An Encryption Scheme based on Random Split of St-Gen Codes

Simona Samardjiska* and Danilo Gligoroski†,
*Faculty of Computer Science and Engineering,
"Ss Cyril and Methodius" University, Skopje, Republic of Macedonia,
Email: simona.samardjiska@finki.ukim.mk
†Department of Telematics,
Norwegian University of Science and Technology, Trondheim, Norway,
Email: danilog@item.ntnu.no

Abstract

Staircase-Generator codes (St-Gen codes) have recently been introduced in the design of code-based public key schemes and for the design of steganographic matrix embedding schemes. In this paper we propose a method for random splitting of St-Gen Codes and use it to design a new coding based public key encryption scheme. The scheme uses the known list decoding method for St-Gen codes, but introduces a novelty in the creation of the public and private key. We modify the classical approach for hiding the structure of the generator matrix by introducing a technique for splitting it into random parts. This approach counters the weaknesses found in the previous constructions of public key schemes using St-Gen codes. Our initial software implementation shows that encryption using Random Split of St-Gen Codes compared to original St-Gen Codes is slower by a linear factor in the number of random splits of the St-Gen code, while the decryption complexity remains the same.

Keywords

Public Key Cryptography, Code Based Cryptosystems, St-Gen Codes, List Decoding.

I. Introduction

The interest for cryptographic primitives that are resistant to quantum algorithms is increasing in the last 10 years. Beside the academic community [1], the standardization bodies such as NIST and ETSI have recently started initiatives for developing cryptographic standards not based on number theory, with a particular focus on primitives resistant to quantum attacks [2], [3]

One of the schemes believed to be quantum secure is the McEliece public key scheme [4], published as early as 1978. Its security is based on the NP-hardness of the problem of decoding random linear codes.

Recently, an encryption and signature variant of the McEliece scheme based on Staircase-Generator codes was introduced in [6], [8]. For the public keys produced with these codes a distinguisher was proposed by Sendrier and Tillich [9], and recently a very similar distinguishing strategy and an ISD attack was presented as a full and practical key recovery

attack by Moody and Perlner [5]. While the public key schemes based on Staircase-Generator codes have to expose some form of the Staircase-Generator matrices, and thus are prone for structural attacks, there are other use-case scenarios where the Staircase-Generator matrices are not public. For example, the steganographic technique of matrix-embedding [7] use the matrices as private information. It was shown in [7] that these codes almost achieve the information theoretical bound for codelengths much smaller than other matrix-embedding schemes known in the literature.

A. Our contribution

In order to thwart the ISD attacks of [9], [5] against the encryption scheme defined in [6], [8] we introduce a novel idea where we split the public generator matrix into s randomly generated matrices. With this technique, we show that the probability of the attacker obtaining conditions under which the attacks [9], [5] can be mounted, becomes negligible. We provide an initial security analysis of the scheme as well as some concrete parameters and instances.

II. Notation and Preliminaries

Throughout the paper, we will denote by $\mathcal{C} \subseteq \mathbb{F}_2^n$ a binary (n, k) code of length n and dimension k. We will denote the generator matrix of the code by G, and $wt(\mathbf{x})$ will denote the Hamming weight of the word $\mathbf{x}$.

We recall some of the basic definitions and properties for St-Gen codes from [6], [7] and the types of errors used.

Definition 1: Let $k_i, n_i \in \mathbb{N}$, and let $k = k_1 + k_2 + \cdots + k_w$ and $n = k + n_1 + n_2 + \cdots + n_w$. Further, let B_i be a random binary matrix of dimension $\sum_{j=1}^{i} k_j \times n_i$. A linear binary (n, k) code $\mathcal{C}$ with the following generator matrix in standard form:

$$G = \begin{pmatrix} & & \overset{n_1}{} & \overset{n_2}{} & & \\ & & \begin{matrix} k_1 \\ \\ k_2 \end{matrix} \overbrace{\begin{matrix} \\ \end{matrix}} & \overbrace{\begin{matrix} \\ \end{matrix}} & & \\ & I_k & B_1 & B_2 & \cdots & B_w \\ & & 0 & & \cdots & \end{pmatrix} \tag{1}$$

is called Staircase-Generator code (St-Gen code).

