sciendo

TATRA MOUNTAINS
Mathematical Publications
DOI: 10.2478/tmmp-2022-0018
Tatra Mt. Math. Publ. **82** (2022), 53–64

AN ATTACK ON A NON-INTERACTIVE KEY EXCHANGE FROM CODE EQUIVALENCE

Edoardo Persichetti [1] — Tovohery Hajatiana Randrianarisoa[2] — Paolo Santini[3]

[1]Florida Atlantic University, Boca Raton, USA

[2]Umeå University, Umeå, SWEDEN

[3]Università Politecnica delle Marche, Ancona, ITALY

ABSTRACT. A recent paper by Zhang and Zhang claims to construct the first code-based non-interactive key exchange protocol, using a modified version of the Code Equivalence Problem. In this paper we explain why this approach is flawed. Namely, we describe an attack which involves only linear algebra and completely breaks the protocol with overwhelming probability. A simple Magma script confirms our results.

1. Introduction

The importance of public-key cryptography in modern society cannot be understated. Primitives such as key exchange mechanisms and digital signatures, in fact, are fundamental building blocks to guarantee confidentiality, authentication, and many other functionalities related to secure communication. With the threat of quantum computers looming on the horizon, the cryptographic community has begun a slow and careful transition to so-called *post-quantum* systems, i.e., protocols that are not affected by the currently-known quantum attacks. These generally include protocols whose hardness relies on problems related to Euclidean lattices, linear codes, isogenies on elliptic curves, and a few others. Efficient protocols for post-quantum encryption and signature functionalities exist, and are under evaluation for standardization by agencies such as the National Institute of Standards and Technology (NIST) (see [1]).

2020 Mathematics Subject Classification: 11T71, 94A60.
Keywords: Code Equivalence, Diffie-Hellman, Post-Quantum.
E. Persichetti is supported by NSF grant n. 1906360 and NSA grant n. H98230-22-1-0328.

At the same time, we still lack fully satisfactory options for static key exchange protocols such as Diffie-Hellman. In fact, to date, the only existing post-quantum constructions to implement this protocol are based on the hardness of finding isogenies between supersingular elliptic curves [10, 11]. This setting, however, is characterized by very complex arithmetic, which leads to slow performance results, at least when compared to the state of the art. What is worse, is the SIKE scheme has been completely broken by a recent, devastating attack [9], leaving the community with even fewer options.

A new line of work in post-quantum cryptography proposes to exploit the hardness of the *code equivalence problem*, which is the problem of determining whether two linear codes are equivalent via a linear isometry. The hardness of code equivalence has been studied through several works [5,7,12,14] and there is no known vulnerability against quantum attackers. Thus, the problem appears to be suitable to build post-quantum cryptosystems. Code equivalence has been used successfully to construct signature schemes (e.g. [4, 6, 8]), yet some other applications have so far been out of reach. For example, the non-commutative nature of the associated *group action* poses an obstacle towards developing a key exchange protocol.

In [15], the authors claim that, by using an alternative version of the code equivalence problem, it is possible to develop a non-interactive key-based protocol, à-la Diffie-Hellman. Their method, in a nutshell, proposes to split the action of the permutation defining the equivalence into two, disjoint sets, so that composing permutations from the two sets is commutative. In other words, the permutations picked by Alice and Bob have orthogonal fixed points: Alice leaves unchanged the coordinates in even positions, while Bob leaves unchanged those in odd positions. By doing this, Alice and Bob end up with the same code, which is then used to compute the same shared key. Note that the associated hard problem is not code equivalence anymore, due to the presence of fixed points: the authors in [15] call it sub-LCE and propose a reduction to the canonical code-equivalence problem.

In this paper, we show that this approach is severely flawed, and that the scheme is completely insecure. Namely, we present an attack which can retrieve, with basic linear algebra, the secret transformations picked by Alice and Bob. The attack exploits the presence of fixed points, and uses them to first retrieve the change of basis which is necessary to hide the permutation action. Once it has been obtained, retrieving the transformations is a trivial task. We have tested our attack, using Magma, on the parameters recommended in [15]: our implementation can break all the proposed instances in just a few milliseconds. This shows that the scheme in [15] is completely insecure.

