LESS is More: Code-Based Signatures without Syndromes

Jean-François Biasse[1], Giacomo Micheli[1], Edoardo Persichetti[2], and Paolo Santini[2,3]

[1] University of South Florida, USA
[2] Florida Atlantic University, USA
[3] Universitá Politecnica delle Marche, Italy
{biasse, gmicheli}@usf.edu, epersichetti@fau.edu, p.santini@pm.univpm.it

Abstract. Devising efficient and secure signature schemes based on coding theory is still considered a challenge by the cryptographic community. In this paper, we construct a signature scheme by exploring a new approach to the area. To do this, we design a zero-knowledge identification scheme, which we then render static via standard means (e.g. Fiat-Shamir). We show that practical instances of our protocol have the potential to outperform the state of the art on code-based signatures, achieving small data sizes with a low computational complexity.

1 Introduction

Digital signatures are arguably one of the most important cryptographic primitives in the modern times. Many famous examples include schemes based on RSA, as well as discrete logarithm assumptions (DSA, ECDSA), all currently standardized. However, none of the above will remain secure once a quantum computer with sufficient power and stability becomes available, due to the seminal work of Shor [34]. As a consequence, the cryptographic world is focusing its efforts on producing *Post-Quantum* secure signature schemes. At present, the scene is dominated by protocols based on lattice problems such as LWE and SIS, as well as multivariate equations (MQ, UOV), with the noticeable exception of isogeny-based signatures (e.g. [20, 17]), a newer family of primitives with very promising data size. Also, hash-based schemes such as SPHINCS [8] offer a conservative choice with reasonable performance and confidence in security. At the contrary, the community is still struggling to produce efficient and consolidated code-base signature schemes. A testament of this is given by the ongoing Post-Quantum Standardization effort by NIST [27], where only 4 code-based signature schemes were initially submitted, none of which progressed to further rounds of the competition. Indeed, many code-based schemes have been proposed over the years, either following the hash-and-sign approach like CFS [11], or relying on the Fiat-Shamir transform [18] to convert an identification scheme into a signature scheme. Unfortunately, many of the various proposals have been broken, and all those that are still considered secure suffer from one or more flaws, be that a huge public key, a large signature or a slow signing algorithm, which ultimately make them unsuitable for practical applications.

Our contribution. In this paper, we propose a signature scheme based on a novel approach from coding theory. The scheme is built upon an identification scheme that relies on the hardness of the Linear Code Equivalence problem; consequently, we name our protocol LESS as in Linear Equivalence Signature Scheme. This problem has been studied for a long time with regards to its application to the McEliece and Niederreiter cryptosystem but, to the best of our knowledge, no scheme has ever been instantiated on it as a stand-alone problem. In a 2013 paper [32], where the hardness of the problem is studied, there is a brief reference to zero-knowledge protocols, and the authors describe a version of Girault's identification scheme [21] using monomial matrices. However, this is still fundamentally a protocol based on the hardness of Syndrome Decoding, following the traditional approach of code-based cryptography. Our approach, on the other hand, does not involve syndromes and decoding at all, and is purely based on the hardness of determining the linear isometry between two codes. This allows us to choose parameters that are much smaller than those usually selected by schemes based on SDP, which have to protect against generic decoding attacks such as those in the Information-Set Decoding (ISD) family, or equivalent. As a consequence, we are able to design extremely practical instances, while at the same time setting up a new framework for code-based signatures.

The paper is organized as follows. In Section 2, we define our notation and give some preliminary definitions about coding theory and identification schemes. In Section 3, we present the central notion upon which our protocol is based, the Code Equivalence Problem. We then introduce our scheme, in Section 4, along with a proof of security. A careful security analysis and description of attack techniques is given in Section 5. In Section 6, we make further considerations about the problem, and we discuss the applicability of Quantum attacks. We briefly describe the Fiat-Shamir transform in Section 7, and give other details of how we can convert our identification scheme into a full-fledged signature scheme. In Section 8, we provide an accurate comparison with the state-of-the-art code-based signature schemes, present some performance figures, and explain our computational advantage. Finally, we conclude in Section 9.

2 Preliminaries

We denote scalars with lowercase letters, and sets with uppercase letters. Vectors and matrices are written in boldface, respectively lowercase and upper case. We will use $\otimes$ to denote the Kronecker product between matrices (or vectors). We write a for a function or relation, and $\mathcal{A}$ for an algorithm. I_n stands for the $n \times n$ identity matrix, and $[a; b]$ for the set of integers $\{a, a+1, \dots, b\}$. Finally, we use $\mathbb{U}(A)$ to indicate the uniform distribution over the set A, and $\overset{\$}{\leftarrow} A$ for the action of sampling uniformly at random from A.

Let $\mathbb{F}_q$ be the finite field of order q. We write $\mathrm{GL}_k(q)$ for the set of invertible $k \times k$ matrices with elements in $\mathbb{F}_q$. Let S_n be the set of permutations over n elements. These can equivalently be described as functions $\pi : \mathbb{F}_q^n \to \mathbb{F}_q^n$ or in matrix form as $n \times n$ matrices with exactly one 1 per row and column. By analogy, we denote with $M_n(q)$ the set of monomial matrices with elements in $\mathbb{F}_q$, i.e. all the matrices of the form $\boldsymbol{Q} = \boldsymbol{DP}$ where $\boldsymbol{P}$ is an $n \times n$ permutation matrix and $\boldsymbol{D} = \{d_{ij}\}$ is an $n \times n$ diagonal matrix such that $d_{ii} = d_i \in \mathbb{F}_q^*$. Given a vector $\boldsymbol{x} = (x_1, \cdots, x_n) \in \mathbb{F}_q^n$ and a permutation $\pi \in S_n$, we write the action of π on $\boldsymbol{x}$ as $\pi(\boldsymbol{x}) = (x_{\pi^{-1}(1)}, \cdots, x_{\pi^{-1}(n)})$.

2.1 Coding Theory

An $[n, k]$-*linear code* $\mathfrak{C}$ of length n and dimension k over $\mathbb{F}_q$ is a k-dimensional vector subspace of $\mathbb{F}_q^n$. It can be represented by a matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$, called *generator matrix*, whose rows form a basis for the vector space.Then, the generator matrix defines the code as a mapping between vectors $\boldsymbol{u} \in \mathbb{F}_q^k$ and the corresponding words $\boldsymbol{uG}$. Obviously, there exist more than one generator matrix for the same code, corresponding to different choices of basis. It follows that all generator matrices are connected via a change-of-basis matrix, i.e. an invertible matrix $\boldsymbol{S} \in \mathrm{GL}_k(q)$ such that $\boldsymbol{G}' = \boldsymbol{SG}$. Alternatively, a linear code can be represented as the kernel of a matrix $\boldsymbol{H} \in \mathbb{F}_q^{(n-k) \times n}$, known as *parity-check matrix*, i.e. $\mathfrak{C} = \{\boldsymbol{x} \in \mathbb{F}_q^n : \boldsymbol{H}\boldsymbol{x}^T = 0\}$. Once again, the parity-check matrix of a code is not unique. For both cases, there exists a standard choice, called *systematic form.* For the generator matrix, this corresponds to $\boldsymbol{G} = (\boldsymbol{I}_k \mid \boldsymbol{M})$, which can be obtained as the row-reduced echelon form starting from any other generator matrix. The systematic form of the parity-check matrix is given by $\boldsymbol{H} = (-\boldsymbol{M}^T \mid \boldsymbol{I}_{n-k})$. Note that, in general, it is possible that computing the row-reduced echelon form of $\boldsymbol{G}$ returns a matrix that does not have full rank. If this is the case, there are procedures to obtain a matrix in systematic form by reducing with respect to a different minor (for example, the Round 2 specification document of [10] describes one that works in constant-time). We denote such a procedure with SF.