Definition 2: Let ℓ be a positive integer and let $p_d \in \mathbb{F}_2[x_1, x_2, \ldots, x_\ell]$ be a multivariate polynomial of degree $\geqslant 2$. We say that E_ℓ is an *error set* if it is the solution set of p_d, i.e. $E_\ell = \{\mathbf{e} \in \mathbb{F}_2^\ell \mid p_d(\mathbf{e}) = 0\}$. We will refer to p_d as the defining polynomial.

We define the *density* of the error set E_ℓ to be $D(E_\ell) = |E_\ell|^{1/\ell}$. We will refer to the integer $\ell > 0$ as the *granulation* of E_ℓ. In [6] it was proven that if two error sets $E_{\ell_1} \subseteq \mathbb{F}_2^{\ell_1}$, $E_{\ell_2} \subseteq \mathbb{F}_2^{\ell_2}$, have the same density ρ, then $D(E_{\ell_1} \times E_{\ell_2}) = \rho$.

Example 1: 1. Let $E_2 = \{\mathbf{x} \in \mathbb{F}_2^2 \mid wt(\mathbf{x}) < 2\} = \{(0, 0), (0, 1), (1, 0)\}$. Then the error set can be described using the defining polynomial $p_d = x_1 x_2$, and for the density of the error set we have $D(E_2) = |E_2|^{1/2} = 3^{1/2}$.

2. Let $E_4 = \{\mathbf{x} \in \mathbb{F}_2^4 \mid 2 \leq wt(\mathbf{x}) \leq 3\}$. Then, the defining polynomial for E_4 is $p_d = 1 + x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + x_3x_4$ and the density is $D(E_4) = D(E_4^m) = (\sum_{i=2}^{3} \binom{4}{i})^{1/4} = 10^{1/4}$ for any positive integer m.

The decoding of St-Gen codes relies on the technique of *list decoding*, a notion that dates back to the work of Elias [10] and Wozencraft [11] in the 1950's. In list decoding, the decoder is allowed to output a list of possible messages one of which is correct. List decoding can handle a greater number of errors than that allowed by unique decoding. In order for the decoding to be efficient, the size of the resulting list has to be polynomial in the code length. The following Proposition from [6] determines the parameters of a St-Gen code that provide an efficient decoding.

Proposition 1 ([6]): Let $\mathcal{C}$ be any binary (n, k) code and $E \subset \mathbb{F}_2^n$ be an error set of density ρ. Let $\mathbf{w}$ be any word of length n, $W_E = \{\mathbf{w} + \mathbf{e} \mid \mathbf{e} \in E\}$ and let $\mathcal{C}_{W_E}$ denote the set of codewords in W_E. Suppose there exists a codeword $\mathbf{c} \in W_E$. Then the expected number of codewords in $W_E \setminus \{\mathbf{c}\}$ is approximately $\rho^n 2^{k-n}$ for large enough n and k.

Let E_ℓ be an error set with density ρ where ℓ divides n and $m = n/\ell$. We recall Alg. 1 from [6], that is an efficient algorithm for decoding a code $\mathcal{C}$, that corrects errors from the set E_ℓ^m.

Algorithm 1 Decoding

Input: Vector $\mathbf{y} \in \mathbb{F}_2^n$, and generator matrix G of the form (1).

Output: A list $L_w \subset \mathbb{F}_2^k$ of valid decodings of $\mathbf{y}$.