The paper is organized as follows.

In Section 2, we establish the notation we use throughout the paper and recall basic concepts about code equivalence. In Section 3, we briefly present the scheme proposed in [15]. The proposed attack is described and benchmarked in Section 4. Notice that the existence of this attack does not imply that code-equivalence is not secure; indeed, as we show in Section 4, the reduction considered in [15] is not meaningful for an average code-equivalence instance. Finally, in Section 5, we draw some concluding remarks.

2. Background

We begin by clarifying the notation that will be used throughout this manuscript and provide basic notions about linear codes and code equivalence.

2.1. Notation

We denote with $\mathbb{F}_q$ the finite field of order q, with q being a prime power, as is customary. We will use $\mathbb{F}_q^*$ to indicate the multiplicative group of $\mathbb{F}_q$. We write $\mathsf{GL}_k(q)$ for the set of invertible $k \times k$ matrices with elements in $\mathbb{F}_q$, or simply GL_k when the finite field is implicit. Let S_n be the Symmetric group of order n, i.e., the group of permutations over n elements. Given a vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ and a permutation $\pi \in \mathsf{S}_n$, we write the action of π on $\boldsymbol{x}$ as $\pi(\boldsymbol{x}) = (x_{\pi(1)}, \ldots, x_{\pi(n)})$. Note that a permutation can equivalently be described as an $n \times n$ matrix with exactly one 1 per row and column. Analogously, for *linear isometries*, i.e., transformations $\tau = (\boldsymbol{v}; \pi) \in \mathbb{F}_q^{*n} \rtimes \mathsf{S}_n$, we write the action on a vector $\boldsymbol{x}$ as $\tau(\boldsymbol{x}) = (v_1 x_{\pi(1)}, \ldots, v_n x_{\pi(n)})$. Then, we can also describe these in matrix form as a product $\boldsymbol{Q} = \boldsymbol{DP}$, where $\boldsymbol{P}$ is an $n \times n$ permutation matrix and $\boldsymbol{D} = \{d_{ij}\}$ is an $n \times n$ diagonal matrix with diagonal entries in $\mathbb{F}_q^*$. We denote with $\mathsf{M}_n(q)$ the set of such matrices, usually known as *monomial* matrices, or again simply M_n when the context is clear.

2.2. Linear codes

An $[n, k]$-*linear code* $\mathcal{C}$ of length n and dimension $k \leq n$ over $\mathbb{F}_q$ is a k--dimensional vector subspace of $\mathbb{F}_q^n$. It can be represented by a full-rank matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$ with rank k, called *generator matrix*, whose rows form a basis for the vector space, i.e., $\mathcal{C} = \{\boldsymbol{uG},\ \ \boldsymbol{u} \in \mathbb{F}_q^k\}$. Alternatively, a linear code can be represented as the kernel of a full-rank matrix $\boldsymbol{H} \in \mathbb{F}_q^{(n-k) \times n}$, known as *parity-check matrix*, i.e., $\mathcal{C} = \left\{\boldsymbol{x} \in \mathbb{F}_q^n : \boldsymbol{Hx}^T = 0\right\}$. For both representations, up to some permutations of columns, there exists a standard choice, called *systematic form*, which corresponds, respectively, to $\boldsymbol{G} = (\boldsymbol{I}_k \mid \boldsymbol{M})$ and $\boldsymbol{H} = (-\boldsymbol{M}^T \mid \boldsymbol{I}_{n-k})$. Generator (resp. parity-check) matrices in systematic form can be obtained very

simply by calculating the row-reduced echelon form starting from any other generator (resp. parity-check) matrix. The parity-check matrix is important also as it is a generator for the *dual code*, defined as the set of words that are orthogonal to the code, i.e.,

$$\mathcal{C}^{\perp} = \left\{ \boldsymbol{y} \in \mathbb{F}_q^n \ : \ \forall \boldsymbol{x} \in \mathcal{C}, \quad \boldsymbol{x} \cdot \boldsymbol{y}^T = 0 \right\}.$$

Note that, if $\mathcal{C}$ has length n and dimension k, then its dual $\mathcal{C}^{\perp}$ has dimension $n - k$.

