A novel attack to the permuted kernel problem

# A Novel Attack to the Permuted Kernel Problem

Paolo Santini, Marco Baldi, Franco Chiaraluce
Università Politecnica delle Marche
Ancona, Italy
{p.santini, m.baldi, f.chiaraluce}@univpm.it

***Abstract*—The Permuted Kernel Problem (PKP) asks to find a permutation of a given vector belonging to the kernel of a given matrix. The PKP is at the basis of PKP-DSS, a post-quantum signature scheme deriving from the identification scheme proposed by Shamir in 1989. The most efficient solver for PKP is due to a recent paper by Koussa et al. In this paper we propose an improvement of such an algorithm, which we achieve by considering an additional collision search step applied on kernel equations involving a small number of coordinates. We study the conditions for such equations to exist from a coding theory perspective, and we describe how to efficiently find them with methods borrowed from coding theory, such as information set decoding. We assess the complexity of the resulting algorithm and show that it outperforms previous approaches in several cases. We also show that, taking the new solver into account, the security level of some instances of PKP-DSS turns out to be slightly overestimated.**

***Index Terms*—Digital signatures, information set decoding, permuted kernel problem, post-quantum cryptography, PKP-DSS.**

## I. Introduction

One of the oldest paradigms to achieve digital signatures consists in converting a Zero-Knowledge Identification (ZK-ID) scheme into a signature scheme through the Fiat-Shamir approach [1]. In a ZK-ID protocol a *prover*, holding the secret key, proves their identity through an interactive procedure, by replying to random challenges provided by a *verifier*. Fiat-Shamir makes the protocol non interactive; in the resulting scheme, the signature corresponds to the transcript of the protocol, i.e, to the list of exchanged messages. Usually, in a ZK-ID scheme, the key pair is generated by choosing a random instance of some hard problem: no trapdoor is involved and, consequently, the security guarantees are rather strong.

However, with a straightforward application of Fiat-Shamir, the resulting signatures are normally rather large. For this reason, ZK-ID signatures have received little attention for many years. It seems, however, that this trend is changing, since several works describing modern ZK-ID signatures have recently appeared [2]–[8]. These schemes make use of several optimizations, ranging from simulating a multiparty computation phase [9] to using hash-based functions (e.g., PRNGs and tree structures), which can lead to compact signatures with essentially no impact on security. This renewed interest is also motivated by the fact that devising secure and efficient post-quantum digital signatures looks difficult, especially as concerns the possibility to achieve the advisable diversity with respect to the sole availability of schemes based on structured lattices [10], [11]. ZK-ID signatures actually represent a promising and concrete avenue in this direction.

In 1989, Shamir proposed a ZK-ID protocol based on the Permuted Kernel Problem (PKP) [12]. This protocol is at the core of PKP-DSS [6], a recently proposed signature scheme with competitive performance (e.g., public keys of 57 bytes, signatures of 20.5 kilobytes and constant time signing in 2.5 millions of cycles, for 128-bit security). The PKP, which has been extensively studied along the years [13]–[17], is an NP-hard problem [18] that asks to find the permutation of a given vector which belongs to the kernel of a given matrix. The state-of-the-art PKP solver analyzed in the recent paper [19], in a nutshell, works by first reducing the problem to a smaller instance of the same problem, which is then solved with a meet-in-the-middle search strategy. The complexity of such an algorithm has been considered to recommend parameters for PKP-DSS.

In this paper we improve upon the state-of-the-art solver for the PKP. Technically, our algorithm can be thought of as an improvement of the one in [19], where we include a filtering step to cut some of the elements in the initial lists. To do this, we need to find kernel equations which bind a small number of coordinates. A similar idea has already been briefly discussed in [15], [19]; in both those works, however, the authors conclude that such equations are extremely hard to find and that, in practice, cannot be exploited. We adopt a coding theory perspective and show that, instead, useful equations of this type can be efficiently found by exploiting Information Set Decoding (ISD) algorithms. The resulting solver runs in a time which is lower than that of [19] and can attack some of the instances recommended for PKP-DSS (namely, those for 128 and 192 bits of security) with a smaller complexity than that claimed in [6]. The performance of the proposed algorithm has been tested with a proof-of-concept software implementation, which is publicly available[1].

The paper is organized as follows. In Section II we settle the notation we use throughout the paper and provide some basic notions about linear codes. In Section III we briefly recall the definition of PKP and the algorithm in [19]. In Section IV we describe how to find kernel equations with the desired properties. In Section V we describe and analyze the new PKP solver. In Section VI we draw some conclusive remarks.

[1] https://github.com/secomms/pkpattack/

## II. Notation and preliminaries

In this section we define the notation we use throughout the paper and recall some basic notions about linear codes.

### A. Notation

We use $\mathbb{F}_q$ to denote the finite field with $q$ elements. Bold lowercase (resp., uppercase) letters indicate vectors (resp., matrices). Given $\mathbf{a}$ (resp., $\mathbf{A}$), $a_i$ (resp., $a_{i,j}$) denotes the entry in position $i$ (resp., the entry in the $i$-th row and $j$-th column). $\mathrm{GL}_{m,n}$ is the set of $m \times n$ matrices over $\mathbb{F}_q$ with full rank $\min\{m,n\}$. The identity matrix of size $n$ is indicated as $\mathbf{I}_n$, while $\mathbf{0}$ denotes the all-zero vector. Given a set $A$, $|A|$ denotes its cardinality (i.e., the number of elements) and $a \xleftarrow{\$} A$ means that $a$ is picked uniformly at random over $A$. Given a matrix $\mathbf{A}$ and a set $J$, $\mathbf{A}_J$ is the matrix formed by the columns of $\mathbf{A}$ that are indexed by $J$; analogous notation is used for vectors. We denote by $\mathsf{RREF}(\mathbf{A}, J)$ the algorithm that outputs $\mathbf{A}_J^{-1}\mathbf{A}$ if $\mathbf{A}_J$ is square and non singular, otherwise returns a failure. We use $S_n$ to denote the group of length-$n$ permutations. Given $\mathbf{a} = (a_1, \cdots, a_n)$ and $\pi \in S_n$, we write $\pi(\mathbf{a}) = \big(a_{\pi(1)}, \cdots, a_{\pi(n)}\big)$. Given $\mathbf{a}$, $\mathbf{b}$, we define $\mathbf{a} \cap \mathbf{b}$ as the set of entries which appear in both $\mathbf{a}$ and $\mathbf{b}$. For a vector $\mathbf{a} \in \mathbb{F}_q^n$ with no repeated entries, we define $\mathscr{S}_\ell(\mathbf{a})$ as the set of length-$\ell$ vectors with entries picked from those of $\mathbf{a}$. Notice that $|\mathscr{S}_\ell(\mathbf{c})| = \frac{n!}{(n-\ell)!}$.