Procedure: Let $K_i = k_1 + \cdots + k_i$. Represent $\mathbf{x} \in \mathbb{F}_2^k$ as $\mathbf{x} = \mathbf{x}_1 \parallel \mathbf{x}_2 \parallel \cdots \parallel \mathbf{x}_w$ where each $\mathbf{x}_i$ has length k_i. Similarly, represent $\mathbf{y} \in \mathbb{F}_2^n$, as $\mathbf{y} = \mathbf{y}_0 \parallel \mathbf{y}_1 \parallel \mathbf{y}_2 \parallel \cdots \parallel \mathbf{y}_w$, where each $\mathbf{y}_i$ has length n_i and $|\mathbf{y}_0| = k$. We further identify $\mathbf{y}_0$ with $\mathbf{y}_0 = \mathbf{y}_0[1] \parallel \mathbf{y}_0[2] \parallel \cdots \parallel \mathbf{y}_0[w]$, where each $\mathbf{y}_0[i]$ is of length k_i.

During decoding, we will maintain lists $L_1, L_2, \ldots, L_w$ of possible decoding candidates of length K_i.

Step 0: Set a temporary list $T_0 = L_0$ to contain all possible decodings of the first k_1 coordinates of $\mathbf{y}$: $T_0 \leftarrow \{\mathbf{x}' = \mathbf{y}_0[1] + \mathbf{e} \mid \mathbf{e} \in E^{k_1/\ell}\}$.

Step $1 \leq i \leq w$*:* Perform list-decoding to recover a list of valid decodings:

For each candidate $\mathbf{x}' \in T_{i-1} \subset \mathbb{F}_2^{K_i}$, add to L_i all the candidates for which $\mathbf{x}'B_i + \mathbf{y}_i \in E^{n_i/\ell}$:

$$L_i \leftarrow \{\mathbf{x}' \in T_{i-1} \mid \mathbf{x}'B_i + \mathbf{y}_i \in E^{n_i/\ell}\}.$$

If $i < w$ then create the temporary list T_i of candidates of length K_{i+1} from L_i: $T_i \leftarrow \{\mathbf{x}' \parallel (\mathbf{y}_0[i+1] + \mathbf{e}) \mid \mathbf{x}' \in L_i,\ \mathbf{e} \in E^{k_{i+1}/\ell}\}$.

Return: L_w.

III. A new Encryption Scheme

A key parameter of the new encryption scheme is the number of splits s, that determines the number of summands the generator matrix of the code is split in.

This parameter further determines the nature of the error used during encryption. We have the following:

Definition 3: Let $E_\ell \subset \mathbb{F}_2^\ell$ be an error set of granulation ℓ and let s denote the number of splits. The s-element set $ErrorSplit = \{e_1, \ldots, e_s\}$, where $e_i \in \mathbb{F}_2^\ell, i \in \{1, \ldots, s\}$ is called *A Valid Error Split for* E_ℓ if the sum of its elements permuted with any permutation $\sigma_i \in \mathcal{S}_\ell$

is an element of E_ℓ i.e. it holds that $e = \sum_{i=1}^{s} \sigma_i(e_i) \in E_\ell$.

Example 2: Let $\ell = 4$, $E_\ell = \{\{0,0,0,0\}, \{0,0,1,1\}, \{0,1,0,1\}, \{0,1,1,0\}, \{1,0,1,1\}, \{1,1,0,0\}, \{1,1,0,1\}, \{1,1,1,0\}, \{1,1,1,1\}\}$ and $s = 4$. The 4-element set $\{\{1,0,0,0\}, \{1,1,1,1\}, \{1,1,1,1\}, \{1,1,0,1\}\}$ is *a valid error split for* E_ℓ because the sum of all its elements permuted by any of all possible $4! = 24$ permutations always gives an element in E_ℓ. Note that the elements in a valid error split don't need to belong to the error set E_ℓ, as it is the case with the element $\{1,0,0,0\}$ in this example.

A formal description of the scheme is given through the next four algorithms for key generation, error set generation, encryption and decryption.

