A Polynomial-Time Attack on the BBCRS Scheme

Alain Couvreur[1,2]*, Ayoub Otmani[3]**, Jean-Pierre Tillich[1] ***, and Valérie Gauthier–Umaña[4] †

[1] INRIA
[2] LIX, École Polytechnique
[3] University of Rouen
[4] Universidad del Rosario

Abstract. The BBCRS scheme is a variant of the McEliece public-key encryption scheme where the hiding phase is performed by taking the inverse of a matrix which is of the form $\boldsymbol{T} + \boldsymbol{R}$ where $\boldsymbol{T}$ is a sparse matrix with average row/column weight equal to a very small quantity m, usually $m < 2$, and $\boldsymbol{R}$ is a matrix of small rank $z \geqslant 1$. The rationale of this new transformation is the reintroduction of families of codes, like generalized Reed-Solomon codes, that are famously known for representing insecure choices. We present a key-recovery attack when $z = 1$ and m is chosen between 1 and $1 + R + O(\frac{1}{\sqrt{n}})$ where R denotes the code rate. This attack has complexity $O(n^6)$ and breaks all the parameters suggested in the literature.

Keywords. Code-based cryptography; distinguisher; generalized Reed-Solomon codes; key-recovery; component-wise product of codes.

Introduction

Post-quantum cryptography. All public key cryptographic primitives used in practice such as RSA, ElGamal scheme, DSA or ECDSA rely either on the difficulty of factoring or computing the discrete logarithm and would therefore be broken by Shor's algorithm [24] if a large enough quantum computer could be built. Moreover, even if a large enough quantum

* Grace Team & LIX, CNRS UMR 7161, Allée H. D'Estienne d'Orves, 91120 Palaiseau Cedex, France. alain.couvreur@lix.polytechnique.fr

** University of Rouen, LITIS, F-76821 Mont-Saint-Aignan, France. ayoub.otmani@univ-rouen.fr

*** Secret Team, 78153 Le Chesnay Cedex, France. jean-pierre.tillich@inria.fr

† Faculty of Natural Sciences and Mathematics, Department of Mathematics, Universidad del Rosario, Bogotá, Colombia. gauthier.valerie@urosario.edu.co

computer might not be built in the next five years, it should be mentioned that tremendous progress has been made for computing the discrete logarithm over finite fields of small characteristic with the quasi-polynomial time algorithm of [5]. This lack of diversity in public key cryptography has been identified as a major concern in the field of information security. For all these reasons, it would be very desirable to be ready to replace these schemes by others that would rely on other hard problems. However only few other proposals have emerged which are essentially hash-based signature schemes, lattice-based, code-based and multivariate quadratic based schemes. They are either based on the problem of solving multivariate equations over a finite field, the problem of finding a short vector in a lattice and the problem of decoding a linear code. Those problems are known for being NP-hard and are therefore believed to be immune to the quantum computer threat.

The McEliece cryptosystem. Among those, one of the most promising scheme is the McEliece public key cryptosystem [20]. It is also one of the oldest public-key cryptosystem. It uses a family of codes for which there is a fast decoding algorithm (the binary Goppa code family here) which is used in the decryption process whereas an attacker has only a random generator matrix of the Goppa code which reveals nothing about the algebraic structure of the Goppa code that is used in the decoding process. He has therefore to decode a generic linear code for which only exponential time decoding algorithms are known. The main advantage of this system is to have very fast encryption and decryption functions. Depending on how the parameters are chosen for a fixed security level, this cryptosystem is about five times faster for encryption and about 10 to 100 times faster for decryption than RSA [8]. Furthermore, it has withstood many attacking attempts. After more than thirty five years now, it still belongs to the very few public key cryptosystems which remain unbroken.

The use of Reed-Solomon codes in a McEliece scheme. Goppa codes are subfield subcodes of Generalized Reed-Solomon codes (GRS codes in short). This means that a Goppa code defined over $\mathbb{F}_q$ is actually the set of codewords of a GRS code defined over an extension field $\mathbb{F}_{q^\mu}$ (we say that μ is the extension degree of the Goppa code) whose coordinates all belong to the subfield $\mathbb{F}_q$. Actually the fast decoding process of Goppa codes is the decoder of the underlying GRS code. Roughly speaking, a Goppa code of length n and dimension $n-2t\mu$ defined over $\mathbb{F}_q$ can correct t errors[5] and is a subfield subcode of a GRS code that can also correct t

[5] but the dimension can be increased to $n - t\mu$ in the binary case

errors which is of the same length n but has a larger dimension $n-2t$ and is defined over $\mathbb{F}_{q^\mu}$. In this sense, the underlying GRS code has a better error correction capacity than the Goppa code. This raises the issue of using GRS codes instead of Goppa codes in the McEliece system. The better decoding capacity of GRS codes translates into smaller public key sizes for the McEliece scheme which is actually one of the main drawback of this scheme. This approach has been tried in Niederreiter's scheme (whose security is equivalent to the McEliece scheme) but has encountered a dreadful fate when the Sidelnikov-Shestakov attack appeared [25].

Baldi et al. approach for reviving GRS codes. In their Journal of Cryptology article [2], Baldi *et al.* have suggested a new way of using GRS codes in this context. Instead of using directly such a code, they multiplied it by the inverse of the sum $\boldsymbol{T}+\boldsymbol{R}$ where $\boldsymbol{T}$ is a sparse matrix and $\boldsymbol{R}$ is a low rank matrix. By doing this, the attacker sees a code which is radically different from a GRS code but the legitimate user can still use the underlying GRS decoder. This thwarts the Sidelnikov-Shestakov attack completely. However the decoding capacity of the resulting code is basically scaled down by a factor of $\frac{1}{m}$ where m denotes the average weight of rows of the matrix $\boldsymbol{T}$. It should be noted that the very same approach has also been tried for the Low-Density-Parity-Check code family, LDPC in short, which is notoriously known for being insecure in a McEliece scheme [22, 4, 3]. In this case, they did not even use the low rank matrix and despite of this fact the resulting public code obtained by this multiplication is not an LDPC code anymore (it becomes a moderate-density-parity-check code) and it seems now that if the attacker wants to break this scheme he has to be able to solve a generic decoding problem [21]. There are therefore good reasons to believe that this approach can be powerful for disguising the secret code structure.

An earlier attempt. Baldi *et al.* [1] first used this approach with $\boldsymbol{T}$ being a permutation matrix. In this case $m=1$ and nothing is lost in term of decoding capacity compared to a GRS decoder. In other words, this allows to decrease the public key size as if we had a GRS code in the McEliece cryptosystem. This first attempt got broken in [12, 11]. Roughly speaking the reason of this attack in this case can be traced back to two facts (i) it turns out that the resulting code is still close to the underlying GRS code: the intersection of the public code with the secret GRS code is of co-dimension one; (ii) there is a very powerful way of distinguishing a GRS code [12] from a random code by computing the dimension of its square which can be used to unravel the algebraic structure of the public code. On the other hand, when the degree of sparseness of $\boldsymbol{T}$ is > 1 the resulting code does not have a large intersection with a GRS code and there was some hope to obtain a secure scheme.

Our contribution: an attack which works in the regime $1 < m < 2$. In the present article we will show that despite the fact that the public code is far from being a GRS code, a similar trick that has already been used to attack successfully in [14] some wild Goppa codes proposed in [7] when the degree of extension is only 2 can also be used in this context. It consists in computing the dimension of the square of shortenings of the public code. Because of the hidden structure of the public code, the squares of some of its shortenings have a smaller dimension than the squares of shortened random codes of the same dimension. This distinguisher is then used to unravel the structure of the matrix $\boldsymbol{T}$. This gives an attack of polynomial time complexity which can be used to break the examples given in [2]. Several were broken in a few hours, and others in a few days. As an illustration, Example 1 given in [2] with a claimed 90-bit security can be broken in 2.75 hours on a computer equipped with Xeon 2.27GHz processor and 72 Gb of RAM. This attack works up to values of m of order $1 + R + O(\frac{1}{\sqrt{n}})$, where R is the rate of the public code. The attack we present here can obviously be thwarted by taking values for m greater than 2, but in this case, since the price to pay is a decrease of the decoding capacity by a factor of more than 2, we do not obtain better public key sizes than the ones we obtain by using Goppa codes, or more generally alternant codes of extension degree 2, provided we choose non wild Goppa codes in order to avoid the attack of [14]. The complexity of the present attack is similar to that of [11], namely $O(n^6)$ where n is the code length. More precisely, this attack starts with two steps of respective complexity $O(n^3)$ and $O(n^5)$ and then applying the attack of [11] whose complexity is $O(n^6)$ operations in the base field.