The concept of *equivalence* between two codes is central to coding theory. Indeed, two equivalent codes have the same weight enumerator function, which implies that in principle they share the same error correction capability. For our purposes, we present the following formulation for code equivalence.

Definition 2.1 (Code Equivalence)**.** We say that two linear codes $\mathcal{C}$ and $\mathcal{C}'$ are *equivalent*, and write $\mathcal{C} \sim \mathcal{C}'$, if there exist a field automorphism $\alpha \in \mathsf{Aut}(\mathbb{F}_q)$ and a linear isometry $\tau = (\boldsymbol{v}; \pi) \in \mathbb{F}_q^{*n} \rtimes \mathsf{S}_n$ that map $\mathcal{C}$ into $\mathcal{C}'$, i.e., such that

$$\mathcal{C}' = \tau\big(\alpha(\mathcal{C})\big) = \left\{ \boldsymbol{y} \in \mathbb{F}_q^n \ : \boldsymbol{y} = \tau\big(\alpha(\boldsymbol{x})\big), \ \boldsymbol{x} \in \mathcal{C} \right\}.$$

The notion presented above is the most general, and is known as *semilinear equivalence*. However, as explained for instance in [4, 8], the role of the field automorphism in cryptographic applications is minimal, and does not contribute significantly to a scheme's security. Moreover, cryptographic protocols based on code equivalence (including the work of [15]) overwhelmingly utilize prime fields, so that this notion is not meaningful anyway. As a consequence, in this paper, we restrict our attention to the case in which the codes are mapped only through a monomial transformation. This notion is usually known simply as *linear equivalence*. If, instead, one searches for a monomial transformation which corresponds to a permutation (i.e., all scaling coefficients are equal to 1, meaning the diagonal matrix $\boldsymbol{D}$ of the scaling coefficients is the identity), then the notion is known as *permutation equivalence*.

Clearly, if $\mathcal{C}$ and $\mathcal{C}'$ are two codes with generator matrices $\boldsymbol{G}$ and $\boldsymbol{G}'$, respectively, it holds that

$$\mathcal{C} \sim \mathcal{C}' \iff \exists (\boldsymbol{S}; \boldsymbol{Q}) \in \mathsf{GL}_k \rtimes \mathsf{M}_n \text{ s.t. } \boldsymbol{G}' = \boldsymbol{SGQ}.$$

This characterization leads to the following computational problem.

Problem 1 (Linear Code Equivalence (LCE))**.** Let $\boldsymbol{G}, \boldsymbol{G}' \in \mathbb{F}_q^{k \times n}$ be two generator matrices for two linearly equivalent codes $\mathcal{C}$ and $\mathcal{C}'$. Find two matrices $\boldsymbol{S} \in \mathsf{GL}_k$ and $\boldsymbol{Q} \in \mathsf{M}_n$ such that $\boldsymbol{G}' = \boldsymbol{SGQ}$.

3. System Description

In this section we recall the details of the scheme proposed by [15]. We first note that a straightforward, direct translation of the well-known Diffie-Hellman key exchange protocol cannot work in the code-based setting. Indeed, as explained in [4], one can describe the code-equivalence setting in terms of group actions: a public generator matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$ is interpreted as the group generator, while the group action associated to a monomial matrix $\boldsymbol{Q} \in \mathsf{M}_n$ is obtained as

$$\boldsymbol{G}' = \boldsymbol{S}\boldsymbol{G}\boldsymbol{Q}, \quad \text{where} \quad \boldsymbol{S} \in \mathsf{GL}_k.$$