For every linear code, we can define the *dual code* as the set of words that are orthogonal to the code, i.e. $\mathfrak{C}^\perp = \{\boldsymbol{y} \in \mathbb{F}_q^n \ : \ \forall \boldsymbol{x} \in \mathfrak{C}, \ \boldsymbol{x} \cdot \boldsymbol{y}^T = 0\}$. It is then easy to see that a parity-check matrix of a linear code is a generator of its dual, and viceversa. In fact, it must be that $\boldsymbol{G} \cdot \boldsymbol{H}^T = \boldsymbol{0}_{k \times (n-k)}$. Codes that are contained in their dual, i.e. $\mathfrak{C} \subseteq \mathfrak{C}^\perp$, are called *weakly self-dual*, and codes that are equal to their dual, i.e. $\mathfrak{C} = \mathfrak{C}^\perp$, are called simply *self-dual.*

2.2 Identification Schemes and Zero-Knowledge Protocols

We now recall some standard cryptographic notions about the so-called Sigma protocols, and how to derive identification schemes from them. To do so, we follow the general outline given in [20], in turn based on definitions and notation from [23, 12, 37, 1, 5].

Definition 1 (Sigma Protocol). *Consider two sets X and Y parameterized by a security parameter λ. Let R be a relation on $X \times Y$ defining a language $L = \{y \in Y \ : \ \exists x \in X, \mathsf{R}(x, y) = 1\}$. We call* witness *an element $x \in X$ such that, given $y \in L$, verifies $\mathsf{R}(x, y) = 1$. We define a* Sigma Protocol *as a 3-round interactive protocol between two PPT algorithms, a* Prover $\mathcal{P}$ *and a* Verifier $\mathcal{V}$*, as described in Table 1 below.*

Table 1. Sigma Protocol.

Prover Data	A witness x for $y \in L$.
Verifier Data	$y \in L$.

PROVER		VERIFIER
$\alpha \leftarrow \mathcal{P}(x,y)$	$\xrightarrow{\alpha}$	
	$\xleftarrow{\beta}$	$\beta \leftarrow \mathcal{V}(\alpha,y)$
$\gamma \leftarrow \mathcal{P}(\alpha,\beta,x,y)$	$\xrightarrow{\gamma}$	$\{0,1\} \leftarrow \mathcal{V}(\alpha,\beta,\gamma,y)$

The triple (α,β,γ) forms a *transcript* of the protocol, and the three values are usually known as *commitment*, *challenge* and *response*, respectively. A transcript for which the verifier outputs 1 (accept) is called *valid*.

Sigma protocols are often required to satisfy the following properties:

- *Completeness*: when $y \in L$, an honest prover is accepted with probability 1.
- *2-Special Soundness*: there exists an *extractor* algorithm $\mathcal{X}$ such that, for any $y \in L$, given two valid transcripts (α,β,γ) and (α,β',γ') with $\beta \neq \beta'$, the output $\mathcal{X}(\alpha,\beta,\gamma,\beta',\gamma')$ is a witness for R.
- *Honest-Verifier Zero-Knowledge*: there exists a *simulator* algorithm $\mathcal{S}$ such that, on input $y \in L$, is able to generate a valid transcript (α,β,γ) that is distributed identically to one obtained from a real execution of the protocol.

An identification scheme can be defined as a special type of Sigma protocol, where the relation R is defined over key pairs $(\mathsf{sk},\mathsf{pk})$, and one can think of sk as a witness for pk.

Definition 2. *Let λ be a security parameter. A* Canonical Identification Scheme *is composed by a triple of PPT algorithms $(\mathcal{K},\mathcal{P},\mathcal{V})$, respectively Key Generator, Prover and Verifier, and a parameter ℓ, the length of the challenge, interacting as described in Table 2, below.*

Table 2. Canonical Identification Scheme.

Private Key	A private key sk output by $\mathcal{K}(1^\lambda)$.
Public Key	The public key pk corresponding to sk.

PROVER		VERIFIER
$\mathsf{cmt} \leftarrow \mathcal{P}(\mathsf{sk},\mathsf{pk},\rho)$	$\xrightarrow{\mathsf{cmt}}$	
	$\xleftarrow{\mathsf{ch}}$	$\mathsf{ch} \leftarrow \{0,1\}^\ell$
$\mathsf{rsp} \leftarrow \mathcal{P}(\mathsf{sk},\mathsf{pk},\rho,\mathsf{cmt},\mathsf{ch})$	$\xrightarrow{\mathsf{rsp}}$	$\{0,1\} \leftarrow \mathcal{V}(\mathsf{pk},\mathsf{cmt},\mathsf{ch},\mathsf{rsp})$

As before, the exchanged data takes the name of commitment, challenge, and response. Note that we have made explicit the role of the randomness ρ in the generation of the challenge (remember that $\mathcal{P}$ is a probabilistic algorithm). The scheme is said to be *non-trivial* if $\ell \geq \lambda$.

An *impersonator* $\mathcal{I}$ is a PPT adversary that aims to get verified by $\mathcal{V}$ without knowing the private key. The impersonator is able to observe a number of transcripts from honest executions, before producing a commitment, receiving a corresponding challenge, and finally outputting its response. The impersonator is commonly said to have *cheating probability* equal to $1/2^\ell$. We say that $\mathcal{I}$ *wins* if $\mathcal{V}(\mathsf{pk}, \mathsf{cmt}, \mathsf{ch}, \mathsf{rsp}) = 1$, and we define $\mathcal{I}$'s advantage as

$$\left|\mathsf{Adv}(\mathcal{I}, \lambda) = \mathsf{Pr}[\mathcal{I} \text{ wins}] - \frac{1}{2^\ell}\right|.$$

We say that an identification scheme is *secure against impersonation under passive attacks* if the advantage of any PPT impersonator is negligible.

Usually, identification schemes are defined using challenges that are too short to obtain a non-trivial instance, the most common case being, as in this paper, $\ell = 1$ (i.e. the challenge is a single bit). However, it is possible to obtain a non-trivial scheme by iterating the protocol t times (which can be done in parallel). Formally, the prover generates commitments cmt_i for $i = 1, \dots, t$, then receives a challenge $\mathsf{ch} \in \{0,1\}^{t\ell}$, parses it into t blocks ch_i of length ℓ each, and produces responses rsp_i for $i = 1, \dots, t$. The verifier receives as input $(\mathsf{pk}, \mathsf{cmt}_1, \dots, \mathsf{cmt}_t, \mathsf{ch}, \mathsf{rsp}_1, \dots, \mathsf{rsp}_t)$ and accepts if and only if $\mathcal{V}(\mathsf{pk}, \mathsf{cmt}_i, \mathsf{ch}_i, \mathsf{rsp}_i) = 1$ for $i = 1, \dots, t$. This reduces the cheating probability to $1/2^{t\ell}$, which makes the scheme non-trivial as long as $t \geq \lambda/\ell$.

3 The Code Equivalence Problem

In this section we introduce the ideas upon which we base the security of our protocol. We first formally define the notion of code equivalence.

Definition 3 (Permutation Code Equivalence). *We say that two codes* $\mathfrak{C}$ *and* $\mathfrak{C}'$ *are* permutationally equivalent*, and write* $\mathfrak{C} \overset{\text{PE}}{\sim} \mathfrak{C}'$*, if there is a permutation* $\pi \in S_n$ *that maps* $\mathfrak{C}$ *into* $\mathfrak{C}$*, i.e.*

$$\mathfrak{C}' = \{\pi(\boldsymbol{x}),\ \ \boldsymbol{x} \in \mathfrak{C}\}.$$

The previous notion of code equivalence can be extended using linear isometries. Indeed, let $\mu = (\boldsymbol{v}, \pi) \in \mathbb{F}_q^{*n} \rtimes S_n$ be an isometry μ, such that

$$\mu(\boldsymbol{x}) = (v_1 x_{\pi^{-1}(1)}, \cdots, v_n x_{\pi^{-1}(n)}).$$

We can then generalize the previous definition as follows.