### B. Linear codes

A linear code $\mathscr{C} \subseteq \mathbb{F}_q^n$ with dimension $k$ and redundancy $r = n-k$ is a linear $k$-dimensional subspace of $\mathbb{F}_q^n$. Any code admits two equivalent representations: a *generator matrix*, that is, any $\mathbf{G} \in \mathrm{GL}_{k,n}$ such that $\mathscr{C} = \{\mathbf{uG} \mid \mathbf{u} \in \mathbb{F}_q^k\}$, or a *parity-check matrix*, that is, any $\mathbf{H} \in \mathrm{GL}_{r,n}$ such that $\mathscr{C} = \{\mathbf{c} \in \mathbb{F}_q^n \mid \mathbf{cH}^\top = \mathbf{0}\}$ (where $\top$ denotes transposition). Given $\mathbf{x} \in \mathbb{F}_q^n$, its syndrome is $\mathbf{s} = \mathbf{xH}^\top$. The dual of $\mathscr{C}$, which we denote by $\mathscr{C}^\perp$, is the space generated by $\mathbf{H}$. For any codeword $\mathbf{c} \in \mathscr{C}$ and any $\mathbf{b} \in \mathscr{C}^\perp$, we have $\mathbf{cb}^\top = 0$. By support of a code we mean the set of indexes $i$ such that there is at least one codeword $\mathbf{c}$ with $c_i \neq 0$. A subcode $\mathscr{B} \subseteq \mathscr{C}$, with dimension $k'$, is a $k'$-dimensional linear subspace of $\mathscr{C}$. The number of such subcodes is counted by $\begin{bmatrix} k \\ k' \end{bmatrix}_q = \prod_{i=0}^{k'-1} \frac{1-q^{k-i}}{1-q^{i+1}}$.

## III. The Permuted Kernel Problem

The Permuted Kernel Problem (PKP) reads as follows.

**Problem III.1.** ***Permuted Kernel Problem (PKP)*** *Given $\mathbf{A} \in \mathbb{F}_q^{m \times n}$ with $1 \leq m < n$ and $\mathbf{c} \in \mathbb{F}_q^n$, find $\pi \in S_n$ such that $\pi(\mathbf{c})\mathbf{A}^\top = \mathbf{0}$.*

The problem is notably known to be NP-hard, via reduction from the Subset Sum Problem (SSP) [18]. In the following sections we briefly recall the features of the hardest PKP instances and recall the algorithm in [19], which is deemed as the currently known best solver for PKP.

**Remark 1.** *The PKP can be equivalently formulated as a codeword finding problem. In fact, Problem III.1 asks to find a codeword $\widetilde{\mathbf{c}} \in \mathscr{C}$, where $\mathscr{C}$ is the code having $\mathbf{A}$ as parity-check matrix, such that $\widetilde{\mathbf{c}} \in \mathscr{S}_n(\mathbf{c})$.*

### A. Considerations for practical hardness

As in all previous works [13]–[17], [19], we study the PKP under the conditions leading to the hardest instances. Namely, we consider $\mathbf{A}$ such that $\mathrm{rank}(\mathbf{A}) = m$, $\mathbf{c}$ with all distinct entries and consider parameters $q, n, m$ so that, on average, the problem has exactly one solution. To this end, we assume that the PKP instance is generated by first picking $\mathbf{A} \xleftarrow{\$} \mathrm{GL}_{m,n}$ and then by choosing a random vector $\widetilde{\mathbf{c}} \in \mathbb{F}_q^n$ with distinct entries and such that $\widetilde{\mathbf{c}}\mathbf{A}^\top = \mathbf{0}$. Then, we set $\mathbf{c} = \pi(\widetilde{\mathbf{c}})$, with $\pi \xleftarrow{\$} S_n$. Since $\mathbf{A}$ and $\widetilde{\mathbf{c}}$ are picked at random, on average we expect to have $\frac{|\mathscr{S}_n(\mathbf{c})|}{q^m} = \frac{n!}{q^m}$ solutions. Consequently, we consider $q, n, m$ such that $n!q^{-m} < 1$.

Basically any solver for the PKP considers that it is always possible to craft additional constraints binding the entries of $\widetilde{\mathbf{c}}$. Namely, we can exploit any relation of the form

$$\sum_{i=1}^{n} \tilde{c}_i{}^u = \sum_{i=1}^{n} c_i{}^u, \quad u \in \{1, \cdots, q-1\}. \tag{1}$$

However, for $u \geq 2$ the above expression is not linear in the unknowns $\tilde{c}_i$, so that only the case of $u = 1$ is employed.