It is immediately seen that the codes $\mathcal{C}$ and $\mathcal{C}'$, generated respectively by $\boldsymbol{G}$ and $\boldsymbol{G}'$, are equivalent according to Definition 2.1. However, differently from the group actions used in the original Diffie-Hellman key exchange and in the Elliptic Curve counterpart, the code-equivalence group action is not commutative. In fact, as is widely known, permutations in general do not commute, and thus neither do monomial matrices. This means that, for two monomials

$$\boldsymbol{Q}_a, \boldsymbol{Q}_b \in \mathsf{M}_n, \quad \text{one normally has} \quad \boldsymbol{Q}_a\boldsymbol{Q}_b \neq \boldsymbol{Q}_b\boldsymbol{Q}_a.$$

So, if Alice and Bob compute and exchange

$$\boldsymbol{G}_a = \boldsymbol{S}_a\boldsymbol{G}\boldsymbol{Q}_a \quad \text{and} \quad \boldsymbol{G}_b = \boldsymbol{S}_b\boldsymbol{G}\boldsymbol{Q}_b,$$

then the reconciliation phase would fail with very high probability, because the code generated by

$$\boldsymbol{G}_{ka} = \boldsymbol{G}_b\boldsymbol{Q}_a = \boldsymbol{S}_b\boldsymbol{G}\boldsymbol{Q}_b\boldsymbol{Q}_a$$

would be different from that generated by

$$\boldsymbol{G}_{kb} = \boldsymbol{G}_a\boldsymbol{Q}_b = \boldsymbol{S}_a\boldsymbol{G}\boldsymbol{Q}_a\boldsymbol{Q}_b.$$

To achieve the commutativity property in the code-equivalence setting, the authors in [15] propose to restrict the choice on $\boldsymbol{Q}_a$ and $\boldsymbol{Q}_b$ to special transformations. Namely, they split the action of the monomial matrix into two parts, where only the even- or odd-numbered columns are affected. To do this, they define the sets $\mathsf{EM}_n(q)$ and $\mathsf{OM}_n(q)$ (that we will shorten to EM_n and OM_n), which are the subgroups of M_n containing matrices that have a 1 in the odd--numbered and even-numbered elements of the diagonal, respectively. This means that multiplying a matrix $\boldsymbol{G}$ by an element of EM_n, only the even-numbered columns of $\boldsymbol{G}$ are affected (i.e., permuted and scaled), and viceversa. The set $\mathsf{sub} - \mathsf{M}_n$ is then defined as their union, and leads to the following problem.

Problem 2 (sub-LCE)**.** Let $\boldsymbol{G}, \boldsymbol{G}' \in \mathbb{F}_q^{k\times n}$ be two generator matrices for two linearly equivalent codes $\mathcal{C}$ and $\mathcal{C}'$. Find two matrices $\boldsymbol{S} \in \mathsf{GL}_k$ and $\boldsymbol{Q} \in \mathsf{sub-M}_n$ such that

$$\boldsymbol{G}' = \boldsymbol{SGQ}.$$

The key exchange protocol described in Section 3.1 of [15] is summarized in the following table.

Table 1. The Key Exchange Protocol of [15].

Alice		Bob
$\boldsymbol{Q}_a \xleftarrow{\$} \mathsf{OM}_n$		$\boldsymbol{Q}_b \xleftarrow{\$} \mathsf{EM}_n$
$\boldsymbol{S}_a \xleftarrow{\$} \mathsf{GL}_k$		$\boldsymbol{S}_b \xleftarrow{\$} \mathsf{GL}_k$
$\boldsymbol{G}_a = \boldsymbol{S}_a\boldsymbol{G}\boldsymbol{Q}_a$		$\boldsymbol{G}_b = \boldsymbol{S}_b\boldsymbol{G}\boldsymbol{Q}_b$
	$\boldsymbol{G}_a \rightleftharpoons \boldsymbol{G}_b$	
$\boldsymbol{G}_{ka} = \boldsymbol{G}_b\boldsymbol{Q}_a$		$\boldsymbol{G}_{kb} = \boldsymbol{G}_a\boldsymbol{Q}_b$
$K = f(\boldsymbol{G}_{ka})$		$K = f(\boldsymbol{G}_{kb})$