Definition 4 (Linear Code Equivalence). *We say that two codes* $\mathfrak{C}$ *and* $\mathfrak{C}'$ *are* linearly equivalent*, and write* $\mathfrak{C} \overset{\text{LE}}{\sim} \mathfrak{C}'$*, if there is a linear isometry* $\mu = (\boldsymbol{v}; \pi) \in \mathbb{F}_q^{*n} \rtimes S_n$ *such that* $\mathfrak{C}' = \mu(\mathfrak{C})$*, i.e.* $\mathfrak{C}' = \{\mu(\boldsymbol{x}),\ \ \boldsymbol{x} \in \mathfrak{C}\}$.

It is clear the previous definitions can equivalently be stated in terms of generator (or parity-check) matrices; furthermore, the application of a permutation (resp. linear isometry) corresponds to the right multiplication by a permutation matrix $\boldsymbol{P}$ (resp. monomial matrix $\boldsymbol{Q}$).

Let $\mathfrak{C}$ and $\mathfrak{C}'$ be two codes with respective generator matrices $\boldsymbol{G}$ and $\boldsymbol{G}'$: we then have

$$\mathfrak{C} \overset{\text{PE}}{\sim} \mathfrak{C}' \iff \exists(\boldsymbol{S}, \boldsymbol{P}) \in \mathrm{GL}_k(q) \times S_n \text{ s.t. } \boldsymbol{G}' = \boldsymbol{SGP},$$
$$\mathfrak{C} \overset{\text{LE}}{\sim} \mathfrak{C}' \iff \exists(\boldsymbol{S}, \boldsymbol{Q}) \in \mathrm{GL}_k(q) \times M_n(q) \text{ s.t. } \boldsymbol{G}' = \boldsymbol{SGQ}.$$

Another notion of code equivalence (using *semilinear* isometries) is often found in the literature; however, it is not needed for our protocol, and therefore we do not present it here. We instead refer the interested reader to [32] for further details, and move on to present the hard problems connected to the notions we just described.

Problem 1 (Permutation Code Equivalence) *Let $\boldsymbol{G}, \boldsymbol{G}' \in \mathbb{F}_q^{k\times n}$ be two generator matrices for, respectively, linear codes $\mathfrak{C}$ and $\mathfrak{C}'$. Determine whether the two codes are permutationally equivalent, i.e. if there exist two matrices $\boldsymbol{S} \in GL_k(q)$ and $\boldsymbol{P} \in S_n$ such that $\boldsymbol{G}' = \boldsymbol{SGP}$.*

Problem 2 (Linear Code Equivalence) *Let $\boldsymbol{G}, \boldsymbol{G}' \in \mathbb{F}_q^{k\times n}$ be two generator matrices for, respectively, linear codes $\mathfrak{C}$ and $\mathfrak{C}'$. Determine whether the two codes are linearly equivalent, i.e. if there exist two matrices $\boldsymbol{S} \in GL_k(q)$ and $\boldsymbol{Q} \in M_n(q)$ such that $\boldsymbol{G}' = \boldsymbol{SGQ}$.*

The two problems above are clearly two different flavors of the same problem, namely, deciding whether two codes are equivalent, which differ according to which notion of code equivalence is considered. However, as we will see, the connection between the two is not as obvious as it seems.

3.1 Hardness

As proven in [28], the permutation equivalence problem is unlikely to be NP-complete, since this property would imply a collapse of the polynomial hierarchy. While the problem can be efficiently solved for some families of codes, there are however many instances that, after almost 40 years of study, are still intractable.

The first algorithm to solve this problem was proposed by Leon in 1982 [24], and is able to reconstruct the secret permutation from its action on the set of codewords with fixed weight. The permutation can efficiently be recovered when this set is not too large. The bottleneck is in the codewords search, whose time complexity is $nq^{O(k)}$. Thus, as noted in [4], Leon's algorithm is impractical, unless considering codes of small dimension defined over small finite fields.

The Support Splitting Algorithm (SSA), due to Sendrier [31, 32], strongly improves upon Leon's algorithm. The algorithm is based on the concept of the hull, that is, the intersection between a code and its dual. The hull computation requires simple linear algebra while the time complexity of the whole algorithm essentially grows as q^h, where h is the hull's dimension. For random codes, this dimension is with high probability equal to a small constant [33], de facto making SSA a polynomial-time solver for permutation equivalence in many cases.

One case in which SSA fails is that of codes with a trivial (i.e. zero) hull. However, an efficient treatment of this situation has recently been provided [4] through a reduction, running in time $O(n^{\omega})$ (with $2 \leq \omega \leq 3$), from permutation equivalence to an instance of the graph isomorphism problem between undirected weighted graphs. Another case which cannot efficiently be solved through SSA is that of codes with a large hull. In fact, since the time complexity is dominated by q^h, SSA becomes quickly unfeasible as h grows. This is, for instance, the case of self-dual (or weakly self-dual codes), for which $h = k$: for such codes, SSA can be made arbitrarily hard by choosing codes with a sufficiently large dimension. The hardness of such instances is corroborated by the reduction to graph isomorphism in [4] which, for non-trivial hulls, runs in time $O(hn^{\omega+h+1})$ and, as expected, becomes quickly unfeasible for large values of h.

We conclude this section with a note on the hardness of linear equivalence. As shown in [32], the problem of establishing the linear equivalence between two codes can always be reduced to that of finding a permutation equivalence between their closures. Thus, constructing the closures (as we detail in Section 5) and applying SSA is enough to solve the linear equivalence. However, when $q \geq 5$, the closure of a code is always weakly-self dual. It follows that such instances are exactly the hardest ones for SSA to solve. These results are confirmed by the analysis in [29], which includes a study of algebraic approaches to the code equivalence problem.

4 Protocol Description

We begin by describing the underlying identification scheme, in Table 3.

Table 3. The LESS Identification Scheme.

Public Data	Parameters $q, n, k \in \mathbb{N}$, matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$ and hash function H.
Private Key	Invertible matrix $\boldsymbol{S} \in \mathrm{GL}_k(q)$ and monomial matrix $\boldsymbol{Q} \in M_n(q)$.
Public Key	$\boldsymbol{G}' = \boldsymbol{SGQ}$.

PROVER		VERIFIER
Choose $\tilde{\boldsymbol{Q}} \overset{\$}{\leftarrow} \mathbb{F}_q^{n \times n}$ and set $\tilde{\boldsymbol{G}} = \boldsymbol{G}\tilde{\boldsymbol{Q}}$. Set $h = \mathsf{H}(\mathsf{SF}(\tilde{\boldsymbol{G}}))$.	$\xrightarrow{h}$	
	$\xleftarrow{b}$	$b \overset{\$}{\leftarrow} \{0, 1\}$.
If $b = 0$ then $\mu = \tilde{\boldsymbol{Q}}$. If $b = 1$ then $\mu = \boldsymbol{Q}^{-1}\tilde{\boldsymbol{Q}}$.	$\xrightarrow{\mu}$	Accept if $\mathsf{H}(\mathsf{SF}(\boldsymbol{G}\mu)) = h$. Accept if $\mathsf{H}(\mathsf{SF}(\boldsymbol{G}'\mu)) = h$.

We now show that our protocol satisfies the necessary security requirements for identification schemes.

Completeness. It is immediate to check that the protocol is correct, and an honest prover always gets accepted. In fact, if $b = 0$ the verifier receives $\mu = \tilde{\boldsymbol{Q}}$ and then obviously can check that $\mathsf{H}(\mathsf{SF}(\boldsymbol{G}\mu)) = \mathsf{H}(\mathsf{SF}(\boldsymbol{G}\tilde{\boldsymbol{Q}})) = \mathsf{H}(\mathsf{SF}(\tilde{\boldsymbol{G}})) = h$ since by construction $\tilde{\boldsymbol{G}} = \boldsymbol{G}\tilde{\boldsymbol{Q}}$. On the other hand, if $b = 1$, then $\mu = \boldsymbol{Q}^{-1}\tilde{\boldsymbol{Q}}$ and we have $\mathsf{H}(\mathsf{SF}(\boldsymbol{G}'\mu)) = \mathsf{H}(\mathsf{SF}(\boldsymbol{SGQQ}^{-1}\tilde{\boldsymbol{Q}})) = \mathsf{H}(\mathsf{SF}(\boldsymbol{S}\tilde{\boldsymbol{G}}))$, which is also equal to h since $\boldsymbol{S}\tilde{\boldsymbol{G}}$ generates the same code as $\tilde{\boldsymbol{G}}$ and therefore the two matrices have the same systematic form.