Note. Due to space limitation, several proofs are omitted. A longer version of the present paper including the missing proofs can be found online.

1 GRS Codes and the Square Code Construction

We recall in this section a few relevant results and definitions from coding theory and bring in the fundamental notion of square code construction.

Definition 1 (Generalized Reed-Solomon code). *Let k and n be integers such that $1 \leqslant k < n \leqslant q$ where q is a prime power. The code*

$\mathbf{GRS}_k(\boldsymbol{x},\boldsymbol{y})$ of dimension k is associated to a pair $(\boldsymbol{x},\boldsymbol{y})$ where $\boldsymbol{x}$ is an n-tuple of distinct elements of $\mathbb{F}_q$ and $\boldsymbol{y} \in (\mathbb{F}_q^{\times})^n$, is defined as:

$$\mathbf{GRS}_k(\boldsymbol{x},\boldsymbol{y}) \stackrel{def}{=} \Big\{(y_1 p(x_1), \dots, y_n p(x_n)) \mid p \in \mathbb{F}_q[X], \deg p < k\Big\}.$$

The first work that suggested to use GRS codes in a public-key encryption scheme was [23]. But Sidelnikov and Shestakov [25] showed that for any GRS code it is possible to recover in polynomial time a pair $(\boldsymbol{x},\boldsymbol{y})$ defining it, which is all that is needed to decode efficiently such codes and is therefore enough to break any McEliece type cryptosystem [20] that uses GRS codes.

Definition 2 (Componentwise products). *Given two vectors $\boldsymbol{a} = (a_1,\dots,a_n)$ and $\boldsymbol{b} = (b_1,\dots,b_n) \in \mathbb{F}_q^n$, we denote by $\boldsymbol{a} \star \boldsymbol{b}$ the componentwise product*

$$\boldsymbol{a} \star \boldsymbol{b} \stackrel{def}{=} (a_1 b_1, \dots, a_n b_n).$$

The star product $\boldsymbol{a} \star \boldsymbol{b}$ should be distinguished from a more common operation, namely the canonical inner product:

$$\boldsymbol{a} \cdot \boldsymbol{b} \stackrel{\text{def}}{=} \sum_{i=1}^{n} a_i b_i.$$

Definition 3 (Product of codes & square code). *Let $\mathscr{A}$ and $\mathscr{B}$ be two codes of length n. The* star product code *denoted by $\mathscr{A} \star \mathscr{B}$ of $\mathscr{A}$ and $\mathscr{B}$ is* the vector space spanned by all products *$\boldsymbol{a} \star \boldsymbol{b}$ where $\boldsymbol{a}$ and $\boldsymbol{b}$ range over $\mathscr{A}$ and $\mathscr{B}$ respectively. When $\mathscr{B} = \mathscr{A}$ then $\mathscr{A} \star \mathscr{A}$ is called the* square code *of $\mathscr{A}$ and is rather denoted by $\mathscr{A}^2$.*

Proposition 1. *Let $\mathscr{A}$ be a code of length n, then*

$$\dim(\mathscr{A}^2) \leqslant \min\left\{n, \binom{\dim(\mathscr{A})+1}{2}\right\}.$$

Proposition 2. *Let $\mathscr{A} \subset \mathbb{F}_q^n$ be a code of dimension k. The complexity of the computation of a basis of $\mathscr{A}^2$ is $O(k^2 n^2)$ operations in $\mathbb{F}_q$.*

See for instance [11], for proofs of Propositions 1 and 2.

The importance of the square code construction becomes clear when we compare the dimension of the square of structured codes like GRS codes with the dimension of the square of a random code. Roughly speaking, given a code of dimension k, the dimension of its square is linear in k if it is a GRS code and quadratic if it is a random code as explained in the two following propositions.

Proposition 3. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})^2 = \mathbf{GRS}_{2k-1}(\boldsymbol{x}, \boldsymbol{y} \star \boldsymbol{y})$.

Proof. See for instance [18, Proposition 10].

Remark 1. This property can also be used in the case $2k - 1 > n$. To see this, consider the dual of the Reed-Solomon code, which is itself a generalized Reed-Solomon code [17, Theorem 4, p.304].

Theorem 1. *Let $\mathscr{A}$ be a random code of length n and dimension k such that $n > \binom{k+1}{2}$. Then, for all integer $\ell < \binom{k+1}{2}$,*

$$\mathsf{Prob}\left(\dim \mathscr{A}^2 \leqslant \binom{k+1}{2} - \ell\right) = O\left(q^{-\ell} \cdot q^{-\left(n - \binom{k+1}{2}\right)}\right), \qquad (k \to +\infty).$$

Proof. See [10].

Remark 2. A slightly weaker result was already obtained in the papers [15, 16] (see also [19]).

For this reason, $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})$ can be distinguished from a random linear code of the same dimension by computing the dimension of the associated square code. In [15, 16], this phenomenon was already observed for q-ary alternant codes (in particular Goppa codes) at very high rates whose duals are distinguishable from random codes by the very same manner. Subsequently, the very same phenomenon lead to attacks on GRS based cryptosystems [12, 11], to a polynomial time attack on Wild Goppa codes over quadratic extensions [14] and to a polynomial time attack on algebraic geometry codes [13].

Historically, the star product of codes has been used for the first time by Wieschebrink to cryptanalyze a McEliece-like scheme [6] based on subcodes of Reed-Solomon codes [26]. The use of the star product here is nevertheless different from the way it is used in [26]. In Wieschebrink's paper, the star product is used to identify, given a certain low codimensional subcode $\mathscr{C}$ of a GRS code $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})$, a possible pair $(\boldsymbol{x}, \boldsymbol{y})$.

This is achieved by computing $\mathscr{C}^2$ which turns out to be $\mathbf{GRS}_k(\boldsymbol{x},\boldsymbol{y})^2 = \mathbf{GRS}_{2k-1}(\boldsymbol{x},\boldsymbol{y}\star\boldsymbol{y})$ with a high probability. The Sidelnikov and Shestakov algorithm is then used on $\mathscr{C}^2$ to recover a possible $(\boldsymbol{x},\boldsymbol{y}\star\boldsymbol{y})$ pair to describe $\mathscr{C}^2$ as a GRS code, and hence, a pair $(\boldsymbol{x},\boldsymbol{y})$ is deduced for which $\mathscr{C} \subset \mathbf{GRS}_k(\boldsymbol{x},\boldsymbol{y})$.