In the protocol above, the matrix $\boldsymbol{G}$ is public information, and f denotes a function that is used by both parties to derive the shared secret; specifically, this consists of computing the systematic form of the matrix and then extracting the secret in the desired format. The correctness of the scheme is based on the fact that, when

$$Q_a \in \mathsf{OM}_n \quad \text{and} \quad Q_b \in \mathsf{EM}_n,$$

then trivially

$$Q_aQ_b = Q_bQ_a.$$

The parameters recommended in [15] are reported in Table 2. The security level of the scheme is given in $\log_2$ of binary operations, as customary. Parameters are chosen with respect to the best generic attacks, by which we mean attacks that work on the code equivalence problem in all generality (as opposed to "specialized" attacks such as SSA [14]). We do not go into details here, but refer the interested reader to [7, 12] instead.

TABLE 2. Parameters of the key-exchange protocol proposed in [15].

Security (bits)	q	n	k
128	31	400	194
192	31	600	294
256	31	800	394

4. The Attack

In this section we describe an attack to the key exchange protocol proposed in [15]. Without loss of generality, let us restrict our attention to Alice's side (the case of Bob is entirely equivalent). In the scheme, Alice's private information consists of the matrices $\boldsymbol{S}_a \in \mathsf{GL}_k$ and $\boldsymbol{Q}_a \in \mathsf{OM}_n$, where $\boldsymbol{Q}_a$ only affects the odd-numbered columns of $\boldsymbol{G}$. Assume for simplicity that the code parameters are such that $k = n/2$. Then, one can generate a new instance $\hat{\boldsymbol{G}}_a = \boldsymbol{S}_a\hat{\boldsymbol{G}}$, where $\hat{\boldsymbol{G}}_a$ and $\hat{\boldsymbol{G}}$ are obtained by removing all the odd-numbered columns from $\boldsymbol{G}_a$ and $\boldsymbol{G}$, respectively. This is because the monomial transformation does not affect the even-numbered columns. Now, since these are square matrices, solving the associated system of equations is immediate, yielding $\boldsymbol{S}_a$ with overwhelming probability. Once $\boldsymbol{S}_a$ is known, finding $\boldsymbol{Q}_a$ from the original equation becomes trivial.

Note that this attack also works if $k \neq n/2$. Indeed, in case $k < n/2$, the system of equations is overdetermined and, with overwhelming probability, will yield a single solution. Instead, if $k > n/2$, it may be that the system allows for multiple solutions, and finding the correct one can be made very expensive by choosing parameters to this extent. To be sure, the authors do mention an attack which is analogous to this one in Section 3.2.3, and suggest to choose[1] $k > n/2$ to avoid it. In fact, with this parameter choice, the system has a number of free variables equal to $(2k-n)k/2 > 0$. However, the authors crucially overlook the fact that the same attack can be performed on the dual of the specified codes. To be precise, consider a code generated by $\boldsymbol{G}$ and let $\boldsymbol{H}$ be a generator for its dual. Then, it is a well-known fact that any matrix of the form $\boldsymbol{THQ}^{-T}$ is a generator for the dual of the code generated by $\boldsymbol{SGQ}$, where

$$\boldsymbol{S} \in \mathsf{GL}_k,\ \boldsymbol{T} \in \mathsf{GL}_{n-k} \quad \text{and} \quad \boldsymbol{Q} \in \mathsf{M}_n,$$

[1]Note that, despite their own recommendation, the authors propose wrong parameters, as shown in Table 2, taking e.g., $n = 400, k = 194$.

and we denote by $\boldsymbol{Q}^{-T}$ the matrix $(\boldsymbol{Q}^{-1})^T$, for short. Using this, an attacker trying to recover Alice's information can write a new instance using the dual codes, as follows. First, the attacker computes the matrix $\boldsymbol{H}$ from $\boldsymbol{G}$, and after receiving $\boldsymbol{G}_a$, he computes the corresponding parity-check matrix $\boldsymbol{H}_a$. Then, it must be that

$$\boldsymbol{H}_a = \boldsymbol{T}\boldsymbol{H}\boldsymbol{Q}_a^{-T} \quad \text{for some} \quad \boldsymbol{T} \in \mathsf{GL}_{n-k}.$$

The attacker can now generate the instance $\hat{\boldsymbol{H}}_a = \boldsymbol{T}\hat{\boldsymbol{H}}$ as above, and solve for $\boldsymbol{T}$. In fact, this system is now overdetermined and, again, it will produce the correct solution with overwhelming probability. The matrix $\boldsymbol{Q}_a$ is then straightforwardly revealed, as before.