Honest-Verifier Zero-Knowledge. In this section we show that the produced responses do not leak information about the private key. We do this by proving that there exists a probabilistic polynomial time simulator algorithm $\mathcal{S}$ that, without the knowledge of the private key, is able to produce a transcript which is indistinguishable from one obtained after an interaction with an honest verifier. To this end, we introduce the following straightforward Lemma.

Lemma 1. *Let $M_n(q)$ be the set of monomial matrices as defined in Section 2. Then for any $\boldsymbol{A} \in M_n(q)$ and $\boldsymbol{B} \xleftarrow{\$} M_n(q)$, we have $\boldsymbol{A}^{-1}\boldsymbol{B} \sim \mathbb{U}(M_n(q))$.*

The simulator works as follows.

- When the challenge is $b = 0$, it can trivially simulate correctly by choosing a matrix $\tilde{\boldsymbol{Q}}$ uniformly at random. This, in fact, corresponds to a legitimate response for this challenge, and doesn't include the secret.
- When the challenge is $b = 1$, the simulator again chooses a matrix, say $\boldsymbol{Q}^*$, uniformly at random. By Lemma 1, we have seen that the product $\boldsymbol{Q}^{-1}\tilde{\boldsymbol{Q}}$ that would be output by an honest execution of the protocol is uniformly distributed among all monomial matrices. Therefore $\mathcal{S}$ is able to simulate correctly in this case.

This simple argument shows that both responses are actually indistinguishable from randomly generated ones, and thus do not reveal any secret.

Soundness. Finally, we prove that the protocol is 2-special sound. We do this by describing an extractor algorithm and showing that it is able to find a witness, i.e. solve the code equivalence problem. To this end, let $\mathcal{A}$ be an adversary that is given an instance $\{\boldsymbol{G}, \boldsymbol{G}'\}$ as in Problem 2. The algorithm proceeds as follows.

To begin, set $\boldsymbol{G}$ and $\boldsymbol{G}'$ as public data and public key for the identification scheme. Then, obtain a transcript $(\mathsf{cmt}, \mathsf{ch}_0, \mathsf{ch}_1, \mathsf{rsp}_0, \mathsf{rsp}_1)$ such that $\mathsf{ch}_0 \neq \mathsf{ch}_1$ and the verifier accepts $(\mathsf{cmt}, \mathsf{ch}_i, \mathsf{rsp}_i)$ for $i \in \{0, 1\}$: in other words, the transcript is such that both challenges are satisfied for the same commitment. Thus, the two responses must be two monomial matrices $\tilde{\boldsymbol{Q}}$ and $\boldsymbol{Q}^*$ such that

$$\mathsf{H}(\mathsf{SF}(\boldsymbol{G}\tilde{\boldsymbol{Q}})) = \mathsf{H}(\mathsf{SF}(\boldsymbol{G}'\boldsymbol{Q}^*)).$$

Unless he is able to find a collision for the hash function, this means

$$\mathsf{SF}(\boldsymbol{G}\tilde{\boldsymbol{Q}}) = \mathsf{SF}(\boldsymbol{G}'\boldsymbol{Q}^*).$$

At this point, since two matrices with the same systematic form define the same linear code, we have that

$$\hat{\boldsymbol{S}}\boldsymbol{G}\tilde{\boldsymbol{Q}} = \boldsymbol{G}'\boldsymbol{Q}^*$$

for some invertible matrix $\hat{\boldsymbol{S}}$ or, if we write $\hat{\boldsymbol{Q}} = \tilde{\boldsymbol{Q}}(\boldsymbol{Q}^*)^{-1}$,

$$\hat{\boldsymbol{S}}\boldsymbol{G}\hat{\boldsymbol{Q}} = \boldsymbol{G}'.$$

It is then easy to verify that $\hat{\boldsymbol{Q}}$, which can be calculated immediately from the two responses, and $\hat{\boldsymbol{S}}$, which can be then computed via linear algebra, provide the desired witness.

5 Security Analysis

In this section we assess the complexity of the state-of-the-art algorithms for solving the code equivalence problem. We begin by analyzing Leon's algorithm and SSA, which both originally target the permutation equivalence problem. We then describe how the algorithms can be applied to solve linear equivalence.

5.1 Leon's Algorithm

Leon's algorithm [24] solves permutation equivalence by analyzing its action on the subset of codewords with fixed weight ω. Once such a set is computed, it gets partitioned into smaller subsets, which are then used to retrieve the permutation mapping one code to the other. The partitioning phase has very low complexity, while finding all codewords of weight ω is the actual bottleneck of the algorithm. Usually ω is set as the minimum distance of the code (which, for random codes, can be estimated with the the GV bound); if this set does not have sufficient structure, then ω is slightly increased. We now briefly describe how the codeword enumeration can be performed. Let $\boldsymbol{G}$ be the generator of a code $\mathfrak{C}$ of length n and dimension k, and $\boldsymbol{G}_{\mathsf{SF}} = \mathsf{SF}(\boldsymbol{G})$. For $\delta \leq w$, and $i \leq k - \delta$, we define

$$U(\delta, i) = \left\{\boldsymbol{u} \in \mathbb{F}_q^k \text{ s.t. } \mathrm{wt}(\boldsymbol{u}) = \delta,\ u_i = 1,\ \ u_j = 0\ \ \forall j < i\right\}.$$

It can then be easily seen that, when $\omega \leq k$ (which is the case we consider in this paper) we have

$$\{\boldsymbol{c} \in \mathfrak{C} \text{ s.t. } \mathrm{wt}(\boldsymbol{c}) = \omega\} \subseteq \left\{a(\boldsymbol{u}\boldsymbol{G}_{\mathsf{SF}}),\ \ a \in \mathbb{F}_q^* \setminus \{1\},\ \ \boldsymbol{u} \in \bigcup_{\delta=1}^{\omega}\bigcup_{i=0}^{k-\delta} U(\delta, i)\right\}.$$

From a practical point of view, the codeword search can be performed by testing all codewords of the form $\boldsymbol{u}\boldsymbol{G}_{\mathsf{SF}}$. Once a codeword of weight ω is found, then all of its scalar multiples are computed. In particular, few scalar multiples will be computed, with respect to the whole number of tested codewords, since we expect the set of weight-ω codewords to be relatively small; thus, we can neglect the computational cost of this step. For each candidate $\boldsymbol{u}$, we need to compute $n - k$ codeword symbols; when $\boldsymbol{u}$ has weight δ, this can be done with $\delta - 1$ multiplications (since the first non null entry of $\boldsymbol{u}$ is 1) and $\delta - 1$ sums in $\mathbb{F}_q$.

Since all sets $U(\delta, i)$ are disjoint, it can be straightforwardly shown that the number of vectors $\boldsymbol{u}$ that are tested is $\sum_{\delta=1}^{\omega}\binom{k}{\delta}(q-1)^{\delta-1}$. Then, by neglecting the cost of the partitioning step, we have

$$C_{LEON} = O\left(4(n-k)\sum_{\delta=1}^{\omega}(\delta-1)\binom{k}{\delta}(q-1)^{\delta-1}\right). \tag{1}$$

One final remark is about eventual future developments regarding Leon's algorithm. Indeed, the algorithm is inefficient for large codes, or for large finite fields, since the codeword enumeration quickly becomes unfeasible. This step cannot be avoided, as the algorithm requires to find *all* the codewords of weight ω. We do not exclude the possibility of strong improvements in Leon's algorithm, leading to the possibility of operating with just a subset of all such codewords. In such a case, codewords of some weight ω can be efficiently determined by means of ISD algorithms which, at each call, randomly pick a codeword of the desired weight. Thus, multiple ISD calls can be used to find the required number of codewords of the desired weight. If this scenario ever became a concern, the issue could be entirely avoided by choosing code parameters such that even a single ISD call is computationally too expensive.