Taking into account all the previous considerations, the PKP formulation in Problem III.1 can be slightly modified. Indeed, let $\mathbf{H} = \begin{pmatrix} \mathbf{A} \\ 1 \quad \cdots \quad 1 \end{pmatrix} \in \mathbb{F}_q^{r \times n}$, with $r = m + 1$. Then, solving the PKP corresponds to finding $\widetilde{\mathbf{c}} \in \mathscr{S}_n(\mathbf{c})$ such that

$$\widetilde{\mathbf{c}}\mathbf{H}^\top = \big(0, \cdots, 0, \sum_{i=1}^{n} c_i\big) = \mathbf{s}. \tag{2}$$

With overwhelming probability (approximately $1 - q^{-m}$), the all-ones vector is not a linear combination of the rows of $\mathbf{A}$, so that we can safely assume that $\mathbf{H}$ has full rank $r$.

Finally, we consider that to solve the PKP we can restrict our attention to a subset of the entries of $\widetilde{\mathbf{c}}$. Indeed, for any $\mathbf{B} \in \mathrm{GL}_{\ell,r}$ with $\ell \leq r$, it must be

$$\widetilde{\mathbf{c}}(\mathbf{BH})^\top = \widetilde{\mathbf{c}}\widetilde{\mathbf{H}}^\top = \mathbf{sB}^\top = \widetilde{\mathbf{s}}. \tag{3}$$

Let $J \subset \{1, \ldots, n\}$ of size $n - r$ such that $\mathbf{H}_{\{1,\cdots,n\}\setminus J}$ is non singular, and $\mathbf{B} = \mathbf{H}^{-1}_{\{1,\cdots,n\}\setminus J}$. Then, $\widetilde{\mathbf{H}} = \mathbf{BH} = \mathsf{RREF}(\mathbf{H}, \{1, \cdots, n\} \setminus J)$, from which

$$\tilde{c}_{i_u} = \tilde{s}_i - \sum_{j \in J} \tilde{c}_j \tilde{h}_{j,u}, \quad \{i_1, \cdots, i_r\} = \{1, \cdots, n\} \setminus J. \tag{4}$$

Hence, it is enough to find the entries of $\widetilde{\mathbf{c}}$ in the positions indexed by $J$ to retrieve the whole solution $\widetilde{\mathbf{c}}$.

**Remark 2.** *Adopting again a coding theory formulation, one can see the PKP as a syndrome decoding problem: given a parity-check matrix $\mathbf{H}$ and a syndrome $\mathbf{s}$ as in* (2)*, find a vector $\widetilde{\mathbf{c}} \in \mathscr{S}_n(\mathbf{c})$ whose syndrome is $\mathbf{s}$.*

### B. State-of-the-art solver for PKP

The currently known best solver for the PKP is Algorithm 1 in [19]. The algorithm works with three parameters $\ell, \ell_1, \ell_2 \in \mathbb{N}$, such that $1 \leq \ell \leq r$, $\ell_1, \ell_2 \geq 1$ and $\ell_1 + \ell_2 = n - r + \ell$. The procedure is initialized by choosing a matrix $\mathbf{B} \in \mathrm{GL}_{\ell,r}$ so

that $\widetilde{\mathbf{H}} = \mathbf{B}\mathbf{H}$ has support size $n-r+\ell$. To do this, we first compute $\mathbf{H}' = \mathsf{RREF}(\mathbf{H}, \{n-r+1, \cdots, n\})$ and then sets $\widetilde{\mathbf{H}}$ as the sub-matrix formed by the entries of $\mathbf{H}'$ in the first $\ell$ rows and the columns in positions $\{1, \cdots, n-r+\ell\}$. The same transformation is applied to $\mathbf{s}$, obtaining $\widetilde{\mathbf{s}} = \mathbf{s}\mathbf{B}^\top \in \mathbb{F}_q^\ell$. Then, we partition $\widetilde{\mathbf{H}}$ as $(\widetilde{\mathbf{H}}_1, \widetilde{\mathbf{H}}_2)$, where $\widetilde{\mathbf{H}}_1 \in \mathbb{F}_q^{\ell \times \ell_1}$ and $\widetilde{\mathbf{H}}_2 \in \mathbb{F}_q^{\ell \times \ell_2}$, and construct two lists

$$\mathcal{L}_1 = \left\{ (\mathbf{x}, \mathbf{x}\widetilde{\mathbf{H}}_1^\top) \middle| \, \mathbf{x} \in \mathscr{S}_{\ell_1}(\mathbf{c}) \right\},$$

$$\mathcal{L}_2 = \left\{ (\mathbf{y}, \widetilde{\mathbf{s}} - \mathbf{y}\widetilde{\mathbf{H}}_2^\top) \middle| \, \mathbf{y} \in \mathscr{S}_{\ell_2}(\mathbf{c}) \right\}.$$

Let $\mathcal{L} = \mathcal{L}_1 \bowtie \mathcal{L}_2$, where $\bowtie$ is computed as follows:

1) use an efficient search algorithm (e.g., permutation plus binary search) to find collisions, i.e., pairs $(\mathbf{x}, \mathbf{t}) \in \mathcal{L}_1$ and $(\mathbf{y}, \mathbf{v}) \in \mathcal{L}_2$ such that $\mathbf{t} = \mathbf{v}$;
2) keep only the collisions for which $\mathbf{x} \cap \mathbf{y} = \varnothing$.

By construction, $\mathcal{L} = \left\{ (\mathbf{x}, \mathbf{y}) \in \mathscr{S}_{\ell_1+\ell_2}(\mathbf{c}) \mid (\mathbf{x}, \mathbf{y})\widetilde{\mathbf{H}}^\top = \widetilde{\mathbf{s}} \right\}$. Then, we find $J$ of size $n-r$ so that $J \subseteq \{1, \cdots, n-r+\ell\}$ and $\mathbf{H}_{\{1,\cdots,n\}\setminus J}$ is non singular, compute $\widetilde{\mathbf{H}} = \mathsf{RREF}(\mathbf{H}, \{1, \cdots, n\} \setminus J)$ and use (4) to test each element in $\mathcal{L}$. Namely, for each $\mathbf{p} \in \mathcal{L}$, we use the entries of $\mathbf{p}_J$ as $\widetilde{\mathbf{c}}_J$ and see if the resulting $\widetilde{\mathbf{c}}$ belongs to $\mathscr{S}_n(\mathbf{c})$.

