, Heidelberg (2020). https://doi.org/10.1007/978-3-030-56880-1_12, https://doi.org/10.1007/978-3-030-56880-1_12
15. Dartois, P., Eriksen, J.K., Invernizzi, R., Vercauteren, F.: qt-pegasis: Simpler and faster effective class group actions. In: Daemen, J., Thomé, E. (eds.) Advances in Cryptology – EUROCRYPT 2026. pp. 636–665. Springer Nature Switzerland, Cham (2026)
16. Ducas, L., Postlethwaite, E.W., Pulles, L.N., Woerden, W.v.: Hawk: Module lip makes lattice signatures fast, compact and simple. In: Advances in Cryptology – ASIACRYPT 2022: 28th International Conference on the Theory and

Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part IV. p. 65–94. Springer-Verlag, Berlin, Heidelberg (2022). https://doi.org/10.1007/978-3-031-22972-5_3, https://doi.org/10.1007/978-3-031-22972-5_3

17. Ducas, L., van Woerden, W.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: Advances in Cryptology – EUROCRYPT 2022: 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 – June 3, 2022, Proceedings, Part III. p. 643–673. Springer-Verlag, Berlin, Heidelberg (2022). https://doi.org/10.1007/978-3-031-07082-2_23, https://doi.org/10.1007/978-3-031-07082-2_23
18. Dumas, J., Pernet, C., Wan, Z.: Efficient computation of the characteristic polynomial. In: Kauers, M. (ed.) Symbolic and Algebraic Computation, International Symposium ISSAC 2005, Beijing, China, July 24-27, 2005, Proceedings. pp. 140–147. ACM (2005). https://doi.org/10.1145/1073884.1073905, https://doi.org/10.1145/1073884.1073905
19. Esser, A., May, A., Verbel, J., Wen, W.: Partial key exposure attacks on bike, rainbow and ntru. In: Advances in Cryptology – CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part III. p. 346–375. Springer-Verlag, Berlin, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15982-4_12, https://doi.org/10.1007/978-3-031-15982-4_12
20. Faugère, J.C., Perret, L.: Polynomial equivalence problems: Algorithmic and theoretical aspects pp. 30–47 (2006)
21. Feo, L.D., Fouotsa, T.B., Kutas, P., Leroux, A., Merz, S.P., Panny, L., Wesolowski, B.: Scallop: Scaling the csi-fish. In: Public-Key Cryptography – PKC 2023: 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7–10, 2023, Proceedings, Part I. p. 345–375. Springer-Verlag, Berlin, Heidelberg (2023). https://doi.org/10.1007/978-3-031-31368-4_13, https://doi.org/10.1007/978-3-031-31368-4_13
22. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing. p. 218–229. STOC '87, Association for Computing Machinery, New York, NY, USA (1987). https://doi.org/10.1145/28395.28420, https://doi.org/10.1145/28395.28420
23. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. J. ACM **38**(3), 690–728 (Jul 1991). https://doi.org/10.1145/116825.116852, https://doi.org/10.1145/116825.116852
24. Johansson, F.: On a fast and nearly division-free algorithm for the characteristic polynomial. CoRR **abs/2011.12573** (2020), https://arxiv.org/abs/2011.12573
25. Kirshanova, E., May, A.: Decoding mceliece with a hint – secret goppa key parts reveal everything. In: Security and Cryptography for Networks: 13th International Conference, SCN 2022, Amalfi (SA), Italy, September 12–14, 2022, Proceedings. p. 3–20. Springer-Verlag, Berlin, Heidelberg (2022). https://doi.org/10.1007/978-3-031-14791-3_1, https://doi.org/10.1007/978-3-031-14791-3_1
26. May, A., Nowakowski, J.: Too many hints – when lll breaks lwe. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 106–137. Springer Nature Singapore, Singapore (2023)