5.2 The Support Splitting Algorithm

A fundamental concept to analyze SSA is that of *signature function*, introduced in [31], which is defined in the following way.

Definition 5. *Let $\mathfrak{C}$ be a linear code of length n; we say that a function S is a* signature function *over a set F if it maps $\mathfrak{C}$ and a position $i \in [0; n-1]$ to F and is such that*

$$\mathsf{S}(\mathfrak{C}, i) = \mathsf{S}\big(\pi(\mathfrak{C}), \pi(i)\big), \quad \forall \pi \in S_n.$$

We say a signature function is fully discriminant *if* $\mathsf{S}(\mathfrak{C}, i) \neq \mathsf{S}(\mathfrak{C}, j), \quad \forall i \neq j.$

Signature functions can be used to recover information about the permutation that is acting on the code; in particular, once in possession of a fully discriminant signature, the permutation π can immediately be recovered, since

$$\mathsf{S}(\mathfrak{C}, i) = \mathsf{S}(\mathfrak{C}', j) \iff j = \pi(i). \tag{2}$$

Assuming that such a fully discriminant function S is available, SSA corresponds to the trivial algorithm that searches for collisions between the sets of values $\mathsf{S}(\mathfrak{C}, i)$ and $\mathsf{S}(\mathfrak{C}', j)$, for $(i, j) \in [1; n] \times [1; n]$. We point out that the existence of such a function (and one that doesn't require unfeasible computation) is clearly not guaranteed for all pairs of codes. In such cases, SSA makes use of signatures refining, that is, new computations and combinations of signatures, that proceed until a fully discriminant function is obtained [32]. In this paper, with a conservative choice, we assume that the chosen signature function is fully discriminant for the pair of codes considered, and that the refining of signatures is never required. In this way, we are guaranteed to provide a lower bound on the actual complexity of SSA. The signature function proposed by Sendrier in [31] is based on the *hull space* of a code, which is defined as

$$\mathrm{Hull}\,(\mathfrak{C}) = \mathfrak{C} \cap \mathfrak{C}^{\perp}.$$

An efficiently computable signature function, which at the same time is sufficiently discriminant, can be obtained from the weight enumerator function of the hull of a code (with some proper additional operations). For the complete details, we refer to reader to [31]; for the purpose of this paper, we are just interested in the associated complexity, which grows as $q^{d_{\mathrm{Hull}}}$, where d_{Hull} denotes the dimension of the hull. Then, a conservative estimate for the complexity of using SSA to solve the Permutation Equivalence Problem is given by

$$C_{SSA} = O\big(n^3 + n^2 q^{d_{\mathrm{Hull}}} \log n\big). \tag{3}$$

The leading term in Equation (3) is clearly $q^{d_{\mathrm{Hull}}}$. Thus, the dimension of the hull plays a central role in determining the complexity of the algorithm. As verified empirically in [31], the hull of random codes is very likely to have small dimension. Furthermore, it has been shown in [33] how, as n grows, the size of the hull of a random code approaches a small constant which depends only on q. It follows that, for random codes, Permutation Code Equivalence can be efficiently solved by SSA with high probability. However, for special choices such as weakly self-dual codes, the hull can be made arbitrarily large by increasing the code dimension. Thus, for such codes, SSA has exponential complexity.

5.3 Application to Linear Code Equivalence

We now describe how the algorithms can be used to tackle the linear equivalence problem. To do that, we need to introduce the concept of closure of a code.

Definition 6. *Let* $\mathbb{F}_q = \{a_0 = 0, a_1, \cdots, a_{q-1}\}$*, and* $\boldsymbol{a} = (a_1, \cdots, a_{q-1})$*. We define the* closure *of a linear code* $\mathfrak{C}$ *as*

$$\tilde{\mathfrak{C}} = \{\boldsymbol{c} \otimes \boldsymbol{a},\ \ \boldsymbol{c} \in \mathfrak{C}\}.$$

As observed by Sendrier in [32], the linear equivalence problem between a pair of codes $\mathfrak{C}$ and $\mathfrak{C}'$ can be reduced to the permutation code equivalence problem between their closures $\tilde{\mathfrak{C}}$ and $\tilde{\mathfrak{C}}'$. We remark that the above definition for the closure is slightly different from the one considered in [32], since we consider a different order to build the closure's coordinates; clearly, this has no practical impact in the relation between the linear and permutation equivalence problems.

To use Leon's algorithm, it is necessary to enumerate all the low-weight codewords in the closures. Then, one can run the algorithm on the set of codewords of weight $\omega' = (q-1)\omega$, where ω can be approximated by the GV bound for parameters n, k and q. The time complexity can be estimated through Equation (1) by setting $\omega = d_{\mathsf{GV}}(n, k, q)$. To use SSA, instead, it is enough to apply the algorithm directly on the closures. However, a crucial result is that, for $q \geq 5$, closures of codes are always weakly self-dual, i.e., have a hull of maximum dimension k.

It is worth noting that an isometry between $\mathfrak{C}$ and $\mathfrak{C}'$ can be built (with simple linear algebra) from a linear equivalence between their duals $\mathfrak{C}^{\perp}$ and $\mathfrak{C}'^{\perp}$, whose closures have hull of dimension $n-k$. Thus, when $2k > n$, the optimal strategy is to attack the duals of the considered codes.

6 Quantum Attacks on the Code Equivalence Problem

To the best of our knowledge, there are no dedicated quantum algorithms for solving the Code Equivalence Problem. In here, we discuss the applicability of the the usual quantum cryptanalysis approaches.

First, we consider the use of Grover's search algorithm [22], which is known to improve the cryptanalysis of a system in almost all cases. Indeed, the algorithm allows us to efficiently search an unsorted database X, consisting of N entries, for an element $x \in X$ such that $f(x) = 1$. The cost of the algorithm is in $O(\sqrt{N}C_f)$, where $f : X \to \{0,1\}$ and where C_f is the cost of implementing f. Note that here "cost" means either number of gates or execution time (i.e. circuit depth). With regards to Leon's algorithm, it can indeed be expected that an application of Grover can improve the search part of the algorithm, leading to the usual speedup which corresponds, in the worst case, to roughly halving the complexity exponent (if one ignores the remaining part). This is similar to what happens in the case of Information-Set Decoding (see for instance [7]). Interestingly, though, a Grover search over all possible secrets (i.e. $P \in S_n$) would not outperform the classical SSA because of the size of S_n.