According to [19], the time complexity of the algorithm is given by

$$T(\ell_1, \ell_2) = \frac{n!}{(n-\ell_1)!} + \frac{n!}{(n-\ell_2)!} + \frac{(n!)^2 q^{n-r-\ell_1-\ell_2}}{(n-\ell_1)!(n-\ell_2)!}. \quad (5)$$

## IV. Finding subcodes with small support

Next we show that, differently from the claims in [6], [15], we can efficiently find kernel equations which involve a small number of coordinates. We first substantiate the existence of such equations with coding theory arguments, and then describe how to efficiently find them.

### A. Number of subcodes with small support

As shown above, we can see the matrix $\mathbf{H}$ of a given PKP instance as the parity-check matrix of some linear code $\mathscr{C}$ with redundancy $r$. The space generated by the rows of $\mathbf{H}$ corresponds to $\mathscr{C}^\perp$, and a set of $d \le r$ independent equations from this space, involving $w$ coordinates, is a basis for a subcode $\mathscr{B} \subseteq \mathscr{C}^\perp$ with dimension $d$ and support size $w$. For a random code, the number of such subcodes can be estimated as follows.

**Theorem IV.1.** *For a code $\mathscr{C} \subseteq \mathbb{F}_q^n$, we define $\mathcal{A}_{w,d}(\mathscr{C})$ as the set of subcodes of $\mathscr{C}$ with dimension $d$ and support size $w$. Let $N_{w,d}$ be the average value of $|\mathcal{A}_{w,d}(\mathscr{C})|$, when $\mathscr{C}$ is picked at random among all codes with dimension $k$. Then $\widecheck{N}_{w,d} \le N_{w,d} \le \widehat{N}_{w,d}$, with*

$$\widecheck{N}_{w,d} = \binom{n}{w} (q^d - 1)^{w-d} \frac{\begin{bmatrix} k \\ d \end{bmatrix}_q}{\begin{bmatrix} n \\ d \end{bmatrix}_q},$$

$$\widehat{N}_{w,d} = \binom{n}{w} \frac{(q^d-1)^w}{\prod_{i=0}^{d-1}(q^d - q^i)} \frac{\begin{bmatrix} k \\ d \end{bmatrix}_q}{\begin{bmatrix} n \\ d \end{bmatrix}_q}.$$

*Proof:* Let $\mathcal{U}_k$ be the set of all linear codes over $\mathbb{F}_q$ with length $n$ and dimension $k$. We observe that

$$N_{w,d} = \frac{\sum_{\mathscr{C} \in \mathcal{U}_k} |\mathcal{A}_{w,d}(\mathscr{C})|}{|\mathcal{U}_k|} = \frac{\sum_{\mathscr{C} \in \mathcal{U}_k} \sum_{\mathscr{B} \in \mathcal{A}_{w,d}(\mathbb{F}_q^n)} p(\mathscr{B}, \mathscr{C})}{\begin{bmatrix} n \\ k \end{bmatrix}_q},$$

where $p(\mathscr{B}, \mathscr{C}) = 1$ if $\mathscr{B} \subseteq \mathscr{C}$, and 0 otherwise. With a simple rewriting, we obtain

$$N_{w,d} = \frac{\sum_{\mathscr{B} \in \mathcal{A}_{w,d}(\mathbb{F}_q^n)} \sum_{\mathscr{C} \in \mathcal{U}_k} p(\mathscr{B}, \mathscr{C})}{\begin{bmatrix} n \\ k \end{bmatrix}_q} = \frac{\sum_{\mathscr{B} \in \mathcal{A}_{w,d}(\mathbb{F}_q^n)} \begin{bmatrix} n-d \\ k-d \end{bmatrix}_q}{\begin{bmatrix} n \\ k \end{bmatrix}_q},$$

where the r.h.s. term is justified by the observation that $\sum_{\mathscr{C} \in \mathcal{U}_k} p(\mathscr{B}, \mathscr{C})$ is equal to the number of $k$-dimensional codes having $\mathscr{B}$ as a subcode; this quantity is given by $\begin{bmatrix} n-d \\ k-d \end{bmatrix}_q$ (that is, the number of $(k-d)$-dimensional subspaces of $\mathbb{F}_q^n \setminus \mathscr{B}$, which has dimension $n-d$). With simple algebra, we find that $\begin{bmatrix} n-d \\ k-d \end{bmatrix}_q / \begin{bmatrix} n \\ k \end{bmatrix}_q = \begin{bmatrix} k \\ d \end{bmatrix}_q / \begin{bmatrix} n \\ d \end{bmatrix}_q$. So, we further obtain

$$N_{w,d} = |\mathcal{A}_{w,d}(\mathbb{F}_q^n)| \frac{\begin{bmatrix} k \\ d \end{bmatrix}_q}{\begin{bmatrix} n \\ d \end{bmatrix}_q}.$$

If the support of a subcode is $J$, then any of its generator matrices must be such that the columns indexed by $J$ are non-null. For a fixed $J$, the number of such matrices is $(q^d-1)^w$: to consider that any code has multiple generator matrices, we divide this quantity by the number of changes of basis, that is, $\prod_{i=0}^{d-1}(q^d - q^i)$. This way we obtain an upper bound on the size of $|\mathcal{A}_{w,d}(\mathbb{F}_q^n)|$: indeed, some of the matrices we are considering may have rank $< d$. Considering that we have $\binom{n}{w}$ choices for $J$, we obtain an upper bound since

$$|\mathcal{A}_{w,d}(\mathbb{F}_q^n)| \le \binom{n}{w} \frac{(q^d-1)^w}{\prod_{i=0}^{d-1}(q^d - q^i)}.$$