27. Narayanan, A.K., Qiao, Y., Tang, G.: Algorithms for matrix code and alternating trilinear form equivalences via new isomorphism invariants. In: Advances in Cryptology – EUROCRYPT 2024: 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26–30, 2024, Proceedings, Part III. p. 160–187. Springer-Verlag, Berlin, Heidelberg (2024). https://doi.org/10.1007/978-3-031-58734-4_6, https://doi.org/10.1007/978-3-031-58734-4_6
28. van Noorden, C., de Perthuis, P.: Post-quantum anonymous signatures from the lattice isomorphism group action. Cryptology ePrint Archive, Paper 2026/436 (2026), https://eprint.iacr.org/2026/436
29. Panny, L., Petit, C., Stopar, M.: KLaPoTi: An asymptotically efficient isogeny group action from 2-dimensional isogenies. IACR Communications in Cryptology **2**(3), 5 (2025). https://doi.org/10.62056/ahp2wakrz, https://doi.org/10.62056/ahp2wakrz
30. Persichetti, E., Santini, P.: A new formulation of the linear equivalence problem and shorter less signatures. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 351–378. Springer Nature Singapore, Singapore (2023)
31. Preparata, F.P., Sarwate, D.V.: An improved parallel processor bound in fast matrix inversion. Inf. Process. Lett. **7**(3), 148–150 (1978). https://doi.org/10.1016/0020-0190(78)90079-0, https://doi.org/10.1016/0020-0190(78)90079-0
32. Reijnders, K., Samardjiska, S., Trimoska, M.: Hardness estimates of the code equivalence problem in the rank metric. Des. Codes Cryptography **92**(3), 833–862 (Jan 2024). https://doi.org/10.1007/s10623-023-01338-x, https://doi.org/10.1007/s10623-023-01338-x
33. Rostovtsev, A., Stolbunov, A.: PUBLIC-KEY CRYPTOSYSTEM BASED ON ISOGENIES. Cryptology ePrint Archive, Paper 2006/145 (2006), https://eprint.iacr.org/2006/145
34. Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (Jan 1991). https://doi.org/10.1007/BF00196725, https://doi.org/10.1007/BF00196725
35. Tang, G., Duong, D.H., Joux, A., Plantard, T., Qiao, Y., Susilo, W.: Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In: Advances in Cryptology – EUROCRYPT 2022: 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 – June 3, 2022, Proceedings, Part III. p. 582–612. Springer-Verlag, Berlin, Heidelberg (2022). https://doi.org/10.1007/978-3-031-07082-2_21, https://doi.org/10.1007/978-3-031-07082-2_21
36. Tang, G., Duong, D.H., Joux, A., Plantard, T., Qiao, Y., Susilo, W.: Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In: Advances in Cryptology – EUROCRYPT 2022. Lecture Notes in Computer Science, vol. 13277, pp. 582–612. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_21, https://doi.org/10.1007/978-3-031-07082-2_21

A Bounding weak keys

Here we will give a proof for the lemma bounding the amount of weak keys.

Lemma 2. *Let q, n, ℓ be integers, then we have the following probability*

$$\mathbb{P}_{\mathcal{T}\in\mathfrak{T}_n}(\mathcal{T} \textit{ is not } \textit{weak } \ell\textit{-collision resistant}) \leq q^{2n-4-(\ell^3-3\ell^2+\ell)}.$$

Proof. We will proof this by using the union bound. The idea is to count all tuples $(\mathcal{T}, \mathbf{u}, \mathbf{u}')$ with linear independent paths so that their canonical forms are equivalent. However, it turns out to be easier to count all tuples

$$(\mathcal{T}, (\mathbf{u}_1, \ldots, \mathbf{w}_\ell), (\mathbf{u}'_1, \ldots, \mathbf{w}'_\ell), (\alpha_1, \ldots, \alpha_\ell), (\beta_2, \ldots, \beta_\ell), (\gamma_2, \ldots, \gamma_\ell))$$

consisting of a tensor and two potential paths of projective points and the non-zero scaling factors. Here we take all $\mathbf{u}_i$ linear independent, similar for $\mathbf{v}_i$ and $\mathbf{w}_i$.

By assuming linear independence we can, without loss of generality, apply a group action to the tuple so that we get a tuple

$$(g \star \mathcal{T}, (\mathbf{e}_1, \ldots, \mathbf{e}_\ell), (\mathbf{e}_{\ell+1}, \ldots, \mathbf{e}'_{2\ell})).$$

Since $g \star \mathfrak{T} = \mathfrak{T}$ we can simply forget the g.