In principle, it is also possible to use Grover's algorithm *within* SSA. Indeed, for each $i \in [1;n]$, the search for $j \in [1;n]$ such that $j = \pi(i)$ corresponds to finding $j \in [1;n]$ such that $f(j) = 1$, where the function $f : [1;n] \to \{0,1\}$ is defined as

$$f(j) = \begin{cases} 1 \text{ if } \mathsf{S}(\mathfrak{C}', j) = \mathsf{S}(\mathfrak{C}, i) \\ 0 \qquad \text{otherwise} \end{cases}$$

for a fully discriminant function S. Following the application of Bennett's generic method [6] (which converts any algorithm taking time T and space S into a reversible algorithm taking time $T^{1+\varepsilon}$ and space $O(S \log T)$), the cost of a quantum circuit evaluating f is that of S, which is in $\tilde{O}(nq^{d_{\text{Hull}}} \log n)$. Thus, the search for $j \in [1;n]$ such that $j = \pi(i)$ costs

$$O(\sqrt{|[1;n]|}C_f) = \tilde{O}(n^{3/2}q^{d_{\text{Hull}}} \log n).$$

This process needs to be repeated $n/2$ times. Every time a pair $(i, \pi(i))$ is found, both elements can be removed from the search space. This means that, in the previous formulas, we replace $[1;n]$ with $[1;n]^{(k)}$, where $n-2k \leq |[1;n]^{(k)}| \leq n-k$ (at each stage we remove either 1 or 2 elements depending on whether $\pi(i) = i$). Our total cost is

$$O\left(\left(\sum_{k\leq n/2}\sqrt{|[1;n]^{(k)}|}\right)C_f\right).$$

We can bound this using the fact that

$$\underbrace{\sum_{k=1}^{n/2}\sqrt{2k}}_{\Omega(n^{3/2})} \leq \sum_{k\leq n/2}\sqrt{|[1;n]^{(k)}|} \leq \underbrace{\sum_{k=n/2}^{n}\sqrt{k}}_{\Omega(n^{3/2})}.$$

In the end, the complexity of the overall procedure is $\tilde{O}(n^{5/2}q^{d_{\text{Hull}}}\log n)$, which does not outperform the classical method consisting in $2n$ evaluations of S followed by a matching of the values obtained.

The other famous family of algorithms for quantum cryptanalysis is based on quantum Fourier sampling. These algorithms can be seen as generalizations of Shor's algorithm for factoring and solving the Discrete Logarithm Problem [34]. The general approach is to rephrase a problem as the search for a secret subgroup H within a known "control group" G. The Quantum Fourier Transform (QFT) over G allows us to create a state whose measurement (hopefully) yields an element in $\hat{H}$. By repeating this operation and using ad-hoc methods depending on H, one can recover H and solve the problem. In [15] and in the follow up work [14], Dinh, Moore and Russell show that to use a similar approach for solving the Permutation Equivalence Problem, one would have to choose $G = (\mathrm{GL}_k(q) \times S_n) \rtimes \mathbb{Z}_2$. A criterion is given in Corollaries 1 and 2 of [14] for linear codes to be *HSP-hard*, meaning that it does not reveal any information about $\hat{H}$. The criterion asks that the code has very high rate, namely, that $q^{k^2} \leq n^{0.2n}$, and that the automorphism group of the code has very small degree.

The authors give some concrete examples of families of codes that satisfy the criterion. This is the case, for instance, of Alternant codes and Goppa codes. For these families, it is possible to give explicit bounds on the size of the automorphism group. Moreover, since these codes are subfield subcodes of Generalized Reed-solomon codes, the criterion can be satisfied by considering a generator matrix over the extension field and referring to the dimension of the "parent" code. This makes it so that the resulting code does not need to have the very high rate mentioned above, thus generating practical cryptographic instances.

The results just presented naturally extend to the Linear Equivalence Problem via the use of the closure. We note that these conditions, as interesting as they are from a theoretical point of view, are not necessary for our codes to offer quantum resistance. Indeed, no attack relying on the quantum Fourier sampling has been described so far in literature. Interestingly, the conditions are also not sufficient to claim post-quantum resistance since other attacks not based on quantum Fourier sampling might exist. This is for example the case of certain Goppa codes which satisfy the conditions described in [15] showing the impossibility of using the quantum Fourier sampling method, despite being attacked by the classical SSA because their hull has a small dimension.

7 Signature Scheme

The usual security notion that is required for signature schemes is Existential Unforgeability against Chosen-Message Attacks, or simply EUF-CMA. The attack model allows an adversary to perform polynomially-many queries to a signing oracle, in order to obtain valid message-signature pairs that could be used to extrapolate information. The adversary's goal is to be able to produce a single valid message-signature pair (different than those queried).

There is a standard conversion mechanism due to Fiat and Shamir, that allows to transform a canonical identification scheme into a signature scheme. The idea of the so-called Fiat-Shamir transform [18] is to make the protocol non-interactive by having the prover run the scheme with itself, using a random oracle to generate the challenge. The prover can then send the whole transcript $(\mathsf{cmt}_1, \dots, \mathsf{cmt}_t, \mathsf{rsp}_1, \dots, \mathsf{rsp}_t)$ as a signature to the verifier, who accepts if and only if $\mathcal{V}(\mathsf{pk}, \mathsf{cmt}_i, \mathsf{ch}_i, \mathsf{rsp}_i) = 1$ for all $i = 1, \dots, t$.

Table 4. The Fiat-Shamir Transform.

Private Key	A (signing) private key sk output by $\mathcal{K}(1^\lambda)$.
Public Key	The (verification) public key pk corresponding to sk.

SIGNER		VERIFIER
Input message $\boldsymbol{m}$		
$\mathsf{cmt}_i \leftarrow \mathcal{P}(\mathsf{sk}, \mathsf{pk}, \rho_i)$		
$\mathsf{ch} = \mathsf{H}(\boldsymbol{m}, \mathsf{cmt}_1, \dots, \mathsf{cmt}_t)$		
$\mathsf{ch}_i \in \{0,1\}^\ell \leftarrow \mathsf{ch}$		
$\mathsf{rsp}_i \leftarrow \mathcal{P}(\mathsf{sk}, \mathsf{pk}, \rho_i, \mathsf{cmt}_i, \mathsf{ch}_i)$		
$\sigma = (\mathsf{cmt}_1, \dots, \mathsf{cmt}_t, \mathsf{rsp}_1, \dots, \mathsf{rsp}_t)$	$\xrightarrow{\sigma}$	
		$\mathsf{ch} = \mathsf{H}(\boldsymbol{m}, \mathsf{cmt}_1, \dots, \mathsf{cmt}_t)$
		$\mathsf{ch}_i \in \{0,1\}^\ell \leftarrow \mathsf{ch}$
		$\{0,1\} \leftarrow \mathcal{V}(\mathsf{pk}, \mathsf{cmt}_i, \mathsf{ch}_i, \mathsf{rsp}_i)$

The following theorem was proved in [1] and states the security of the Fiat-Shamir transform in all generality.

Theorem 1. *Consider a non-trivial canonical identification protocol that is secure against impersonation under passive attacks. Then the signature scheme derived using the Fiat-Shamir transform is secure against chosen-message attacks in the random oracle model.*

In the attack scenario that includes a quantum adversary, able to make quantum queries to the random oracle, the Fiat-Shamir transform could in principle not suffice to guarantee security, as the strategy employed in the proof requires techniques that are not compatible (e.g. rewinding). As a consequence, Unruh designed an alternative transform [36], which is proved to be secure in the QROM.

The transform is considerably less practical than Fiat-Shamir, and this prompted a follow-up body of work trying to analyze the situation. Recently, two contributions [16, 25] appeared at CRYPTO 2019, explaining how it may be safe, in certain instances, to still employ Fiat-Shamir in the presence of a quantum adversary. In particular, in [16], the case of lattice signatures is analyzed explicitly, and the authors show that popular schemes, such as those based on the work of Lyubashevsky [26], satisfy the *collapsing* property necessary to achieve existential unforgeability in the QROM. This is done by introducing a (rather plausible) assumption, which is justified by the authors, mentioning that the separation between the collision resistance and collapsingness properties is usually only artificial. As a matter of fact, the former is already a feature in the majority of Sigma protocols that are used with Fiat-Shamir, since it is necessary to guarantee unforgeability, and our scheme is no exception. Following the arguments detailed in Section 4, we can argue that applying Fiat-Shamir to the LESS identification scheme is enough to preserve EUF-CMA security in the QROM.

8 Concrete Instances

In this section we present concrete instances of the LESS protocol, as well as a thorough comparison with the state of the art of code-based signatures. To highlight the novelty of our approach, we remind the reader that all existing schemes in literature are based on the traditional method in code-based cryptography, which relies on the hardness of the syndrome decoding problem.