To prove the lower bound, we fix again a set $J$ and, among all the matrices with support $J$, consider only those for which the leftmost $d \times d$ submatrix is the identity matrix. This way, we avoid multiple counting of the same code: any two matrices $(\mathbf{I}_d, \mathbf{V})$ and $(\mathbf{I}_d, \mathbf{V}')$ (restricting to the columns indexed by $J$) such that $\mathbf{V} \neq \mathbf{V}'$ will generate different codes. Note that a matrix $(\mathbf{I}_d, \mathbf{V})$ can generate a code with support size $w$ if and only if $\mathbf{V} \in \mathbb{F}_q^{d \times (w-d)}$ has no null column: the number of such matrices is $(q^d-1)^{w-d}$. This way we obtain a lower bound, since there exist also codes that do not admit a generator matrix in the form $(\mathbf{I}_d, \mathbf{V})$. Considering again the number of choices for $J$, we set a lower bound as

$$|\mathcal{A}_{w,d}(\mathbb{F}_q^n)| \ge \binom{n}{w} (q^d-1)^{w-d}.$$

■

**Remark 3.** *When $d = 1$, a subcode corresponds to the orbit of a codeword under scalar multiplication by the elements of $\mathbb{F}_q$. The bounds in Theorem IV.1 coincide, so that*

$$N_{w,1} = \binom{n}{w}(q-1)^{w-1}\frac{q^k-1}{q^n-1} \approx \binom{n}{w}(q-1)^{w-1}q^{k-n}.$$

### *B. Using ISD to find subcodes with small support*

The result in Theorem IV.1 can be used to set values for $w$ and $d$ such that, given a random code $\mathscr{C}$, $\mathcal{A}_{w,d}(\mathscr{C})$ is non empty with high probability. As a rule of thumb, we consider that whenever $\widetilde{N}_{w,d} > 1$, the code contains at least one subcode with the desired properties. When $w$ is much smaller than $n$, then such subcodes can be efficiently found using ISD algorithms. If $d = 1$, finding subcodes with small support is equivalent to find codewords with small Hamming weight: we consider the algorithm in [20] and denote its time complexity as $T^{(1)}_{\mathcal{ISD}}(n,k,w)$. When $d > 1$, we can apply minor tweaks to ISD algorithms and use them to find subcodes. To the best of our knowledge, this idea has been considered only in [21], for the case of 2-dimensional codes and adapting Prange's simple ISD [22]. We consider a generalization of this method, where subcodes can have any dimension $d$; the corresponding procedure is detailed in Algorithm 1.

**Algorithm 1:** One iteration of ISD for $d > 1$

**Input:** generator matrix $\mathbf{G} \in \mathrm{GL}_{k,n}$ for $\mathscr{C}$, $w, d \in \mathbb{N}$
**Output:** failure, or generator matrix for $\mathscr{B} \subseteq \mathscr{C}$ with dimension $d$ and support size $w$

$\sigma \xleftarrow{\$} S_n$;
**if** $\mathsf{RREF}\big(\sigma(\mathbf{G}), \{1,\cdots,k\}\big)$ fails **then**
  Report failure;
**else**
  $(\mathbf{I}_d, \mathbf{V}) \leftarrow \mathsf{RREF}\big(\sigma(\mathbf{G}), \{1,\cdots,k\}\big)$
**for** $U \subseteq \{1,\cdots,k\}$ with size $d$ **do**
  $\mathbf{B} \leftarrow$ matrix formed by rows of $\mathbf{V}$ indexed by $U$;
  **if** $\mathbf{B}$ has support size $w-d$ **then**
    Return $\sigma^{-1}\big((\mathbf{I}_d, \mathbf{B})\big)$
Report failure;

For the algorithm to work, it must be $w \le n+d-k$. The probability that the computation of RREF does not fail can be estimated as $\frac{\prod_{i=0}^{d-1} q^d - q^i}{q^{d^2}}$ and, for large $q$, it can be assumed to be equal to 1. Let $\mathscr{B} \subseteq \mathcal{A}_{w,d}(\mathscr{C})$; then, the probability that one iteration finds $\mathscr{B}$ is given by $p(n,k,d,w) = \frac{\binom{w}{d}\binom{n-w}{k-d}}{\binom{n}{k}}$. When we have $|\mathcal{A}_{w,d}(\mathscr{C})|$ subcodes and we are simply interested in finding one of them, the success probability can be estimated as $1-(1-p(n,k,d,w))^{|\mathcal{A}_{w,d}(\mathscr{C})|}$. By using the lower bound in Theorem IV.1, we conservatively set this probability as $1-(1-p(n,k,d,w))^{\widetilde{N}_{w,d}}$. Computing RREF comes with a broad cost of $O(k^3)$, while the number of sets $U$ that are tested is $\binom{k}{d}$. Consequently, we assess the cost of finding a subcode of $\mathcal{A}_{w,d}(\mathscr{C})$ as

$$T^{(d)}_{\mathcal{ISD}}(n,k,w) = O\left(\frac{k^3+\binom{k}{d}}{1-(1-p(n,k,d,w))^{\widetilde{N}_{w,d}}}\right). \quad (6)$$

## V. New PKP solver

In this section we describe and analyze the algorithm we propose to solve the PKP. The method we propose is described in Algorithm 2 and represented in Figure 1.