Algorithm 2 Key Generation

Parameters: Let $\ell | n$, $m = n/\ell$ and $E \subset \mathbb{F}_2^\ell$ be an error set of granulation ℓ and density ρ. Let s be the number of splits.

Key generation:
The following matrices make up the private key: - A generator matrix G of a binary (n,k) code of the form (1).
- An invertible matrix $S \in \mathbb{F}_2^{k\times k}$.
- An array of permutation matrices $P_1, P_2, \ldots, P_s$ created as follows:
1. Select a permutation π on $\{1,2,\ldots,m\}$, and let $P \in \mathbb{F}_2^{n\times n}$ be the permutation matrix induced by π, so that for any $\mathbf{y} = \mathbf{y}_1 \parallel \mathbf{y}_2 \parallel \ldots \parallel \mathbf{y}_m \in (\mathbb{F}_2^\ell)^m$:

$$\mathbf{y}P = \mathbf{y}_{\pi(1)} \parallel \mathbf{y}_{\pi(2)} \parallel \ldots \parallel \mathbf{y}_{\pi(m)}, \tag{2}$$

i.e., P only permutes the m substrings of $\mathbf{y}$ of length ℓ.
2. For $i := 1$ to s:
- Select randomly m permutations $\sigma_j^i \in \mathcal{S}_\ell$, $j \in \{1,\ldots,m\}$.
- Let P_i be defined by

$$\mathbf{y}P_i = \sigma_1^i(\mathbf{y}_{\pi(1)}) \parallel \sigma_2^i(\mathbf{y}_{\pi(2)}) \parallel \ldots \parallel \sigma_m^i(\mathbf{y}_{\pi(m)}),$$

where $\sigma_j^i(\mathbf{x}) = \sigma_j^i(x_1, x_2, \ldots, x_\ell)$.

The public key is formed as follows:
- Generate uniformly at random $s-1$ matrices $G_1, \ldots, G_{s-1}$ of size $k \times n$ over $\mathbb{F}_2$.
- Set $G_s = G + G_1 + \cdots + G_{s-1}$.
- For all $i \in \{1,2,\ldots,s\}$, set $G_{\text{pub}}^i = SG_iP_i$.

Public key: $G_{\text{pub}}^1, \ldots, G_{\text{pub}}^s$.

Private key: S, G and $P_1, P_2, \ldots, P_s$.

Algorithm 3 Valid Error Splits (ℓ, E_ℓ, s)

Input: Granulation ℓ, error set E_ℓ, number of splits s.
Output: An error set *ErrorSet* of all possible valid error splits.

1: **For all** $(e_1, \ldots, e_s) \in (\mathbb{F}_2^\ell)^s$ **do**
2: **If** $\sum_{i=1}^s \sigma_i(e_i) \in E_\ell \quad \forall (\sigma_1, \ldots, \sigma_s) \in (\mathcal{S}_\ell)^s$ **then**
3: Add $(e_1, \ldots, e_s)$ to *ErrorSet*.
4: Return *ErrorSet*.

Note that Algorithm 3 is run only once at the time of the initialization of the system with parameters ℓ, E_ℓ, s. Even more, in practice, this set can be pre-calculated and publicly available.

Algorithm 4 Encryption $(\mathbf{m}, G^1_{\text{pub}}, \ldots, G^s_{\text{pub}}, ErrorSet)$

Input: Message to be encrypted $\mathbf{m}$, the public key $G^1_{\text{pub}}, \ldots, G^s_{\text{pub}}$ and a set $ErrorSet$ of all possible valid error splits.
Output: A ciphertext $\mathbf{c} = (\mathbf{c}_1, \ldots, \mathbf{c}_s)$.