Identification Schemes. The credit for the first code-based identification scheme is attributed to Stern [35]. The protocol, proposed in 1989, is a very simple 3-pass scheme with three commitments, and thus a cheating probability of 2/3, which in turn means the number of rounds necessary to guarantee security is quite high. Since the size of the public key is also very large, the scheme is quite impractical, and remains in literature mostly as a reference. The scheme was then marginally improved by Véron [38], using a slightly different formulation for the private key. In 2010, Cayrel, Véron and El Yousfi introduce a new scheme [9] with a few interesting modifications, such as the use of q-ary codes and a 5-pass framework, leading to a cheating probability is $q/(2q-1)$ which, for large enough values of q, can be approximated as $1/2$. It follows that, despite the large alphabet size, the scheme performs better than its predecessors. The entire line of work can be further improved by using circulant matrices, as shown in [19], a variation of Stern's scheme instantiated with quasi-cyclic codes, and later by Aguilar, Gaborit and Schrek [2]. The latter, a 5-pass scheme similar to [9], is usually regarded as the most efficient proposal to obtain a signature scheme from an identification scheme. Yet, as we will see, the communication cost is still very high, leading to an impractical signature size.

Other Approaches. Two schemes have recently come to attention as promising solutions for code-based signatures. Wave [13] describes a family of trapdoor one-way preimage sampleable functions, following the CFS framework and utilizing a new class of codes known as *Generalized* $(U \mid U+V)$ *Codes* to sample preimages of high weight, rather than low weight as usual. This novel approach is extremely interesting, but is still far from practical, leading to a scheme with a huge public key (about 4Mb) and a high-complexity signing algorithm (in the order of λ^3 for a security level of λ bits, as mentioned by the authors). Durandal [3] obtains a signature scheme applying the Fiat-Shamir transform to an identification scheme using codes in the *rank metric*. The scheme is based on the framework of Schnorr [30], successfully exploited by Lyubashevsky for the lattice case [26], and obtains relatively small keys and signature sizes. However, there are some concerns about security, mostly due to the lack of an explicit proof of leakage immunity and to a security reduction that relies on a new ad-hoc problem which is rather convoluted and not so well-studied.

8.1 Choice of Parameters

We now provide some concrete instances of the scheme, which we depict in Tab. 5. In light of what explained in Section 6, our main concern is the classical security, so we choose system parameters to achieve 128-bit security against Leon's algorithm and SSA.

Table 5. Proposed LESS instances, targeting 128-bit security.

	n	k	q	Type
LESS-I	54	27	53	MONO
LESS-II	106	45	7	MONO
LESS-III	60	25	31	PERM

Clearly, to instantiate the scheme we need to choose a public code which does not allow for an easy solution of the corresponding code equivalence problem. The first and most natural approach is to rely on the hardness of Linear Equivalence by using random codes over $\mathbb{F}_q$ with $q \geq 5$, and choose n and k such that the complexities of Leon's algorithm and SSA are above the desired security level. LESS-I and LESS-II instances have been designed with this criteria; in the last column of Table 5 we remark the fact that monomial matrices are used in the protocol. In particular, LESS-I parameters have been obtained with the goal of optimizing the trade-off between security and performance, by looking for the triplet of values (n, k, q) that minimizes the (maximum) communication cost per round and, at the same time, guarantees that the complexities of both Leon's algorithm and SSA are above the desired security level. Note that this is not the case in some of the previous works such as [9], where the *average* communication cost is considered, and the average is taken over the cost of different responses. However, when designing a signature scheme, one is only interested in the maximum size of the signature, and therefore we deem more relevant to take into the account the maximum cost for each round. On the other hand, LESS-II parameters have been obtained by seeking the best trade-off between the scheme security and the computational efficiency of the algebra in the underlying finite field: for this reason, the field size is relatively small (i.e., $q = 7$ versus $q = 53$ for LESS-I) and of practical use.

The final choice, aimed at obtaining a performance advantage, is to restrict the scheme to permutations. In this case, in fact, the communication cost is reduced by the amount of bits necessary to transmit the scaling factors in each monomial matrix. However, to provide security, random codes are no longer enough, since, as we have seen, they have usually a very small hull. Therefore, it becomes necessary to choose a weakly-self dual code. It is possible to show that such a code can be generated in polynomial time. We call this parameter set LESS-III, and remark the fact that it uses only permutations in the last column of Table 5.

8.2 Performance and Comparison

The maximum communication cost per round is calculated as follows. We denote by l_{Hash} and l_{Seed} the sizes of, respectively, a hash and a seed for a pseudorandom generator. In our scheme, the commitment is a hash value (thus, requiring l_{Hash} bits), and the challenge is a single bit. When $b = 0$, the reply is a random monomial matrix and can be compactly transmitted by sending the corresponding seed. This trick however cannot be applied in the case $b = 1$ which, requires the transmission of $n\big(\lceil \log_2 n \rceil + \lceil \log_2 (q-1) \rceil\big)$ bits. Then, the maximum communication cost per round is

$$l_{\mathsf{Hash}} + 1 + \max\left\{ n\big(\lceil \log_2 n \rceil + b\lceil \log_2 (q-1) \rceil\big), l_{\mathsf{Seed}} \right\},$$

where $b = 0$ or 1 depending on whether permutations or monomials are used.

Table 6. Comparison between code-based signature schemes obtained from identification schemes, for 128-bit security. All sizes in bits, except where indicated.

	Véron [38]	CVE [9]	AGS [2]	LESS-I	LESS-II	LESS-III
Public Matrix	262,144	86,528	599	8,748	14,310	7,500
Public Key	1,024	832	599	8,748	14,310	7,500
Max. Comm. Cost per Round	2,434	3,593	2,792	777	1,189	489
Number of rounds	219	129	128	128	128	128
Signature size (kB)	66.63	57.94	44.67	12.43	19.02	7.82

Note that, in order to have a fair comparison, we had to scale up parameters for all the compared schemes, since those were given according to a variety of different metrics (none of which were sufficient to guarantee a secure signature scheme).This means for example that we require 128 bits of security against impersonation (commonly given at 2^{-16}), and assume that hash digests and seeds are 128 bits long. Following the suggestion of [20, Remark 2], we instantiate Fiat-Shamir with a number of rounds equal to the desired security level (in this case 128). The resulting signature scheme achieves 128-bit security with a signature size which, with respect to the AGS scheme, gets reduced by a factor which ranges from 57% for LESS-II to 82% for LESS-III.

Regarding a comparison with the two new approaches, the numbers are as follows. For Wave, the key size is given by $0.368n^2$ bits and the signature size by $2n$. The authors suggest using a code of length $n = 9,078$, which leads to 30,326,911 bits of public key, i.e. roughly 3.8 MB, and 2.2 kB of signature. Durandal features more practical sizes: two sets of parameters are proposed, the smallest of which has 121,961 bits of public key and 32,514 bits of signature, which corresponds to approximately 15 kB and 4 kB, respectively. The proposed LESS instances feature a much smaller public key, while the signature size is only a few times bigger.

9 Conclusion

In this paper, we have presented LESS, a new code-based signature scheme derived from a zero-knowledge identification scheme. Our protocol is based on an innovative use of a long-standing problem in code-based cryptography, the Code Equivalence problem. Rather than looking at this in the context of McEliece-like encryption, in fact, our scheme exploits the action of linear isometries on codes as a stand-alone tool to provide security. This problem and its hardness have been thoroughly studied over the years, and therefore it is possible to give an accurate security assessment.