**Algorithm 2:** New algorithm to solve PKP

**Data:** $w, w_1, w_2, d, \ell \in \mathbb{N}$, such that $w \le n$, $w = w_1 + w_2$, $d \le r$, $\ell \le n-r$.
**Input:** $\mathbf{H} \in \mathrm{GL}_{r,n}$, $\mathbf{s} \in \mathbb{F}_q^r$, $\mathbf{c} \in \mathbb{F}_q^n$
**Output:** $\widetilde{\mathbf{c}} \in \mathscr{S}_n(\mathbf{c})$ such that $\widetilde{\mathbf{c}}\mathbf{H}^\top = \mathbf{s}$

Use ISD to find $\widehat{\mathbf{H}}$, generator matrix of $\mathscr{B} \subseteq \mathscr{C}^\perp$, with dimension $d$ and support size $w$;
Compute $\mathbf{S} \in \mathrm{GL}_{d,r}$ such that $\widehat{\mathbf{H}} = \mathbf{S}\mathbf{H}$, $\sigma \in S_n$ such that $\mathrm{Supp}\big(\sigma(\widehat{\mathbf{H}})\big) = \{n-r+\ell-w+1,\cdots,n-r+\ell\}$;
$\hat{\mathbf{s}} \leftarrow \mathbf{s}\mathbf{S}^\top$, $\mathbf{Z} \leftarrow \sigma(\widehat{\mathbf{H}})$;
Set $K_1 = \{n-r+\ell-w+1,\cdots,n-r+\ell-w_2\}$, $K_2 = \{n-r+\ell-w_2,\cdots,n-r+\ell\}$;
Prepare $\mathcal{K}_1 = \Big\{\big(\mathbf{y}_1, \mathbf{y}_1\mathbf{Z}_{K_1}^\top\big) \,\Big|\, \mathbf{y}_1 \in \mathscr{S}_{w_1}(\mathbf{c})\Big\}$, $\mathcal{K}_2 = \Big\{\big(\mathbf{y}_2, \hat{\mathbf{s}} - \mathbf{y}_2\mathbf{Z}_{K_2}^\top\big) \,\Big|\, \mathbf{y}_2 \in \mathscr{S}_{w_2}(\mathbf{c})\Big\}$;
$\mathcal{K} \leftarrow \mathcal{K}_1 \bowtie \mathcal{K}_2$;
Compute $\mathbf{M} \in \mathrm{GL}_{r,r}$ such that $\mathbf{M}\sigma(\mathbf{H}) = (\mathbf{U}, \mathbf{I}_r)$;
$\hat{\mathbf{s}} \leftarrow \mathbf{s}\mathbf{M}^\top$;
$\widetilde{\mathbf{H}} \leftarrow$ matrix formed by rows and columns of $(\mathbf{U}, \mathbf{I}_r)$ at positions $\{d+1,\cdots,\ell\}$ and $\{1,\cdots,n-r+\ell\}$;
Set $L_1 = \{1,\cdots,n-r+\ell-w\}$ and $L_2 = \{n-r+\ell-w+1,\cdots,n-r+\ell\}$;
Prepare $\mathcal{L}_1 = \Big\{\big(\mathbf{x}_1, \mathbf{x}_1\widetilde{\mathbf{H}}_{L_1}^\top\big)\Big| \,\mathbf{x}_1 \in \mathscr{S}_{n-r+\ell-w}(\mathbf{c})\Big\}$, $\mathcal{L}_2 = \Big\{\big(\mathbf{x}_2, \hat{\mathbf{s}} - \mathbf{x}_2\widetilde{\mathbf{H}}_{L_2}^\top\big)\Big| \,\mathbf{x}_2 \in \mathcal{K}\Big\}$;
$\mathcal{L} \leftarrow \mathcal{L}_1 \bowtie \mathcal{L}_2$;
**for** $\mathbf{x} \in \mathcal{L}$ **do**
  Plug $\mathbf{x}$ into (4) to get $\widetilde{\mathbf{c}}$;
  **if** $\widetilde{\mathbf{c}} \in \mathscr{S}_n(\mathbf{c})$ **then**
    Return $\sigma^{-1}(\widetilde{\mathbf{c}})$;

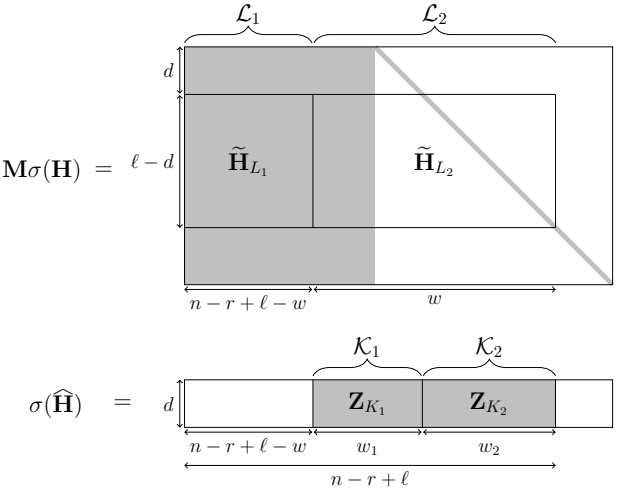


Fig. 1. Representation of the operations of Algorithm 2.

The correctness of the algorithm can be easily proven by considering that it essentially corresponds to the Algorithm

1 of [19], plus an additional filtering stage in which we cut some of the candidates for $\mathcal{L}_2$. To do this, we first find a $d$-dimensional subcode of $\mathscr{C}^{\perp}$, generated by $\widehat{\mathbf{H}}$, with support size $w$. We then find $\mathbf{S} \in \mathrm{GL}_{d,r}$ such that $\widehat{\mathbf{H}} = \mathbf{S}\mathbf{H}$ and compute $\hat{\mathbf{s}} = \mathbf{s}\mathbf{S}^{\top}$. Recalling (3), we use $\widehat{\mathbf{H}}$ to produce candidates for the entries of $\widetilde{\mathbf{c}}$ in the positions indexed by the support of $\sigma(\widehat{\mathbf{H}})$, that is, $\{n-r+\ell-w+1, \cdots, n-r+\ell\}$. To do this, we use a meet-in-the-middle approach (lines 4–6). We then apply another transformation (lines 7–9) to both $\mathbf{H}$ and $\mathbf{s}$, employing the systematic form of $\mathbf{H}$ to obtain $\ell - d$ new kernel equations involving exactly $n - r + \ell$ entries. We then have another round of lists merging, corresponding to the same procedure employed in Algorithm 1 of [19], with the only difference that to build $\mathcal{L}_2$ we use the elements of $\mathcal{K}$, instead of those in $\mathscr{S}_w(\mathbf{c})$. This difference is crucial since the gain of our method lies in this step: we expect $|\mathcal{K}| < |\mathscr{S}_w(\mathbf{c})|$, which yields a final list $\mathcal{L}$ with less elements.