1: Set $\mathbf{c}_i = \mathbf{m}G^i_{\text{pub}} + \mathbf{e}_i$, $i = 1, \ldots, s$, where $\mathbf{e}_i = (e_{1,i}, \ldots, e_{\frac{n}{l},i})$ and $(e_{j,1}, \ldots, e_{j,s})$, $j = 1, \ldots, \frac{n}{l}$ are randomly drawn from $ErrorSet$.
2: Return $\mathbf{c} = (\mathbf{c}_1, \ldots, \mathbf{c}_s)$

Algorithm 5 Decryption $(\mathbf{c}, S, G, P_1, P_2, \ldots, P_s)$

Input: Ciphertext $\mathbf{c}$, matrix S, the generator matrix G and the permutation matrices $P_1, P_2, \ldots, P_s$.
Output: A decrypted message $\mathbf{m}$.

1: Set $\mathbf{c}'_i = \mathbf{c}_i P_i^{-1}$
2: Set $\mathbf{c}' = \sum_{i=1}^{s} \mathbf{c}'_i$
3: Set $\mathbf{m}'$ as the output of Algorithm 1 (List decoding of $\mathbf{c}'$ with generator matrix G).
4: Set $\mathbf{m} = \mathbf{m}' S^{-1}$
5: Return $\mathbf{m}$

Compared to the original encryption using St-Gen Codes [6], the computational complexity of our encryption Algorithm 4 is slower only by a linear factor s, while the decryption complexity is almost the same (with a small overhead for Step 1 and Step 2 in the decryption Algorithm 5). This was also confirmed experimentaly, using an implementation of the scheme in both C and MAGMA [25].

IV. Security Analysis

A. Weaknesses in a previous scheme based on St-Gen codes

Information Set Decoding (ISD) is a technique first introduced by Prange [12], and later improved several times in the works of Lee and Brickell [13], Leon [14], Stern [15], and many others [16], [17], [18], [19]. ISD algorithms are usually used to find the error vector being used in the encryption of a message $\mathbf{m}$. In this case, the basic idea is the following: Find an information set $\mathcal{I}$ i.e. an index set of k columns of the generator matrix G that form an invertible matrix $G_{\mathcal{I}}$, such that the error vector has a specific error pattern $\mathbf{e}_{\mathcal{I}}$ with respect to $\mathcal{I}$. With the error pattern being correctly guessed, we can find the message as $\mathbf{m} = (\mathbf{c}_{\mathcal{I}} + \mathbf{e}_{\mathcal{I}})G_{\mathcal{I}}^{-1}$, where $\mathbf{c}_{\mathcal{I}}$ is the part of the ciphertext $\mathbf{c}$ corresponding to the information set $\mathcal{I}$.

However, ISD, in its essence, is a technique for finding low weight codewords. Thus, if for a given code $\mathcal{C}$ with generator matrix G, it is known that it contains a subcode of low support, we can find that support, by repeatedly running ISD for codewords of weight smaller than the support (usually half or third the size of the support).

This approach has been used to attack a McEliece variant based on convolutional codes [20] and against the KKS scheme in [21]. It also applies to the schemes in [6], as discovered in [5], [9]. We briefly describe how the attack can be mounted against [6].

Suppose the private key of a McEliece-type scheme is given by a $k \times n$ generator matrix G of the form (1). Then the public key is given by a matrix $G_{\text{pub}} = SGP$, where S is an invertible $n \times n$ matrix, and P is a blockwise permutation matrix defined in [6, Alg.2]. As described in [6], the security of the scheme relies on secrecy of the permutation P, and a

partial recovery of P reveals a part of the staircase structure of G. The main observation in order to mount a successful attack is the following:

Let $\mathbf{x} \in \mathbb{F}_2^k$. Then $\mathbf{x}G_{\text{pub}} = (\mathbf{x}S)GP = \mathbf{x}'GP = \mathbf{y}'P$. From here, if $\mathbf{y}'$ is a codeword of the code with generator matrix G, then $\mathbf{y}'P$ is a codeword of the code with generator matrix G_{pub}. Since P is a permutation matrix, the Hamming weight of $\mathbf{y}'$ and $\mathbf{y}'P$ is the same, and if $\mathbf{y}'$ is a low weight codeword, so will $\mathbf{y}'P$ be.