2 Description of the Scheme

The BBCRS public-key encryption scheme given in [2] can be summarized as follows:

Secret key.
- $\boldsymbol{G_{sec}}$ is a generator matrix of a GRS code of length n and dimension k over $\mathbb{F}_q$.
- $\boldsymbol{Q} \stackrel{\text{def}}{=} \boldsymbol{T}+\boldsymbol{R}$ where $\boldsymbol{T}$ is an $n\times n$ non-singular sparse matrix with elements in $\mathbb{F}_q$ and average row weight $m \ll n$. Note that m is not necessarily an integer. For example $m=1.4$ means that 40% of the rows of $\boldsymbol{T}$ have weight equal to 2 and the other 60% have weight equal to 1.
- $\boldsymbol{R}$ is a rank-z matrix over $\mathbb{F}_q$ such that $\boldsymbol{Q}$ is invertible. In other words there exist $\boldsymbol{\alpha} \stackrel{\text{def}}{=} (\alpha_1,\dots,\alpha_n)$ and $\boldsymbol{\beta} \stackrel{\text{def}}{=} (\beta_1,\dots,\beta_n)$ such that $\boldsymbol{R} \stackrel{\text{def}}{=} \boldsymbol{\alpha}^T\boldsymbol{\beta}$ and α_i and β_i are $z\times 1$ full rank matrices defined over $\mathbb{F}_q$ for all $i\in\{1,\dots,n\}$ and $z \leqslant n$.
- $\boldsymbol{S}$ is a $k\times k$ random invertible matrix over $\mathbb{F}_q$.

Public key.

$$\boldsymbol{G_{pub}} \stackrel{\text{def}}{=} \boldsymbol{S}^{-1}\boldsymbol{G_{sec}}\boldsymbol{Q}^{-1}. \tag{1}$$

Encryption. The ciphertext $\boldsymbol{c}\in\mathbb{F}_q^n$ of a plaintext $\boldsymbol{m}\in\mathbb{F}_q^k$ is obtained by drawing at random $\boldsymbol{e}$ in $\mathbb{F}_q^n$ of weight less than or equal to $\frac{n-k}{2m}$ (recall that m denotes the density of the matrix $\boldsymbol{T}$) and computing $\boldsymbol{c} \stackrel{\text{def}}{=} \boldsymbol{m}\boldsymbol{G_{pub}}+\boldsymbol{e}$.

Decryption. It consists in performing the three following steps:
1. Guessing the value of $\boldsymbol{eR}$.
2. Calculating $\boldsymbol{c}' \stackrel{\text{def}}{=} \boldsymbol{cQ}-\boldsymbol{eR} = \boldsymbol{mS}^{-1}\boldsymbol{G_{sec}}+\boldsymbol{eQ}-\boldsymbol{eR} = \boldsymbol{mS}^{-1}\boldsymbol{G_{sec}}+\boldsymbol{eT}$ and using the decoding algorithm of the GRS code to recover $\boldsymbol{mS}^{-1}$ from the knowledge of $\boldsymbol{c}'$.
3. Multiplying the result of the decoding by $\boldsymbol{S}$ to recover $\boldsymbol{m}$.

Remark 3. In [2], the authors suggest to take $m = 1+\frac{n-k-3}{n} \approx 2-R$ for the density of $\boldsymbol{T}$.

Further details on the construction of the matrix $\boldsymbol{T}$. We deal with the case $m \leqslant 2$. According to [2] the matrix $\boldsymbol{T}$ is constructed[6] as follows.

1. Choose a permutation matrix $\boldsymbol{P}$. Replace each 1 by a random element of $\mathbb{F}_q^{\times}$.
2. Set $t \stackrel{\text{def}}{=} \lfloor \frac{n-k}{2} \rfloor$, $\delta_t \stackrel{\text{def}}{=} t - \lfloor \frac{t}{m} \rfloor$ and $\ell \stackrel{\text{def}}{=} \lfloor (m-1)n \rfloor$. Choose a random set $\mathcal{C}$ of δ_t columns and a random set $\mathcal{J}_2$ of ℓ rows of $\boldsymbol{P}$.
3. For all $i \in \mathcal{J}_2$, we denote by $\pi(i)$ the integer such that $\boldsymbol{P}_{i,\pi(i)} \neq 0$. For each $i \in \mathcal{J}_2$, choose a random element $j \in \mathcal{C} \setminus \pi(i)$ and add a random element of $\mathbb{F}_q^{\times}$ at position (i, j).

We also tested another construction allowing to have row and column weight upper bounded by 2. The sparse matrix $\boldsymbol{T}$ is constructed as $\boldsymbol{T} = \boldsymbol{T}_1 + \boldsymbol{T}_2$ where:

– $\boldsymbol{T}_1$ is of the form $\boldsymbol{T}_1 = \boldsymbol{D}_1 \boldsymbol{P}_1$, where $\boldsymbol{D}_1$ is diagonal invertible and $\boldsymbol{P}_1$ is a permutation matrix;
– $\boldsymbol{T}_2 = \boldsymbol{D}_2 \boldsymbol{P}_2$, where $\boldsymbol{D}_2$ is diagonal with $(m-1)n$ nonzero diagonal coefficients and $\boldsymbol{P}_2$ is a permutation matrix;
– The matrices do not overlap, that is, there is no pair (i, j) with $1 \leqslant i, j \leqslant n$ such that both $(\boldsymbol{T}_1)_{ij}$ and $(\boldsymbol{T}_2)_{ij}$ are nonzero.

Our attack works for both choices of the matrix $\boldsymbol{T}$. The experimental results in Sec. 6 rely on the first construction for $\boldsymbol{T}$.

2.1 Previous attacks and discussion on the parameters

The BBCRS scheme has been subject to an attack [11] in the case $m = 1$, *i.e.* the matrix $\boldsymbol{T}$ is a permutation matrix and $z = 1$, *i.e.* the matrix $\boldsymbol{R}$ has rank 1. The attack presented here holds for $m < 1 + R + O(\frac{1}{\sqrt{n}})$ and $z = 1$. The relevance of choosing higher m or z is discussed in Section 7.

The attack of the present article uses in its last step the attack [11] on the original system [1].

2.2 Notation

It will be convenient to bring the following notation.

[6] Actually, the authors propose three constructions for $\boldsymbol{T}$ and express a clear preference for the one described in the present article.

- $\mathscr{C}_{\text{pub}}$ is the code with generator matrix $\boldsymbol{G_{pub}}$;
- $\mathscr{C}_{\text{sec}}$ is the GRS code with generator matrix $\boldsymbol{G_{sec}}$, we assume that it is specified by its dual (which is itself a GRS code) as $\mathscr{C}_{\text{sec}}^{\perp} = \mathbf{GRS}_{n-k}(\boldsymbol{x}, \boldsymbol{y})$;
- $\mathcal{J}_1$ is the set of positions which correspond to rows of $\boldsymbol{T}$ of Hamming weight 1. The elements of $\mathcal{J}_1$ are called the *positions of degree* 1. For any row $i \in \mathcal{J}_1$ of $\boldsymbol{T}$, we define $j(i)$ as the unique column of $\boldsymbol{T}$ for which $T_{ij(i)} \neq 0$;
- $\mathcal{J}_2$ is the set of positions which correspond to rows of $\boldsymbol{T}$ of Hamming weight 2. The positions in $\mathcal{J}_2$ are called the *positions of degree* 2. When i belongs to $\mathcal{J}_2$, let j_1 and j_2 be the columns of $\boldsymbol{T}$ for which we have $T_{ij_1} \neq 0$ and $T_{ij_2} \neq 0$. We define similarly $j(i)$ as the set $\{j_1, j_2\}$ in this case.

2.3 Structure of the public code

The following result explains how $\mathscr{C}_{\text{pub}}$ and $\mathscr{C}_{\text{sec}}$ and their duals are related.

Lemma 1.

$$\mathscr{C}_{pub} = \mathscr{C}_{sec}(\boldsymbol{T} + \boldsymbol{R})^{-1} \tag{2}$$
$$\mathscr{C}_{pub}^{\perp} = \mathscr{C}_{sec}^{\perp}(\boldsymbol{T} + \boldsymbol{R})^{T}. \tag{3}$$

Proof. The first equality follows immediately from (1), whereas the second one was is observed in [2, p.6, Equation (8)] where a parity-check matrix for the public code $\mathscr{C}_{\text{pub}}$ is expressed in terms of a parity-check matrix of the secret code. This can be proved as follows. For all $c \in \mathscr{C}_{\text{sec}}$, $c' \in \mathscr{C}_{\text{sec}}^{\perp}$,

$$(c(\boldsymbol{T} + \boldsymbol{R})^{-1}) \cdot (c'(\boldsymbol{T} + \boldsymbol{R})^{T}) = (c(\boldsymbol{T} + \boldsymbol{R})^{-1}(\boldsymbol{T} + \boldsymbol{R})) \cdot c' = c \cdot c' = 0.$$

Moreover, since $\boldsymbol{Q} = \boldsymbol{T} + \boldsymbol{R}$ is invertible, we get $\dim \mathscr{C}_{\text{sec}}^{\perp}(\boldsymbol{T} + \boldsymbol{R})^{T} + \dim \mathscr{C}_{\text{sec}}(\boldsymbol{T} + \boldsymbol{R})^{-1} = n$, hence the codes are dual to each other.