4.1. Formal Security Clarification

In Theorem 2 of [15], the authors show that an instance of the Linear Code Equivalence problem (Problem 2) can be reduced to a sub-LCE instance in polynomial time. Presumably, the intent is to give some guarantee about the hardness of the problem. We explain this reduction, and why this does not in fact say anything meaningful about hardness, for the proposed use case.

First, note that the setting of [15] is note unique, and their choice of splitting the action into odd- and even-numbered columns entirely arbitrary. Indeed, any other split into disjoint sets gives a completely equivalent formulation; for instance, splitting into left half and right half of the columns. This latter formulation makes it easier to present the reduction, and so we will use it[2] in the following explanation.

Now, suppose that $\boldsymbol{G}' = \boldsymbol{S}\boldsymbol{G}\boldsymbol{Q}$ for some invertible matrix $\boldsymbol{S}$ and monomial matrix $\boldsymbol{Q}$. By generating a random invertible $k \times k$ matrix $\boldsymbol{S}_2$ and a random $k \times n$ matrix $\boldsymbol{R}_1$, one can always construct two matrices

$$\tilde{\boldsymbol{G}} = \begin{pmatrix} \boldsymbol{G} & 0 \\ 0 & \boldsymbol{R}_1 \end{pmatrix} \qquad \tilde{\boldsymbol{G}}' = \begin{pmatrix} \boldsymbol{G}' & 0 \\ 0 & \boldsymbol{S}_2\boldsymbol{R}_1 \end{pmatrix}.$$

It is not difficult to show that

$$\begin{pmatrix} \boldsymbol{S} & 0 \\ 0 & \boldsymbol{S}_2 \end{pmatrix} \tilde{\boldsymbol{G}} \begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & \boldsymbol{I} \end{pmatrix} = \tilde{\boldsymbol{G}}'. \tag{1}$$

Thus an instance of the LCE problem always produces an instance of sub-LCE problem. One can check that a solution to Equation (1) will always produce a solution to $\boldsymbol{G}' = \boldsymbol{S}\boldsymbol{G}\boldsymbol{Q}$.

[2]Note that the authors themselves do this in their proof of Theorem 2.

By this reduction of an LCE problem to a sub-LCE problem, it is suggested that in general, it should be difficult to solve an instance of the sub-LCE problem. However, this does not tell us the whole story. The reduction method actually tells us that a sub-LCE instance in the form of Equation (1) is not easy to solve, hence we can say this is a worst-case scenario. However, from this we cannot tell anything about the average case and, as we have shown in the previous part, the sub-LCE problem can be efficiently solved with very high probability. To see why the previous attack does not work on the instance of Equation (1), one can try to apply the attack. For simplicity, we assume that $n = 2k$ and let

$$\tilde{\boldsymbol{G}} = \begin{pmatrix} \boldsymbol{G} & 0 \\ 0 & \boldsymbol{R}_1 \end{pmatrix} \qquad \tilde{\boldsymbol{G}'} = \begin{pmatrix} \boldsymbol{G}' & 0 \\ 0 & \boldsymbol{R}_2 \end{pmatrix}.$$

Suppose that

$$\boldsymbol{S}\tilde{\boldsymbol{G}} \begin{pmatrix} \boldsymbol{Q} & 0 \\ 0 & \boldsymbol{I} \end{pmatrix} = \tilde{\boldsymbol{G}'}.$$

Note that this implies that $\boldsymbol{S}$ is of the form

$$\boldsymbol{S} = \begin{pmatrix} \boldsymbol{S}_1 & 0 \\ 0 & \boldsymbol{S}_2 \end{pmatrix}.$$

Therefore the equation becomes

$$\begin{pmatrix} \boldsymbol{S}_1\boldsymbol{G}\boldsymbol{Q} & 0 \\ 0 & \boldsymbol{S}_2\boldsymbol{R}_1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{G}' & 0 \\ 0 & \boldsymbol{R}_2 \end{pmatrix}.$$

Since our attack consists first of considering the part where the columns are not permuted, then we take the right half of this equation. That gives us $\boldsymbol{S}_2\boldsymbol{R}_1 = \boldsymbol{R}_2$. This only helps us recover $\boldsymbol{S}_2$ but we do not have any information about $\boldsymbol{S}_1$ and therefore it will not be possible to recover $\boldsymbol{S}$.