Now, because of the structure of G, we know that there is a subspace of the code of dimension k_w whose support is of size k_w+n_w, and thus, finding enough low weight codewords of weight approximately $(k_w+n_w)/3$ will most likely reveal the support of this subspace. Once the support has been found, the permutation P has been partially found, i.e. the position of the last n_w columns in G can be determined. Continuing with the same procedure, for the n_{w-1} second to last columns, reveals even more of the structure, and so on, until the entire G is revealed.

The efficiency of this attack depends on the efficiency of the ISD algorithms, whose complexity directly depends on the weight t of the codeword being searched for, and the size of the code.

In a very recent analysis by Moody and Perlner [9] a modification of Stern's algorithm was provided, dedicated to cryptanalysis of the scheme in [6]. We refer the reader to [9] for details, and here, we mention that the complexity of the attack is in general given by $ISD_{St} = Pr_{St}^{-1} \cdot Cost_{St}$ where Pr_{St}^{-1} is the probability of success, and $Cost_{St}$ the cost of finding the low weight codeword.

We note that, an attacker can choose instead of G_{pub}, to work with the parity check matrix H_{pub} and to apply exactly the same technique for finding a low weight codeword of the dual code with generator matrix H_{pub}. In particular, if $k_1 < n_w$ then it is (roughly) more efficient to work with the parity-check matrix. Thus, for the schemes proposed in [6], the proposed encryption scheme can be more efficiently attacked through the parity-check matrix, and the signature scheme through the generator matrix.

B. Split technique as a measure against ISD attacks

One way of inoculating the scheme from [6] against ISD attacks is to tweak the various parameters, so that the attack is unfeasible. This, in general requires parameters that are over any border of practicality of the scheme. Here, we take a conceptually different approach, in the sense that we want to make the attack very improbable to mount. In other words, our strategy makes the probability of the attacker obtaining conditions under which an ISD attack can be mounted, close to zero, i.e. negligible.

Recall that the public key of our new scheme is given by $G_{\text{pub}}^1, \ldots, G_{\text{pub}}^s$, where $G_{\text{pub}}^i = SG_iP_i$, for $i \in \{1, 2, \ldots, s\}$ and $G_s = G + G_1 + \cdots + G_{s-1}$. Each P_i can be written as $P_i = PP_i^*$, where P_i^* permutes only within the blocks of length ℓ but not globally.

To make things more clear, for the moment, let us focus on the j-th submatrix block of dimension $k \times \ell$ in the matrices G_{pub}^i. We will denote these submatrices by $G_{\text{pub},j}^i$. Similarly, we will denote the submatrices of the j-th block of ℓ columns of SG_iP by $G_{\text{p},j}^i$. Let $\sigma_j^i(G_{\text{p},j}^i)$ be the matrix obtained from $G_{\text{p},j}^i$ by permuting the columns according to the permutation σ_j^i generated in Alg.2. Then $\sigma_j^i(G_{\text{p},j}^i) = G_{\text{pub},j}^i$.

It is not hard to see, that in order to be able to run an ISD attack, one needs, for

sufficient number b of submatrix blocks $j \in \{1, \ldots, n/\ell\}$, to find permutations μ_j^i, such that all $\mu_j^i \sigma_j^i$, $i \in \{1, \ldots, s\}$ agree on one or more coordinates. Indeed, in this case, in the sum $\sum_i \mu_j^i(G_{\text{pub},j}^i) = \sum_i \mu_j^i \sigma_j^i (G_{\text{p},j}^i)$ the columns on which $\mu_j^i \sigma_j^i$ agree, yield a case the same as without splitting of the public key, i.e. can be used in an ISD attack.