Since our scheme doesn't involve syndromes and doesn't require any hardness assumptions or security results connected to decoding, we are able to choose, to our advantage, ad hoc parameters which would not normally be usable within the traditional code-based framework (due to poor error-correction capability). As a result, for instance, all the codes considered have very short lengths, which means the sizes of the objects involved in the signature scheme can be kept small. The public keys in our protocol are among the smallest in code-based cryptography, without needing to resort to families with special structure such as Quasi-Cyclic (QC) codes. Furthermore, the size of our signatures is as short a few Kilobytes (less than 8 for the LESS-III parameter set), in line with the major post-quantum signature schemes. Our design performs better than the traditional solutions based on identification schemes in nearly every aspect, and compares very well with modern approaches to code-based signatures such as Wave [13] and Durandal [3]. Finally, we expect to see very good performance from the computation point of view, due to the simplicity of the underlying arithmetic. Naturally, a full-fledged and optimized implementation will be the topic of a follow-up work. To conclude, we see our work as but the first step in paving the way for a new, very promising trend in code-based cryptography.

Acknowledgments

Jean-François Biasse was supported by the U.S. National Science Foundation under grant 1839805, and grant 1846166, by NIST under grant 60NANB17D184, and by a Seed Grant of the Florida Center for Cyber- security.
Edoardo Persichetti and Paolo Santini were supported by the U.S. National Science Foundation under grant 1906360.

References

[1] M. Abdalla et al. "From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security". In: *EUROCRYPT*. Springer. 2002, pp. 418–433.

[2] C. Aguilar, P. Gaborit, and J. Schrek. "A new zero-knowledge code based identification scheme with reduced communication". In: *2011 IEEE Information Theory Workshop*. Oct. 2011, pp. 648–652. DOI: `10.1109/ITW.2011.6089577`.

[3] N. Aragon et al. "Durandal: a rank metric based signature scheme". In: *EUROCRYPT*. Springer. 2019, pp. 728–758.

[4] M. Bardet, A. Otmani, and M. Saeed-Taha. "Permutation Code Equivalence is Not Harder Than Graph Isomorphism When Hulls Are Trivial". In: *IEEE ISIT 2019*. July 2019, pp. 2464–2468.

[5] M. Bellare, B. Poettering, and D. Stebila. "From identification to signatures, tightly: A framework and generic transforms". In: *ASIACRYPT*. Springer. 2016, pp. 435–464.

[6] C. H. Bennett. "Time/space trade-offs for reversible computation". In: *SIAM Journal on Computing* 18.4 (1989), pp. 766–776.

[7] D. J. Bernstein. "Grover vs. mceliece". In: *International Workshop on Post-Quantum Cryptography*. Springer. 2010, pp. 73–80.

[8] D. J. Bernstein et al. "SPHINCS: Practical Stateless Hash-Based Signatures". In: *Advances in Cryptology – EUROCRYPT 2015*. Ed. by E. Oswald and M. Fischlin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 368–397.

[9] P.-L. Cayrel, P. Véron, and S. M. El Yousfi Alaoui. "A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem". In: *Selected Areas in Cryptography*. Springer Berlin Heidelberg, 2011, pp. 171–186. ISBN: 978-3-642-19574-7.

[10] *https://classic.mceliece.org/*.

[11] N. Courtois, M. Finiasz, and N. Sendrier. "How to Achieve a McEliece-Based Digital Signature Scheme". In: *ASIACRYPT*. 2001, pp. 157–174.

[12] I. Damgård. "On Σ-protocols". In: *Lecture Notes, University of Aarhus, Department of Computer Science* (2002).

[13] T. Debris-Alazard, N. Sendrier, and J.-P. Tillich. "Wave: A new family of trapdoor one-way preimage sampleable functions based on codes". In: *ASIACRYPT*. Springer. 2019, pp. 21–51.

[14] H. Dinh, C. Moore, and A. Russell. "Limitations of Single Coset States and Quantum Algorithms for Code Equivalence". In: *Quantum Info. Comput.* 15.3-4 (Mar. 2015), pp. 260–294. ISSN: 1533-7146.

[15] H. Dinh, C. Moore, and A. Russell. "McEliece and Niederreiter Cryptosystems That Resist Quantum Fourier Sampling Attacks". In: *CRYPTO 2011*. Ed. by P. Rogaway. Springer Berlin Heidelberg, 2011, pp. 761–779.

[16] J. Don et al. “Security of the Fiat-Shamir Transformation in the Quantum Random-Oracle Model”. In: *CRYPTO 2019*, pp. 356–383.

[17] L. D. Feo and S. D. Galbraith. “SeaSign: Compact Isogeny Signatures from Class Group Actions”. In: EUROCRYPT 2019 11478 (2019), pp. 759–789.

[18] A. Fiat and A. Shamir. “How to prove yourself: Practical solutions to identification and signature problems”. In: *CRYPTO*. Springer. 1986, pp. 186–194.

[19] P. Gaborit and M. Girault. “Lightweight code-based identification and signature”. In: *2007 IEEE International Symposium on Information Theory*. IEEE. 2007, pp. 191–195.

[20] S. D. Galbraith, C. Petit, and J. Silva. “Identification protocols and signature schemes based on supersingular isogeny problems”. In: *ASIACRYPT*. Springer. 2017, pp. 3–33.

[21] M. Girault. “A (non-practical) three-pass identification protocol using coding theory”. In: *AUSCRYPT*. Springer. 1990, pp. 265–272.

[22] L. Grover. “A Fast Quantum Mechanical Algorithm for Database Search”. In: *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*. STOC ’96. Philadelphia, Pennsylvania, USA: ACM, 1996, pp. 212–219. ISBN: 0-89791-785-5. DOI: 10.1145/237814.237866. URL: http://doi.acm.org/10.1145/237814.237866.

[23] J. Katz. *Digital signatures*. Springer Science & Business Media, 2010.

[24] J. Leon. “Computing automorphism groups of error-correcting codes”. In: *IEEE Transactions on Information Theory* 28.3 (May 1982), pp. 496–511. ISSN: 1557-9654. DOI: 10.1109/TIT.1982.1056498.

[25] Q. Liu and M. Zhandry. “Revisiting Post-quantum Fiat-Shamir”. In: *Advances in Cryptology - CRYPTO 2019*. 2019, pp. 326–355.

[26] V. Lyubashevsky. “Lattice signatures without trapdoors”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2012, pp. 738–755.

[27] *https://csrc.nist.gov/Projects/post-quantum-cryptography/Post-Quantum-Cryptography-Standardization*.

[28] E. Petrank and R. M. Roth. “Is code equivalence easy to decide?” In: *IEEE Transactions on Information Theory* 43.5 (Sept. 1997), pp. 1602–1604.

[29] M. A. Saeed. “Algebraic Approach for Code Equivalence”. In: *PhD thesis* (2017).

[30] C.-P. Schnorr. “Efficient signature generation by smart cards”. In: *Journal of cryptology* 4.3 (1991), pp. 161–174.

[31] N. Sendrier. “The Support Splitting Algorithm”. In: *Information Theory, IEEE Transactions on* (Aug. 2000), pp. 1193–1203. DOI: 10.1109/18.850662.

[32] N. Sendrier and D. E. Simos. “The Hardness of Code Equivalence over $\mathbb{F}_q$ and Its Application to Code-Based Cryptography”. In: *PQCrypto 2013*. Ed. by P. Gaborit. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 203–216.

[33] N. Sendrier and P. Symbolique. “On the Dimension of the Hull”. In: *SIAM Journal on Discrete Mathematics* 10 (Nov. 1995). DOI: 10.1137/S0895480195294027.

[34] P. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computing* 26.5 (1997), pp. 1484–1509.

[35] J. Stern. “A new identification scheme based on syndrome decoding”. In: *Advances in Cryptology — CRYPTO’ 93*. Ed. by D. R. Stinson. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 13–21.

[36] D. Unruh. “Non-interactive zero-knowledge proofs in the quantum random oracle model”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2015, pp. 755–784.

[37] D. Venturi. “Zero-knowledge proofs and applications”. In: *Lecture Notes, Sapienza University of Rome, Department of Computer Science* (2015).

[38] P. Véron. “Improved identification schemes based on error-correcting codes”. In: *Applicable Algebra in Engineering, Communication and Computing* 8.1 (Jan. 1997), pp. 57–69. ISSN: 1432-0622. DOI: 10.1007/s002000050053.