To conclude, our attack cannot be used to solve a particular instance produced by the reduction of an LCE instance to a sub-LCE instance. In other words, our attack does not help solving the Linear Code Equivalence problem. However, for random instances of sub-LCE, our attack works with overwhelming probability.

4.2. Attack Performance

The attack we have described in the above sections makes use of basic linear algebra, so its time complexity is clearly below the claimed security level. To give a practical evidence about the effectiveness of our attack, we consider the timings

obtained with a simple Magma implementation of the attack. The following times were obtained by considering the average over 100 runs of the attack, running on a machine with 2.6 GHz 6-Core Intel Core i7 processor.

TABLE 3. Timings for the attack on the key-exchange protocol proposed in [15].

q	n	k	Time (ms)
31	400	194	12
31	600	294	37
31	800	394	73
31	2000	994	927

In Table 3, the first three rows correspond to the parameter sets chosen in [15] (see Table 2). In addition, we have included a fourth parameter set, that we have designed artificially, but in accordance with the methods employed in Zhang and Zhang's work. Note that even this extreme choice of parameters, leads to an attack that is tremendously efficient. This shows that the construction is fundamentally flawed, and it is unlikely to be repaired by different parameter choices.

5. Conclusions

Traditional code-based cryptography relies on the usual metric from coding theory, the Hamming metric, which provides interesting results in some settings (e.g., encryption [2, 3]) but is a limitation in many others (e.g., signature schemes). Using alternative metrics, such as the rank metric, has the potential to provide performance improvements, but occasionally comes with security shortcomings [13]; moreover, the underlying approach is still fundamentally the same, based on decoding hardness, and many limitations are still in place. In this light, the code equivalence problem represents an interesting new avenue in code-based cryptography, allowing access to frameworks and techniques that were not previously available. In particular, using code equivalence has so far been helpful in partially filling the gap for primitives in this area: the LESS scheme, in fact, shows great potential in this sense. Even more relevant would be to have a code-based static key exchange. The authors in [15] proposed a construction to achieve such a scheme, but crucially overlooked a vulnerability in their proposed methods. In this paper, we have described an attack on the construction, which shows that the scheme is broken beyond repair.

Thus, designing a Diffie-Hellman key exchange using code-based cryptographic tools remains an open problem.

REFERENCES

[1] 2017. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

[2] ALBRECHT, M. R. — BERNSTEIN, D. J. — CHOU, T.—CID, C. — GILCHER, J. — —LANGE, T. — MARAM, V. — VON MAURICH, I. — MISOCZKI, R.—NIEDERHAGEN, R. — PATERSON, K. G. — PERSICHETTI, E. — PETERS, C. — SCHWABE, P.— —SENDRIER, N.—SZEFER, J.—TJHAI, C. J.—TOMLINSON, M.—WANG, W.: *Classic McEliece: conservative code-based cryptography*, NIST Post-Quantum Standardization, 3rd Round, 2021.

[3] ARAGON, N. — BARRETO, P. S. L. M. — BETTAIEB, S. — BIDOUX, L.— BLAZY, O. – – DENEUVILLE J. C. — GABORIT, P. — GUERON, S. — GÜNEYSU, T. — MELCHOR, C. A. — MISOCZKI, R. — PERSICHETTI, E. — SENDRIER, N. — TILLICH, J. - P. — — VASSEUR, V. — ZÉMOR, G.: *BIKE: Bit Flipping Key Encapsulation*, NIST Post--Quantum Standardization, 3rd Round, 2021.