3 The fundamental tool: shortening and puncturing the dual of the public code

Puncturing and shortening will play a fundamental role in the attack. Recall that for a given code $\mathscr{C} \subset \mathbb{F}_q^n$ and a subset $\mathcal{I}$ of code positions the *punctured* code $\mathcal{P}_{\mathcal{I}}(\mathscr{C})$ and *shortened* code $\mathcal{S}_{\mathcal{I}}(\mathscr{C})$ are defined as:

$$\mathcal{P}_{\mathcal{I}}(\mathscr{C}) \overset{\text{def}}{=} \left\{ (c_i)_{i \notin \mathcal{I}} \mid \boldsymbol{c} \in \mathscr{C} \right\};$$
$$\mathcal{S}_{\mathcal{I}}(\mathscr{C}) \overset{\text{def}}{=} \left\{ (c_i)_{i \notin \mathcal{I}} \mid \exists \boldsymbol{c} = (c_i)_i \in \mathscr{C} \text{ such that } \forall i \in \mathcal{I},\ c_i = 0 \right\}.$$

Given a subset $\mathcal{I}$ of the set of coordinates of a vector $\boldsymbol{u}$, we denote by $\mathcal{P}_{\mathcal{I}}(\boldsymbol{u})$ the vector $\boldsymbol{u}$ *punctured* at $\mathcal{I}$, that is to say, *indexes that are in* $\mathcal{I}$ *are removed.*

First let us recall the influence of these operations on GRS codes.

Lemma 2. *Let $\boldsymbol{x}, \boldsymbol{y}$ be two n–tuples of element sof $\mathbb{F}_q$ such that $\boldsymbol{x}$ has pairwise distinct entries and $\boldsymbol{y}$ has only nonzero entries. Let $k < n$ and $\mathcal{I} \subseteq \{1, \ldots, n\}$. Then*

$$\mathcal{P}_{\mathcal{I}}(\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})) = \mathbf{GRS}_k(\mathcal{P}_{\mathcal{I}}(\boldsymbol{x}), \mathcal{P}_{\mathcal{I}}(\boldsymbol{y})) \tag{4}$$
$$\mathcal{S}_{\mathcal{I}}(\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})) = \mathbf{GRS}_{k-|\mathcal{I}|}(\mathcal{P}_{\mathcal{I}}(\boldsymbol{x}), \boldsymbol{y}_{\mathcal{I}}), \tag{5}$$

for some $\boldsymbol{y}_{\mathcal{I}} \in \mathbb{F}_q^{n-|\mathcal{I}|}$ depends only on $\boldsymbol{y}$ and $\mathcal{I}$.

Next, with these notions at hand, it follows that the dual of the public code punctured in $\mathcal{J}_2$ is very close to a GRS code. We will also need to understand the structure of versions of this code which are shortened in positions belonging to $\mathcal{J}_1$ and then punctured in $\mathcal{J}_2$. It turns out that these codes too are close to GRS codes. First of all, puncturing $\mathscr{C}_{\text{pub}}^{\perp}$ in the positions belonging to $\mathcal{J}_2$ gives “almost” a GRS code, as shown by:

Lemma 3. *Let $\boldsymbol{u} = (u_i)_{i \in \mathcal{J}_1}$ and $\boldsymbol{v} = (v_i)_{i \in \mathcal{J}_1}$ be vectors in $\mathbb{F}_q^{n-|\mathcal{J}_2|}$ defined by*

$$u_i = x_{j(i)}$$
$$v_i = T_{ij(i)} y_{j(i)}.$$

Let $\mathscr{D} \overset{def}{=} \mathscr{C}_{sec}^{\perp} \boldsymbol{T}^T$, then

$$\mathcal{P}_{\mathcal{J}_2}(\mathscr{D}) \subseteq \mathbf{GRS}_{n-k}(\boldsymbol{u}, \boldsymbol{v}). \tag{6}$$

Lemma 4. *Let λ and μ be vectors of $\mathbb{F}_q^n$ such that $\boldsymbol{R}^T = \lambda^T \mu$ and let $\mathscr{C}_{sec}^{\perp}(\lambda) \overset{def}{=} \mathscr{C}_{sec}^{\perp} \cap <\lambda>^{\perp}$, $\mathscr{C}_{pub}^{\perp}(\lambda) \overset{def}{=} \mathscr{C}_{sec}^{\perp}(\lambda)(\boldsymbol{T}^T + \boldsymbol{R}^T)$. Then,*

$$\mathcal{P}_{\mathcal{J}_2}\left(\mathscr{C}_{pub}^{\perp}(\lambda)\right) \subseteq \mathbf{GRS}_{n-k}(\boldsymbol{u}, \boldsymbol{v}), \tag{7}$$

Moreover if $\mathcal{J}_1$ contains an information set[7] *of $\mathscr{C}_{sec}^{\perp}\boldsymbol{T}^T$ and $\boldsymbol{T}^T$ is invertible, then there exist $\boldsymbol{a}$ and $\boldsymbol{b}$ in $\mathbb{F}_q^{n-|\mathcal{J}_2|}$ such that for any $\boldsymbol{c}$ in $\mathcal{P}_{\mathcal{J}_2}\left(\mathscr{C}_{pub}^{\perp}\right)$, there exists a vector $\boldsymbol{p}$ in $\mathbf{GRS}_{n-k}(\boldsymbol{u},\boldsymbol{v})$ for which*

$$\boldsymbol{c} = \boldsymbol{p} + (\boldsymbol{p}\cdot\boldsymbol{b})\boldsymbol{a}. \tag{8}$$

In particular, $\mathcal{P}_{\mathcal{J}_2}\left(\mathscr{C}_{pub}^{\perp}\right) \subseteq \mathbf{GRS}_{n-k}(\boldsymbol{u},\boldsymbol{v}) + <\boldsymbol{a}>$.

If we puncture with respect to $\mathcal{J}_2$ shortened versions of $\mathscr{C}_{\text{pub}}^{\perp}$ in positions belonging to $\mathcal{J}_1$, then we observe a similar phenomenon, namely

Lemma 5. *Let $\mathcal{I}_1$ be a subset of code positions which is a subset of $\mathcal{J}_1$. Let $s \overset{def}{=} |\mathcal{I}_1|$ and assume that $s \leqslant n-k$. Then there exist vectors $\boldsymbol{a},\boldsymbol{u},\boldsymbol{v}$ in $\mathbb{F}_q^{n-s-|\mathcal{J}_2|}$ such that:*

$$\mathcal{P}_{\mathcal{J}_2}\left(\mathcal{S}_{\mathcal{I}_1}\left(\mathscr{C}_{pub}^{\perp}\right)\right) \subseteq \mathscr{E} + <\boldsymbol{a}> \tag{9}$$

and $\mathscr{E}$ is a subcode of $\mathbf{GRS}_{n-k-s}(\boldsymbol{u},\boldsymbol{v})$.