In the following Proposition we derive the time complexity of the proposed algorithm.

**Proposition V.1.** *Let $d, w_1, w_2$ such that $\breve{N}_{w_1+w_2,d} > 1$. Then, Algorithm 2 runs in time*

$$T^{(d)}_{\mathcal{ISD}}(n, r, w_1, w_2) + T_{\mathcal{K}} + T_{\mathcal{L}} + \frac{n! q^{-\ell}}{(n-r+\ell)!},$$

*with $w = w_1 + w_2$ and*

$$T_{\mathcal{K}} = \frac{n!}{(n-w_1)!} + \frac{n!}{(n-w_2)!} + \frac{(n!)^2 q^{-d}}{(n-w_1)!(n-w_2)!},$$

$$T_{\mathcal{L}} = \frac{n!}{(n-r+\ell-w)!} + \frac{n! q^{-d}}{(n-w)!} + \frac{(n!)^2 q^{-\ell}}{(n-w)!(n-r+\ell-w)!}.$$

*Proof:* Since $\breve{N}_{w,d} > 1$, we expect $\mathscr{C}^{\perp}$ to contain at least a subcode with dimension $d$ and support size $w$. To find such a subcode, we have a cost given by $T^{(d)}_{\mathcal{ISD}}(n, r, w)$. Steps 2–4 come with a negligible cost, so we omit them. The cost of building and merging the lists (using a smart binary search algorithm to determine the collisions) is given by $\mathcal{K}_1$ and $\mathcal{K}_2$ and results in $\frac{n!}{(n-w_1)!} + \frac{n!}{(n-w_2)!}$ operations. The number of collisions, on average, is given by $|\mathcal{K}_1| \cdot |\mathcal{K}_2| \cdot q^{-d}$, so that the cost to produce $\mathcal{K}$ is

$$\frac{n!}{(n-w_1)!} + \frac{n!}{(n-w_2)!} + \frac{(n!)^2 q^{-d}}{(n-w_1)!(n-w_2)!}.$$

After the collisions are checked, $\mathcal{K}$ contains $\frac{n!}{(n-w)!} q^{-d}$ elements, on average. The cost of steps 7–9 can be neglected while, to execute steps 11-12, we repeat the previous reasoning and hence their cost is given by $|\mathcal{L}_1| + |\mathcal{L}_2| + |\mathcal{L}_1| \cdot |\mathcal{L}_2| \cdot q^{-(\ell-d)}$, which, on average, is equal to

$$\frac{n!}{(n-r+\ell-w)!} + \frac{n! q^{-d}}{(n-w)!} + \frac{(n!)^2 q^{-\ell}}{(n-w)!(n-r+\ell-w)!}.$$

Notice that, in the above formula, we have considered that $|\mathcal{L}_2| = |\mathcal{K}|$. Finally, we also take into account the cost of iterating steps 13-16, whose number can be considered equal to the size of $\mathcal{L}$, and so, on average, is equal to $\frac{n! q^{-\ell}}{(n-r+\ell)!}$. ■

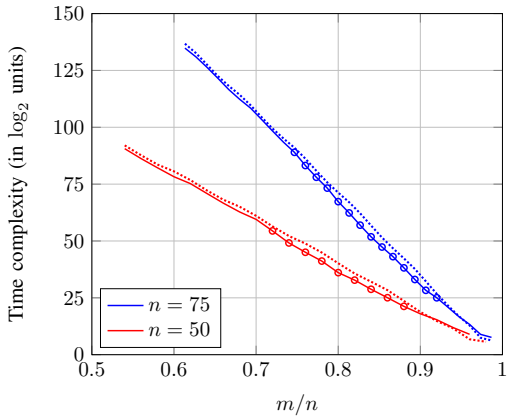


Fig. 2. Comparison of the time complexity of [19, Algorithm 1] (dotted lines) with that of our algorithm (full lines), for $q = 251$. The circles highlight the cases in which our algorithm is optimized with $d \geq 1$.

TABLE I
TIME COMPLEXITY OF OUR ATTACK FOR THE PKP-DSS PARAMETERS RECOMMENDED IN [6], [19].

| $(n, m, q)$ | Claimed cost | $(d, w, w_1, w_2, \ell)$ | Cost of Algorithm 2 |
|---|---|---|---|
| $(69, 41, 251)$ | $2^{130}$ | $(1, 22, 2, 20, 16)$ | $2^{125.47}$ |
| $(94, 54, 509)$ | $2^{193}$ | $(1, 31, 2, 29, 22)$ | $2^{189.77}$ |

In Figure 2 we compare the performance of Algorithm 2 with that of [19, Algorithm 1], for the case of $q = 251$ and several pairs of values $(m, n)$, chosen such that $n! q^{-m} < 1$. As we can see, unless $m$ is close to $n$, our algorithm is faster than the one in [19]; in particular, the speed-up increases when our algorithm is optimized with $d \geq 1$.

To assess the impact of our algorithm on the cryptanalysis of schemes relying on the PKP, in Table I we consider the PKP-DSS instances which have been recommended in [6] for the security levels of 128 and 192 bits. For these instances, the claimed cost of [19, Algorithm 1] is $2^{130}$ and $2^{193}$, respectively. As we can see, our attack is faster and, furthermore, has a cost which is slightly lower than the claimed security levels. For the 256-bit security instance, we found instead that our attack does not improve upon [19].

## VI. CONCLUSION

We have described a novel attack to the PKP which makes use of small support subspaces of kernel equations. Our proposed algorithm is based on techniques borrowed from the code-based cryptography context and is faster than state-of-the-art attacks for several cases. To consider a situation of practical interest, we have shown that the security of some PKP-DSS instances is slightly overestimated. Despite the moderate gain in complexity with respect to the state-of-the-art, our work shows that the PKP can be solved exploiting coding theory techniques and this may lead to new, possibly even more efficient, attack avenues in the future.