[4] BARENGHI, A.—BIASSE, J.-F.—PERSICHETTI, E.—SANTINI, P.: *LESS-FM: Fine--tuning signatures from the code equivalence problem*. In: *International Conference on Post-Quantum Cryptography*, (Jung Hee Cheon, Jean-Pierre Tillich, eds.) Lecture Notes in Comput. Sci. vol. 12841, Springer, Cham, Switzerland, 2021, pp. 23–43.

[5] BARENGHI, A.—BIASSE, J.-F.—PERSICHETTI, E.—SANTINI, P.: *On the computational hardness of the code equivalence problem in cryptography*, Adv. Math. Commun. (2022), Cryptology ePrint Archive. https://eprint.iacr.org/2022/967

[6] BARENGHI, A.—BIASSE, J.-F.—NGO, T.—PERSICHETTI, E.—SANTINI, P.: *Advanced signature functionalities from the code equivalence problem*, Int. J. Comput. Math.: Computer Systems Theory **7** (2022), no. 2, 102–128.

[7] BEULLENS, W.: *Not Enough LESS: An proved algorithm for solving code equivalence problems over* $\mathbb{F}_q$. In: *Selected Areas in Cryptography*: 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21–23, 2020, Revised Selected Papers. Springer, Cham, 2021, pp. 387–403.

[8] BIASSE, J.-F.—MICHELI, G.—PERSICHETTI, E.—SANTINI, P.: *LESS is More: Code-based Signatures Without Syndromes*. Progress in cryptology—AFRICACRYPT 2020, (A. Nitaj, A. Youssef, eds.), Lecture Notes in Comput. Sci. Vol. 12174, Springer, Cham, Switzerland, 2020, pp. 45–65.

[9] CASTRYCK, W.—DECRU, T.: *An efficient key recovery attack on SIDH* (preliminary version), Cryptology ePrint Archive, 2022.

[10] CASTRYCK, W.—LANGE, T.—MARTINDALE, C.—PANNY, L.—RENES, J.: *CSIDH: an efficient post-quantum commutative group action*, In: *ASIACRYPT '18*, *Lecture Notes in Comput. Sci. Vol. 11274*, Springer, Cham, Switzerland, 2018, pp. 395–427.

[11] JAO, D.—FEO, L. D.: *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies*. In: International Workshop on Post-Quantum Cryptography, *Lecture Notes in Comput. Sci. Vol. 7071*, Springer, Switzerland, Heidelberg, 2011, pp. 19–34.

[12] LEON, J.: *Computing automorphism groups of error-correcting codes*, IEEE Trans. Inform. Theory **28** (1982), no. 3, 496–511.

[13] SAMARDJISKA, S.—SANTINI, P.—PERSICHETTI, E.—BANEGAS, G.: *A reaction attack against cryptosystems based on LRPC codes*. In: International Conference on Cryptology and Information Security in Latin America, Progress in cryptology—LATINCRYPT 2019. *Lecture Notes in Comput. Sci. Vol. 11774*, Springer, Cham, 2019, pp. 197–216.

[14] SENDRIER, N.: *Finding the permutation between equivalent linear codes: The support splitting algorithm*, IEEE Trans. Inform. Theory **46** (2000), no. 4, 1193–1203.

[15] ZHANG, Z.—ZHANG, F.: *Code-based non-interactive key exchange can be made*. Cryptology ePrint Archive, Report 2021/1619, 2021. `https://ia.cr/2021/1619`.

Received July 19, 2022

Edoardo Persichetti,
Department of Mathematical Sciences
Florida Atlantic University
777 Glades Road
Boca Raton, FL, 33431
USA
E-mail: epersichetti@fau.edu

Tovohery Hajatiana Randrianarisoa
Department of Mathematics
Campus Umeå (Universitetsområdet)
Umeå
SWEDEN
E-mail: trandrianarisoa@fau.edu

Paolo Santini
Department of Telecommunications Engineering
Università Politecnica delle Marche
Via Brecce Bianche 12
Ancona, 60131
ITALY
E-mail: p.santini@staff.univpm.it