4 Key-Recovery Attack

4.1 Outline

Our key-recovery attack starts with a parity-check matrix $\boldsymbol{H_{pub}}$ of the (public) code $\mathscr{C}_{\text{pub}}$. The main goal is to recover matrices $\boldsymbol{T}$ and $\boldsymbol{R}$, where $\boldsymbol{H_{pub}}(\boldsymbol{T}^T + \boldsymbol{R}^T)^{-1}$ is a parity check matrix of a GRS code, $\boldsymbol{T}$ is a low density square matrix and $\boldsymbol{R}$ a rank 1 matrix. Recall that in our terminology, rows of $\boldsymbol{T}$ belonging to $\mathcal{J}_1$ are positions of degree 1, and those in $\mathcal{J}_2$ are positions of degree 2. It implies, thanks to (3), that some columns of $\boldsymbol{H_{pub}}$ belong to $\mathcal{J}_1$ and the others are in $\mathcal{J}_2$.

Our attack is composed of three mains steps having the following objectives:

1. Detecting columns of $\boldsymbol{H_{pub}}$ that belong to $\mathcal{J}_2$, and then deducing those of $\mathcal{J}_1$.

[7] In coding theory, an *information set* of a code $\mathscr{C}$ of dimension k is a set of k positions $\mathcal{I}$ such that the knowledge of a codeword $\boldsymbol{c} \in \mathscr{C}$ on the positions in $\mathcal{I}$ determines entirely the codeword. Equivalently, if $\boldsymbol{G}$ denotes a $k \times n$ generator matrix of the code, then the $k \times k$ submatrix of $\boldsymbol{G}$ given by extracting the columns indexed by $\mathcal{I}$ is invertible.

2. Transforming columns of $\mathcal{J}_2$ into degree 1 columns by linear combinations with columns of $\mathcal{J}_1$.
3. At this stage, the public code has been transformed into another code $\mathscr{C}$ such that there exists a secret GRS code $\mathscr{C}'_{\text{sec}}$ and a matrix $\mathit{\Pi} + \boldsymbol{R}'$ where $\mathit{\Pi}$ is a permutation matrix and $\boldsymbol{R}'$ is rank-1 matrix such that:
$$\mathscr{C} = \mathscr{C}'_{\text{sec}}(\mathit{\Pi} + \boldsymbol{R}'). \tag{10}$$
The third step consists then in applying the attack developed in [11] which is purposely devised to recover a pair $(\mathit{\Pi}, \boldsymbol{R}')$ from $\mathscr{C}$ as outlined in Section 2.1.

The purpose of the next sections is to describe more precisely the first two steps of the attack.

4.2 A distinguisher of the public code

The attack uses in a crucial way a distinguisher which discriminates the public code from a random code of the same dimension. It is based on square code considerations. The point is the following: if we shorten the dual $\mathscr{C}_{\text{pub}}^{\perp}$ of the public code in a large enough set of positions $\mathcal{I}$, then the square code $\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)^2$ has dimension strictly smaller than that of $\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{rand}}^{\perp}\right)\right)^2$ where $\mathscr{C}_{\text{rand}}$ is a random code of the same dimension as $\mathscr{C}_{\text{pub}}$. The code $\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{rand}}^{\perp}\right)\right)^2$ has dimension which is typically $\min\left\{n - |\mathcal{I}|, \binom{k_{\mathcal{I}}+1}{2}\right\}$ where $k_{\mathcal{I}}$ stands for the dimension of $\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{rand}}^{\perp}\right)$. In general, $k_{\mathcal{I}}$ is equal to $n - k - |\mathcal{I}|$ since $\dim \mathscr{C}_{\text{rand}}^{\perp} = \dim \mathscr{C}_{\text{pub}}^{\perp} = n - k$ whereas we generally have:
$$\dim\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)^2 \leqslant 3(n-k) + |\mathcal{J}_2| - 3|\mathcal{I}| - 1. \tag{11}$$

In other words, when $3(n-k)+|\mathcal{J}_2|-3|\mathcal{I}|-1 < \min\left\{n - |\mathcal{I}|, \binom{k_{\mathcal{I}}+1}{2}\right\}$ we expect to distinguish $\mathscr{C}_{\text{pub}}$ from a random code of the same dimension. We write here "generally" because there are some exceptional cases where such an inequality does not hold. However in the case when $\mathcal{I} \subset \mathcal{J}_1$, this inequality always holds.

Proposition 4. *Let* $\mathcal{I} \subseteq \mathcal{J}_1$*, then* $\dim\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\mathit{pub}}^{\perp}\right)\right)^2 \leqslant 3(n-k) - 3|\mathcal{I}| - 1 + |\mathcal{J}_2|$.

Remark 4. It turns out that a similar inequality also generally holds when $\mathcal{I}$ contains degree 2 positions. However in this case, the situation is more complicated and it might happen in rare cases that this upper-bound is not met but, roughly speaking, when it happens, the actual result remains *close* to this upper bound. Experimentally, we observed that (11) was satisfied even when $\mathcal{I}$ contained positions of $\mathcal{J}_2$.

Remark 5. The use of shortening is important since in general the (dual) public code itself is non distinguishable because its square equals the whole ambient space. However, for a part of the parameters proposed in [2], the dual public code is distinguishable from a random code without shortening. See §6 for further details.

4.3 Description of the attack

First step – Distinguishing between positions in $\mathcal{J}_1$ and $\mathcal{J}_2$ Roughly speaking the attack builds upon an algorithm which allows to distinguish between a position of degree 1 and a position of degree 2. It turns out now that once we are able to distinguish the public code from a random one by shortening it in a set of positions $\mathcal{I}$ such that:

$$\dim \left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)^2 < \min \left\{n - |\mathcal{I}|, \binom{n-k-|\mathcal{I}|+1}{2}\right\}, \qquad (12)$$

we can puncture $\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)$ in a position i that does not belong to $\mathcal{I}$ and this allows to distinguish degree 1 positions from degree 2 positions. The dimension of the square code of this punctured code will differ drastically when i is a degree 1 position (or a certain type of degree 2 position) or a "usual" degree 2 position. When i is a degree 1 position it turns out that

$$\dim \left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)^2 = \dim \left(\mathcal{P}_i\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)\right)^2, \qquad (13)$$

whereas for "usual" degree 2 positions we observe that

$$\dim \left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)^2 = \dim \left(\mathcal{P}_i\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)\right)^2 + 1. \qquad (14)$$

Sometimes (in the "non usual" cases), we can have positions of degree 2 for which

$$\dim \left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)^2 = \dim \left(\mathcal{P}_i\left(\mathcal{S}_{\mathcal{I}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)\right)\right)^2$$

as for degree 1 positions. This happens for instance if shortening in $\mathcal{I}$ "induces" a degree 1 position in i. This arises mostly when the position i of degree 2 is such that $j(i) = \{j_1, j_2\}$ where either $j_1 = j(i')$ or $j_2 = j(i')$ for a position i' of degree 1 that belongs to $\mathcal{I}$. This phenomenon really depends on the choice of $\mathcal{I}$. However, by choosing several random subsets $\mathcal{I}$ we quickly find a shortening set $\mathcal{I}$ for which the degree 2 position we want to test behaves as predicted in (14).

Procedure to compute $\mathcal{J}_2$

- Choose a set of random subsets $\mathcal{I}_1, \dots, \mathcal{I}_s$ (in our experimentations we always chose $s \approx 20$) whose cardinals satisfy (12).
- For $i = 1, \dots, s$ compute $\mathcal{S}_{\mathcal{I}_i}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)^2$ and call $\mathcal{J}_2(i)$ this set of positions satisfying
$$\dim \mathcal{S}_{\mathcal{I}_i}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)^2 \neq \dim \mathcal{P}_j\left(\mathcal{S}_{\mathcal{I}_i}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)^2\right).$$
- Set $\mathcal{J}_2 = \mathcal{J}_2(1) \cup \cdots \cup \mathcal{J}_2(s)$.