## REFERENCES


[1] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology — CRYPTO' 86 Proceedings. CRYPTO 1986*, ser. Lecture Notes in Computer Science, A. M. Odlyzko, Ed., vol. 263. Springer, Berlin, Heidelberg, 1986, pp. 186–194.

[2] S. Gueron, E. Persichetti, and P. Santini, "Designing a practical code-based signature scheme from zero-knowledge proofs with trusted setup," *Cryptography*, vol. 6, no. 1:5, 2022.

[3] S. Bettaieb, L. Bidoux, O. Blazy, and P. Gaborit, "Zero-knowledge reparation of the Véron and AGS code-based identification schemes," in *Proc. 2021 IEEE International Symposium on Information Theory (ISIT 2021)*, Melbourne, Victoria, Australia, Jul. 2021, pp. 55–60.

[4] L. Bidoux, P. Gaborit, M. Kulkarni, and V. Mateu, "Code-based signatures from new proofs of knowledge for the syndrome decoding problem," *arXiv preprint arXiv:2201.05403*, 2022.

[5] A. Becker, A. Joux, A. May, and A. Meurer, "Sigma protocols for MQ, PKP and SIS, and fishy signature schemes," in *Advances in Cryptology – EUROCRYPT 2020*, ser. Lecture Notes in Computer Science, A. Canteaut and Y. Ishai, Eds., vol. 12107. Springer, Cham, 2020, pp. 183–211.

[6] W. Beullens, J.-C. Faugère, E. Koussa, G. Macario-Rat, J. Patarin, and L. Perret, "PKP-based signature scheme," in *Progress in Cryptology – INDOCRYPT 2019*, ser. Lecture Notes in Computer Science, S. R. F. Hao and S. S. Gupta, Eds., vol. 11898. Springer, Cham, 2019, pp. 3–22.

[7] A. Barenghi, J.-F. Biasse, E. Persichetti, and P. Santini, "LESS-FM: fine-tuning signatures from the code equivalence problem," in *Post-Quantum Cryptography - 12th International Workshop, PQCrypto 2021*, ser. Lecture Notes in Computer Science, J. H. Cheon and J.-P. Tillich, Eds., vol. 12841. Springer, 2021, pp. 23–43.

[8] T. Feneuil, A. Joux, and M. Rivain, "Shared permutation for syndrome decoding: New zero-knowledge protocol and code-based signature," Cryptology ePrint Archive, Report 2021/1576, 2021, https://ia.cr/2021/1576.

[9] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Zero-knowledge from secure multiparty computation," in *Proc. Thirty-Ninth Annual ACM Symposium on Theory of Computing - STOC '07*, San Diego, CA, Jun. 2007, pp. 21–30.

[10] G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, J. Kelsey, Y.-K. Liu, C. Miller, D. Moody, R. Peralta *et al.*, "Status report on the second round of the NIST post-quantum cryptography standardization process," *US Department of Commerce, NIST*, 2020.

[11] D. Moody, "Status update on the 3rd round," NIST, Tech. Rep., Jun. 2021. [Online]. Available: https://csrc.nist.gov/presentations/2021/status-update-on-the-3rd-round

[12] A. Shamir, "An efficient identification scheme based on permuted kernels," in *Advances in Cryptology — CRYPTO' 89 Proceedings. CRYPTO 1989*, ser. Lecture Notes in Computer Science, G. Brassard, Ed., vol. 435. Springer, 1989, pp. 606–609.

[13] J. Georgiades, "Some remarks on the security of the identification scheme based on permuted kernels," *Journal of Cryptology*, vol. 5, no. 2, pp. 133–137, 1992.

[14] T. Baritaud, M. Campana, P. Chauvaud, and H. Gilbert, "On the security of the permuted kernel identification scheme," in *Advances in Cryptology — CRYPTO' 92*, ser. Lecture Notes in Computer Science, E. F. Brickell, Ed., vol. 740. Springer, 1992, pp. 305–311.

[15] J. Patarin and P. Chauvaud, "Improved algorithms for the permuted kernel problem," in *Cryptology — CRYPTO' 93 Proceedings. CRYPTO 1993*, ser. Lecture Notes in Computer Science, D. R. Stinson, Ed., vol. 773. Springer, Berlin, Heidelberg, 1993, pp. 391–402.

[16] G. Poupard, "A realistic security analysis of identification schemes based on combinatorial problems," *European Transactions on Telecommunications*, vol. 8, no. 5, pp. 471–480, 1997.

[17] É. Jaulmes and A. Joux, "Cryptanalysis of PKP: a new approach," in *Public Key Cryptography. PKC 2001*, ser. Lecture Notes in Computer Science, K. K., Ed., vol. 1992. Springer, Berlin, Heidelberg, 2001, pp. 165–172.

[18] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Francisco, 1979.

[19] E. Koussa, G. Macario-Rat, and J. Patarin, "On the complexity of the Permuted Kernel Problem," Cryptology ePrint Archive, Report 2019/412, 2019, https://ia.cr/2019/412.

[20] C. Peters, "Information-set decoding for linear codes over $\mathbb{F}_q$," in *Post-Quantum Cryptography 2010*, ser. Lecture Notes in Computer Science, N. Sendrier, Ed., vol. 6061. Springer, Berlin, Heidelberg, 2010, pp. 81–94.

[21] W. Beullens, "Not Enough LESS: An Improved Algorithm for Solving Code Equivalence Problems over $\mathbb{F}_q$," in *International Conference on Selected Areas in Cryptography*. Springer, 2020, pp. 387–403.

[22] E. Prange, "The use of information sets in decoding cyclic codes," *IRE Trans. Inf. Theory*, vol. 8, no. 5, pp. 5–9, 1962.