Now the constraints on $\mathcal{T}$ for this to be a valid tuple are as follows

$$\begin{aligned}
\mathcal{T}(\mathbf{e}_i, \mathbf{e}_i, -) &= \mathbf{0} \quad \forall 1 \leq i \leq 2\ell \\
\mathcal{T}(-, \mathbf{e}_i, \mathbf{e}_i) &= \mathbf{0} \quad \forall 1 \leq i \leq 2\ell \\
\mathcal{T}(\mathbf{e}_{i+1}, -, \mathbf{e}_i) &= \mathbf{0} \quad \forall 1 \leq i \leq \ell - 1 \\
\mathcal{T}(\mathbf{e}_{i+1}, -, \mathbf{e}_i) &= \mathbf{0} \quad \forall \ell + 1 \leq i \leq 2\ell - 1 \\
\mathcal{T}(\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k) = \alpha_i\beta_j\gamma_k\mathcal{T}(\mathbf{e}_{i+\ell}, \mathbf{e}_{j+\ell}, \mathbf{e}_{k+\ell}) &\quad \forall 1 \leq i, j, k \leq \ell.
\end{aligned}$$

We can now easily count how many of these equations are linear independent which is

$$(3\ell - 1)n - (3\ell - 2) + (3\ell - 1)(n - \ell) + \ell^3.$$

Thus, in formula form, we obtain (asymptotically in $q \to \infty$)

$$
\begin{aligned}
&\mathbb{P}_{\mathcal{T}\in\mathfrak{T}_n}(\mathcal{T} \text{ is } \textit{not} \text{ weak } \ell\text{-collision resistant}) \\
&\quad\le \mathbb{E}_{\mathcal{T}\in\mathfrak{T}_n}(\#\{\mathbf{u},\mathbf{u}'\in\mathbb{P}(\mathbb{F}_q^n) : \mathsf{CF}_\ell(\mathcal{T},\mathbf{u}) = \mathsf{CF}_\ell(\mathcal{T},\mathbf{u}') \neq \perp, \\
&\qquad\qquad U\cap U' = V\cap V' = W\cap W' = \emptyset\}) \\
&\quad= q^{-n^3} \sum_{\substack{\mathcal{T}\in\mathfrak{T}_n \\ \mathbf{u},\mathbf{u}'\in\mathbb{P}(\mathbb{F}_q^n)}} \mathbf{1}[\mathsf{CF}_\ell(\mathcal{T},\mathbf{u}) = \mathsf{CF}_\ell(\mathcal{T},\mathbf{u}') \neq \perp] \cdot \mathbf{1}[U\cap U' = V\cap V' = W\cap W' = \emptyset] \\
&\quad\le q^{-n^3} \sum_{\mathcal{T}\in\mathfrak{T}_n} \sum_{\substack{\mathbf{u}_1,\ldots,\mathbf{u}_{2\ell}\in\mathbb{P}(\mathbb{F}_q^n) \\ \text{lin. indpt.}}} \sum_{\substack{\mathbf{v}_1,\ldots,\mathbf{v}_{2\ell}\in\mathbb{P}(\mathbb{F}_q^n) \\ \text{lin. indpt.}}} \sum_{\substack{\mathbf{w}_1,\ldots,\mathbf{w}_{2\ell}\in\mathbb{P}(\mathbb{F}_q^n) \\ \text{lin. indpt.}}} \mathbf{1}[(\mathbf{u}_1,\ldots,\mathbf{w}_\ell) \text{ is a walk for } \mathcal{T}] \\
&\qquad\cdot \mathbf{1}[(\mathbf{u}_{\ell+1},\ldots,\mathbf{w}_{2\ell}) \text{ is a walk for } \mathcal{T}] \cdot \mathbf{1}[\mathsf{CF}_\ell(\mathcal{T},\mathbf{u}_1) = \mathsf{CF}_\ell(\mathcal{T},\mathbf{u}_{\ell+1})] \\
&\quad= q^{-n^3} \cdot q^{6\ell(n-1)} \sum_{\mathcal{T}\in\mathfrak{T}_n} \mathbf{1}[(\mathbf{e}_1,\ldots,\mathbf{e}_\ell) \text{ is a walk for } \mathcal{T}] \\
&\qquad\cdot \mathbf{1}[(\mathbf{e}_{\ell+1},\ldots,\mathbf{e}_{2\ell}) \text{ is a walk for } \mathcal{T}] \cdot \mathbf{1}[\mathsf{CF}_\ell(\mathcal{T},\mathbf{e}_1) = \mathsf{CF}_\ell(\mathcal{T},\mathbf{e}_{\ell+1})] \\
&\quad= q^{-n^3} \cdot q^{6\ell(n-1)} \cdot q^{n^3} \cdot (q-1)^{3\ell-2} \cdot q^{-(3\ell-1)n+(3\ell-2)-(3\ell-1)(n-\ell)-\ell^3} \\
&\quad= q^{2n-4-(\ell^3-3\ell^2+\ell)}.
\end{aligned}
$$