Second step – Transforming degree 2 positions into degree 1 ones

Proposition 5. *Let $i_1 \in \mathcal{J}_1$ and $i_2 \in \mathcal{J}_2$ be a position associated to i_1. Let $\boldsymbol{D}(\alpha, i_1, i_2)$ be an $n \times n$ matrix which is the identity matrix with an additional entry in column i_2 and row i_1 that is equal to α. Define $\mathscr{C} \overset{def}{=} \mathscr{C}_{pub}^{\perp}\boldsymbol{D}_{\alpha,i_1,i_2}$. If $\alpha = -\frac{T_{i_2 j_1}}{T_{i_1 j_1}}$, then there exists $\boldsymbol{R}'$ of rank at most one such that*
$$\mathscr{C} = \mathscr{C}_{sec}^{\perp}(\boldsymbol{T}'^{T} + \boldsymbol{R}'^{T}) \tag{15}$$
where $\boldsymbol{T}'$ differs from $\boldsymbol{T}$ only in row i_2 and column j_1, the corresponding entry being now equal to 0.

This proposition is exploited as follows, we first compute for a degree 1 position i_1 the set of degree 2 positions i_2 such that $j(i_1) \in j(i_2)$. These positions i_2 can be detected by checking if i_2 has now become a degree 1 position for $\mathcal{S}_{\{i_1\}}\left(\mathscr{C}_{\text{pub}}^{\perp}\right)$ (this is the case if and only if $j(i_1) \in j(i_2)$). Once such a pair (i_1, i_2) has been found we try all possible values for $\alpha \in \mathbb{F}_q^{\times}$ until we obtain a code $\mathscr{C}$ for which the corresponding $\boldsymbol{T}'$ contains a row of index i_2 which is now of Hamming weight 1. That is to say: i_2

became a position of degree 1 for $\mathscr{C}$. This can be easily checked by using the previous technique to distinguish between a position of degree 1 or 2.

In other words, when we are successful, we obtain a new code $\mathscr{C}$ for which there is one more row of weight 1. We iterate this process by replacing $\mathscr{C}_{\text{pub}}^{\perp}$ by $\mathscr{C}$ and $\mathcal{J}_1$ by $\mathcal{J}_1 \cup \{i_2\}$ until we do not find such pairs (i_1, i_2). For the values of m chosen in [2] and with rows of $\boldsymbol{T}$ which were all of weight 1 or 2 we ended up with $\boldsymbol{T}'$ which was a permutation matrix and a code $\mathscr{C}$ which was linked to the secret code by

$$\mathscr{C} = \mathscr{C}_{\text{sec}}^{\perp}(\Pi + \boldsymbol{R}')$$

where Π is a permutation matrix and $\boldsymbol{R}'$ a matrix of rank at most 1. To finish the attack, we just apply the attack described in [11, Sec.4] to recover $\mathscr{C}_{\text{sec}}$.

Case of remaining degree-2 positions

It could happen that the previoulsy decribed method is unsufficient to transform every degree 2 position into a degree 1. It could for instance happen if there is a position i of degree 2 such that for all position i' of degree 1, $j(i') \notin j(i)$. In such a situation, no position of degree 1 can be used to eliminate this position of degree 2.

This problem can be addressed as soon as the set of positions of degree 1 contains an information set of the code. We describe the strategy to conclude the attack in such a situation.

Let $\mathscr{C}$ be the code obtained after performing the two steps of the attack and assume that there remains as nonempty set $\mathcal{J}_2$ of positions of degree 2, which are known (since they have been identified during the first step of the attack). Here is the strategy

1. Puncture $\mathscr{C}$ at $\mathcal{J}_2$. The punctured code is of the form
$$\mathscr{C}'(\boldsymbol{I} + \boldsymbol{R}') \tag{16}$$
where $\mathscr{C}'$ is a GRS code, $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{R}'$ a rank 1 matrix.
2. Perform the attack of [11] on $\mathcal{P}_{\mathcal{J}_2}(\mathscr{C})$. We get the knowledge of a support $\boldsymbol{x}'$ a multiplier $\boldsymbol{y}'$ and a rank 1 matrix $\boldsymbol{R}'$ such that
$$\mathscr{C}' = \mathbf{GRS}_k\left(\boldsymbol{x}', \boldsymbol{y}'\right)(\boldsymbol{I} + \boldsymbol{R}').$$

Moreover, we are able to identify the polynomials $P_1, \dots, P_k$ yielding the rows of the public matrix $\boldsymbol{G_{pub}}$.

3. For all $x \in \mathbb{F}_q$ which is not in the support $\boldsymbol{x}'$ of $\mathscr{C}'$, compute the column

$$\begin{pmatrix} P_1(x) \\ P_2(x) \\ \vdots \\ P_k(x) \end{pmatrix}$$

and join it to the matrix $\boldsymbol{G_{pub}}$. By this manner we get new positions of degree 1 which can be used to eliminate the remaining positions of degree 2.

Remark 6. In our experiments, this situation never happened: we have always eliminated all the degree 2 positions using Proposition 5.

5 Limits and Complexity of the Attack

5.1 Choosing appropriately the cardinality of $\mathcal{I}$

By definition of the density m, the sets $\mathcal{J}_1$ and $\mathcal{J}_2$ have respective cardinalities $(2-m)n$ and $(m-1)n$. In what follows, we denote by R the rate of the public code namely $R = k/n$. Let us recall that the attack shortens the dual of a public code which is of dimension $n-k$. The cardinality of $\mathcal{I}$ is denoted by a. We list the constraints we need to satisfy for the success of the attack.

1. The shortened code should be reduced to the zero space, which implies that $a < n-k$.
2. The code punctured at $\mathcal{J}_2$ must contain an information set, that is to say:
$$n - k \leqslant |\mathcal{J}_1|. \tag{17}$$
It is clear that (17) is equivalent to $m \leqslant 1 + R$.
3. The computed square code in Proposition 4 should also be different from the full space which implies:
$$3(n-k-a) + |\mathcal{J}_2| - 1 < n - a \tag{18}$$
One can easily check that (18) is equivalent to:
$$a \geqslant \frac{1}{2}\Big((1+m)n - 3k\Big). \tag{19}$$

4. Finally, to have good chances that the dimension of the square code reaches the upper bound given by Proposition 4, we also need:

$$3(n-k-a)+|\mathcal{J}_2|-1<\binom{n-k-a+1}{2} \tag{20}$$

which is equivalent to the inequality:

$$a^2+\Big(5-2(n-k)\Big)a+(n-k)^2-5(n-k)+2(1-m)n \geqslant 0 \tag{21}$$

Considering (21) as an inequality involving a degree-2 polynomial in a, we can check that its discriminant is equal to $\Delta \stackrel{\text{def}}{=} 8(m-1)n+25$, so that its roots are a_0 and a_1 where:

$$a_0 \stackrel{\text{def}}{=} n-k-\frac{5}{2}-\frac{1}{2}\sqrt{\Delta} \quad \text{and} \quad a_1 \stackrel{\text{def}}{=} n-k-\frac{5}{2}+\frac{1}{2}\sqrt{\Delta}. \tag{22}$$

Let us recall that in order to have (21) satisfied, we should have $a \leqslant a_0$ or $a \geqslant a_1$. Because of the constraint $a < n-k$ and since $a_1 > n-k$, the only case to study is $a \leqslant a_0$. Combining (19) with $a \leqslant a_0$, we obtain:

$$\frac{1}{2}\Big((1+m)n-3k\Big) \leqslant a_0.$$

which is equivalent to the following inequality involving this time a degree-2 polynomial in m:

$$n^2m^2+2n(1-n-k)m+2kn+k^2-10k+n^2-2n \geqslant 0. \tag{23}$$

The discriminant of this polynomial is $n^2(8k+1)$ and the roots are:

$$m_0 \stackrel{\text{def}}{=} 1+R-\frac{1}{n}-\sqrt{\frac{8}{n}R+\frac{1}{n^2}} \quad \text{and} \quad m_1 \stackrel{\text{def}}{=} 1+R-\frac{1}{n}+\sqrt{\frac{8}{n}R+\frac{1}{n^2}}\cdot$$

Because of the fact that $m \leqslant 1+R$ from (17), and since $m_1 > 1+R$, we conclude that the attack can be applied as long as $m \leqslant m_0$, that is to say:

$$m \leqslant 1+R-\frac{1}{n}-\sqrt{\frac{8}{n}R+\frac{1}{n^2}}\cdot \tag{24}$$

5. Finally, the last step of the attack consists in performing the attack of [11].

Remark 7. This upper-bound is roughly $1+R$. In [2], the authors suggest to choose $m \approx 2-R$ for rates $R > \frac{1}{2}$, which is well within the reach of the present attack.

5.2 Estimating the complexity

As explained in Proposition 2, the square of a code of dimension k and length n can be computed in $O(n^2k^2)$. Let us study the costs of the steps of the attack.

- **Step 1. Finding the positions of degree 2.** For a constant number of subsets $\mathcal{I}$ of length $a \leqslant a_0$ where a_0 is defined in (22), we shorten $\mathscr{C}_{\text{pub}}^{\perp}$ and compute its square. If a is close to a_0 then, the shortened code has dimension $n - k - a = O(\sqrt{n})$. Hence, the computation of its square costs $O(n^3)$. Thus this first step costs $O(n^3)$ operations in $\mathbb{F}_q$.
- **Step 2. Transforming degree-2 positions into degree 1 positions.** This is the most expensive part of the attack. For a given position $i_1 \in \mathcal{J}_1$, the computation of positions i_2 of degree 2 such that[8] $j(i_1) \in j(i_2)$ consists essentially in shortening the dual public code at i_1 and applying to the shortened code the first step. This costs $O(n^3)$. Then, the application of Proposition 5 to transform i_2 requires to proceed to at most q linear combinations and, for each one, to check whether the position became of degree 1. Each check has mostly the same cost as the first step, that is $O(n^3)$. Thus, the overall cost to reduce one position of degree 2 is $O(n^4)$ and hence the cost of this second step is $O(n^5)$.
- **Step 3.** According to [11], it is in $O(n^6)$.

6 Experimental Results

Table 1 gathers experimental results obtained when the attack is programmed in Magma V2.20-3 [9]. The attacked parameters are taken from [2, Tables 3 & 4] The timings given are obtained with Intel® Xeon 2.27GHz and 72 Gb of RAM. Our programs are far from being optimized and probably improved programs could provide better timings and memory usage.

The running times for codes of length 346 are below 5 hours and those for codes of length 546 can be a bit longer than one day. The total memory usage remains below 100Mb for codes of length 346 and 500Mb for codes of length 546.

[8] Equivalently, there exists an integer j such that $\boldsymbol{T}_{i_1,j} \neq 0$ and $\boldsymbol{T}_{i_2,j} \neq 0$.

(q, n, k, z)	m	Step 1	Step 2
(347, 346, 180, 1)	1.471	15s	18513s (≈5 hours)
(347, 346, 188, 1)	1.448	8s	10811s (≈3 hours)
(347, 346, 204, 1)	1.402	10s	8150s (≈2.25 hours)
(347, 346, 228, 1)	1.332	15s	9015s (≈2.5 hours)
(347, 346, 252, 1)	1.263	36s	10049s (≈2.75 hours)
(347, 346, 268, 1)	1.217	3s	14887s (≈4 hours)
(347, 346, 284, 1)	1.171	3s	7165s (≈2 hours)
(547, 546, 324, 1)	1.401	60s	58778s (≈16 hours)
(547, 546, 340, 1)	1.372	83s	72863s (≈20 hours)
(547, 546, 364, 1)	1.328	100s	72343s (≈20 hours)
(547, 546, 388, 1)	1.284	170s	85699s (≈24 hours)
(547, 546, 412, 1)	1.240	15s	157999s (≈43 hours)
(547, 546, 428, 1)	1.211	15s	109970s (≈30,5 hours)

Table 1. Running times

Remark 8. Since the algorithms include many random choices, the identification of pairs (i_1, i_2), where $i_1 \in \mathcal{J}_1$ and $i_2 \in \mathcal{J}_2$ such that $j(i_1) \in j(i_2)$ might happen quickly or be rather long. This explains the important gaps between different running times.

Remark 9. Actually some parameters proposed in [2] were directly distinguishable without even shortening. This holds for $(q, n, k) = (347, 346, 268)$, $(q, n, k) = (347, 346, 284)$ and $(q, n, k) = (547, 546, 428)$ with m respectively equal to 1.217, 1.171 and 1.211. This explains why the first step is quicker for these examples.

Remark 10. The examples $[346, 180]_{347}$ and $[346, 188]_{347}$ do not satisfy (24). However, they are distinguishable by shortening and squaring and the attack works on them. Because of some cancellation phenomenon for positions of degree 2 which we do not control, it may happen that the upper bound in Proposition 4 is not sharp and that some shortenings of $\mathscr{C}_{\text{pub}}^{\perp}$ turn out to be distinguishable while our formulas could not anticipate it.

The above remark is of interest since it points out that our attack might work for values of m above $1 + R$.

7 Concluding Remarks

The papers [4, 3, 1, 2] can be seen as an attempt of replacing the permutation matrix in the McEliece scheme by a more complicated transformation. Instead of having as in the McEliece scheme a relation between the secret code $\mathscr{C}_{\text{sec}}$ and the public code $\mathscr{C}_{\text{pub}}$ of the form $\mathscr{C}_{\text{sec}} = \mathscr{C}_{\text{pub}}\mathit{\Pi}$ where $\mathit{\Pi}$ is a permutation matrix, it was chosen in [4, 3] that

$$\mathscr{C}_{\text{sec}} = \mathscr{C}_{\text{pub}}\boldsymbol{T}$$

where $\boldsymbol{T}$ is a sparse matrix of density m or as

$$\mathscr{C}_{\text{sec}} = \mathscr{C}_{\text{pub}}(\boldsymbol{T} + \boldsymbol{R})$$

where $\boldsymbol{T}$ is as before and $\boldsymbol{R}$ is of very small rank z (the case of rank 1 being probably the only practical way of choosing this rank as will be discussed below) as in [1, 2]. It was advocated that this allows to use for the secret code $\mathscr{C}_{\text{sec}}$, codes which are well known to be weak in the usual McEliece cryptosystem such as LDPC codes [4, 3] or GRS codes [1, 2]. Interestingly enough, it turns out that for LDPC codes this basically amounts choosing a McEliece system where the density of the parity-check matrix is increased by a large amount and the error-correction capacity is decreased by the same multiplicative constant. The latter approach has been studied in [21], it leads to schemes with slightly larger decoding complexity but that have at least partial security proofs.

In the case of GRS codes, the first attempt [1] of choosing for $\boldsymbol{T}$ a permutation matrix was broken in [11, Sec.4]. It was suggested later on [2] that this attack can be avoided by choosing $\boldsymbol{T}$ of larger density. In order to reduce the public key size when compared to the McEliece scheme based on Goppa codes, rather moderate values of m between 1 and 2 ($m = 1.4$ for instance) were chosen in [2]. We show here that the parameters proposed in [2] can be broken by a new attack computing first the dimension of the square code of shortened versions of the dual of the public code and using this to reduce the problem to the original problem [1] when $\boldsymbol{T}$ is a permutation matrix. This attack can be avoided by choosing larger values for m and/or z, but this comes at a certain cost as we now show.

Increasing z. Increasing $z = 1$ to larger values of z avoids the attack given here, though some of the ideas of [11] might be used in this new context to get rid of the $\boldsymbol{R}$ part in the scheme and might lead to an attack of reasonable complexity when $z = 2$ by trying first to guess several codewords which lie in the code $\mathscr{C} \overset{\text{def}}{=} \mathscr{C}_{\text{sec}}^{\perp} \boldsymbol{T}^{T} \cap \mathscr{C}_{\text{pub}}^{\perp}$ (this code is of codimension at least z in $\mathscr{C}_{\text{pub}}^{\perp}$). Once $\mathscr{C}$ is found, we basically have to recover $\boldsymbol{T}$ and the approach used in this paper can be applied to it. To avoid such an attack, rather large values of z have to be chosen, but the decryption cost becomes prohibitive by doing so. Indeed, decryption time is of order $q^{z}C$ where C is the decoding complexity of the underlying GRS code. Choosing $z = 2$ is of questionable practical interest and $z > 2$ becomes probably unreasonable.

Increasing m**.** Choosing values for m close enough to 2 will avoid the attack presented here. However this also reduces strongly the gain in key size when compared to the McEliece scheme based on Goppa or alternant codes. Indeed, assume for simplicity $m = 2$. We can use in such a case for the secret code a GRS code over $\mathbb{F}_q$ of dimension $k = n - 2t$ and add errors of weight $\leqslant \frac{t}{2}$ in the BBCRS scheme. The public key size of such a scheme is however not better than choosing in the McEliece scheme a Goppa code of the same dimension $n-2t$ but which is the subfield subcode of a GRS code over $\mathbb{F}_{q^2}$ of dimension $n - t$, and which can also correct $\frac{t}{2}$ errors. This Goppa code has the very same parameters and provides the same security level. For this reason, one loses the advantages of using GRS codes when choosing m close to 2. Thus, to have interesting key sizes and to resist to our attack m should be smaller than 2 and larger than $1 + R$. One should however be careful, since, as explained in §6, it is still unclear whether the attack fails for m closely above $1 + R$.

On the other hand, it might be interesting for theoretical reasons to understand better the security of the BBCRS scheme for larger values of m. There might be a closer connection than what it looks between the BBCRS scheme with density m and the usual McEliece scheme with (possibly non-binary) Goppa codes of extension degree m. The connection is that the case $m = 2$ is in both cases the limiting case where the distinguishing approach of [11, 14] might work (in [14], the attack only works because wild Goppa codes are studied and this brings an additional power to the distinguishing attack). It should also be added that it might be interesting to study the choice of $\mathscr{C}_{\text{sec}}$ being an LDPC code and $\mathscr{C}_{\text{sec}} = \mathscr{C}_{\text{pub}}(\boldsymbol{T} + \boldsymbol{R})$ since here adding $\boldsymbol{R}$ of small rank can also change rather drastically the property of $\mathscr{C}_{\text{pub}}$ being an LDPC code (which is at the heart of the key attacks on McEliece schemes based on LDPC codes).

References

1. Baldi, M., Bianchi, M., Chiaraluce, F., Rosenthal, J., Schipani, D.: Enhanced public key security for the McEliece cryptosystem. preprint (2011), arXiv:1108.2462v2
2. Baldi, M., Bianchi, M., Chiaraluce, F., Rosenthal, J., Schipani, D.: Enhanced public key security for the McEliece cryptosystem. J. of Cryptology (2014), published online: August 15, 2014, see http://link.springer.com/article/10.1007/s00145-014-9187-8 and also ArXiv:1108.2462v4
3. Baldi, M., Bodrato, M., Chiaraluce, G.: A new analysis of the McEliece cryptosystem based on QC-LDPC codes. In: Security and Cryptography for Networks (SCN). pp. 246–262 (2008)
4. Baldi, M., Chiaraluce, G.F.: Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC codes. In: IEEE International Symposium on Information Theory. pp. 2591–2595. Nice, France (Mar 2007)
5. Barbulescu, R., Gaudry, P., Joux, A., Thomé, E.: A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology - EUROCRYPT 2014. Lecture Notes in Comput. Sci., vol. 8441, pp. 1–16. Springer Berlin Heidelberg (2014)
6. Berger, T.P., Loidreau, P.: How to mask the structure of codes for a cryptographic use. Des. Codes Cryptogr. 35(1), 63–79 (2005)
7. Bernstein, D.J., Lange, T., Peters, C.: Wild McEliece. In: Selected Areas in Cryptography. pp. 143–158 (2010)
8. Biswas, B., Sendrier, N.: McEliece cryptosystem implementation: theory and practice. In: Buchmann, J., Ding, J. (eds.) Post-Quantum Cryptography, Second International Workshop, PQCrypto 2008, Cincinnati, OH, USA, October 17-19, 2008, Proceedings. Lecture Notes in Comput. Sci., vol. 5299, pp. 47–62. Springer (2008)
9. Bosma, W., Cannon, J.J., Playoust, C.: The Magma algebra system I: The user language. J. Symbolic Comput. 24(3/4), 235–265 (1997)
10. Cascudo, I., Cramer, R., Mirandola, D., Zémor, G.: Squares of random linear codes, arXiv:1407.0848v1.
11. Couvreur, A., Gaborit, P., Gauthier-Umaña, V., Otmani, A., Tillich, J.P.: Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes. Des. Codes Cryptogr. pp. 1–26 (2014)
12. Couvreur, A., Gaborit, P., Gauthier-Umaña, V., Otmani, A., Tillich, J.P.: Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes. In: International Workshop on Coding and Cryptography, WCC 2013, Bergen, Norway (Apr 15-19 2013)
13. Couvreur, A., Márquez-Corbella, I., Pellikaan, R.: A polynomial time attack against algebraic geometry code based public key cryptosystems. In: IEEE International Symposium on Information Theory (ISIT 2014). Honolulu, US (2014)
14. Couvreur, A., Otmani, A., Tillich, J.P.: Polynomial time attack on wild McEliece over quadratic extensions. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology – EUROCRYPT 2014, Lecture Notes in Comput. Sci., vol. 8441, pp. 17–39. Springer Berlin Heidelberg (2014)
15. Faugère, J.C., Gauthier, V., Otmani, A., Perret, L., Tillich, J.P.: A distinguisher for high rate McEliece cryptosystems. In: Proceedings of the Information Theory Workshop 2011, ITW 2011. pp. 282–286. Paraty, Brasil (2011)
16. Faugère, J.C., Gauthier-Umaña, V., Otmani, A., Perret, L., Tillich, J.P.: A distinguisher for high-rate McEliece cryptosystems. IEEE Trans. Inform. Theory 59(10), 6830–6844 (2013)

17. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North–Holland, Amsterdam, fifth edn. (1986)
18. Márquez-Corbella, I., Martínez-Moro, E., Pellikaan, R.: The non-gap sequence of a subcode of a generalized Reed–Solomon code. Des. Codes Cryptogr. 66(1-3), 317–333 (2013)
19. Márquez-Corbella, I., Pellikaan, R.: Error-correcting pairs for a public-key cryptosystem. preprint (2012)
20. McEliece, R.J.: A Public-Key System Based on Algebraic Coding Theory, pp. 114–116. Jet Propulsion Lab (1978), dSN Progress Report 44
21. Misoczki, R., Tillich, J.P., Sendrier, N., Barreto, P.S.L.M.: MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In: ISIT. pp. 2069–2073 (2013)
22. Monico, C., Rosenthal, J., Shokrollahi, A.: Using low density parity check codes in the McEliece cryptosystem. In: IEEE International Symposium on Information Theory (ISIT 2000). p. 215. Sorrento, Italy (2000)
23. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Problems Control Inform. Theory 15(2), 159–166 (1986)
24. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: Goldwasser, S. (ed.) Proceedings of the 35th Annual Symposium on the Foundations of Computer Science. pp. 124–134. IEEE Computer Society, Los Alamitos, CA (1994)
25. Sidelnikov, V., Shestakov, S.: On the insecurity of cryptosystems based on generalized Reed-Solomon codes. Discrete Math. Appl. 1(4), 439–444 (1992)
26. Wieschebrink, C.: Cryptanalysis of the Niederreiter Public Key Scheme Based on GRS Subcodes. In: Sendrier, N. (ed.) Post-Quantum Cryptography, Third International Workshop, PQCrypto 2010. Lecture Notes in Comput. Sci., vol. 6061, pp. 61–72. Springer, Darmstadt, Germany (May 2010)