We will calculate first the probability that for a fixed j, all $\mu_j^i \sigma_j^i$ agree on t positions. First, the probability that $\mu_j^1 \sigma_j^1$ and $\mu_j^2 \sigma_j^2$ agree on any t positions is $\frac{\binom{\ell}{t} \cdot \lfloor (\ell-t)!/e + 1/2 \rfloor}{\ell!}$, which notably is the same as the probability that a random ℓ permutation has t fixed points. Now, the probability that any other $\mu_j^i \sigma_j^i$ agrees on the same t coordinates is $\frac{(\ell-t)!}{\ell!}$. In total all $\mu_j^i \sigma_j^i$ agree on t coordinates with probability $Pr_t = \frac{\binom{\ell}{t} \cdot \lfloor (\ell-t)!/e + 1/2 \rfloor}{\ell!} \cdot (\frac{(\ell-t)!}{\ell!})^{s-2}$.

Now, if we randomly select k/t blocks $j \in I_{k/t}$, the structure of G will be revealed in the sums $\sum_i G_{\text{pub},j}^i T_j^i$ for some randomly selected $\ell \times \ell$ permutation matrices T_j^i, with probability

$$Pr(k,t) = (Pr_t)^{\frac{k}{t}} = \left(\frac{\binom{\ell}{t} \left\lfloor \frac{(\ell-t)!}{e} + \frac{1}{2} \right\rfloor}{\ell!} \cdot (\frac{(\ell-t)!}{\ell!})^{s-2} \right)^{\frac{k}{t}} \tag{3}$$

For simplicity, we have only considered the case when from each submatrix block, we are looking for t agreements in each of the blocks. The case with different ts is unnecessary complex, and does not add any substantial difference, hence we omit it.

Note that we have calculated the probability of revealing the structure of G in k columns and not more, and certainly an ISD attack can not be mounted with only k columns. This shows that, since the calculated probability is negligible in k, and can be made arbitrarily small for practical values of k, it is infeasible to create sufficient conditions to mount an ISD attack. In other words, the splitting technique immunizes from ISD attacks.

C. Modelling the decoding problem as a Polynomial System Solving problem

Given a public key $(G_{\text{pub}}^1, \ldots, G_{\text{pub}}^s)$ and a ciphertext $\mathbf{c} = (\mathbf{c}_1, \ldots, \mathbf{c}_s)$ produced using this public key, we can model the problem of decoding $\mathbf{c}$ as a Polynomial System Solving (PoSSo) problem. PoSSo is the problem of finding a solution to a system of polynomial equations of degree $\geqslant 2$ over a finite field.

From the encryption algorithm, we have that:

$$\mathbf{e}_i = \mathbf{m} G_{\text{pub}}^i + \mathbf{c}_i, i \in \{1, \ldots, s\} \tag{4}$$

where the set of all $\mathbf{e}_i$ is a valid error split for the error set E_ℓ, characterized by a defining polynomial p_d. We can represent the error vector in the form $\mathbf{e}_i = \mathbf{e}_i^1 \parallel \mathbf{e}_i^2 \parallel \ldots \parallel \mathbf{e}_i^m \in (\mathbb{F}_2^\ell)^m$. In order for the decryption to be possible, the following has to hold:

$$p_d(\sum_{i=1}^{s} \sigma_i^j(\mathbf{e}_i^j)) = 0, \text{ for every } j \in \{1, \ldots, m\} \tag{5}$$

and for arbitrary permutations $\sigma_i^j \in \mathcal{S}_\ell$.

Considering (4) as a change of variables, and plugging it in the system (5), we obtain a system of equations in k variables - the unknown $\mathbf{m}$. Now, the obtained system can be solved using some generic system solving algorithm, like F_4 [22], F_5 [23], or XL [24]. For example, using the F_5 algorithm, the complexity of solving a semiregular system of k variables is

$$\mathcal{O}\left(\binom{k+Dreg_k-1}{Dreg_k}^{\omega}\right),$$

where ω is the linear algebra constant, and $Dreg_k$ is the maximum degree reached during the Gröbner basis computation.

It is possible to introduce an optimization parameter in the form of a guess of some of the errors, or a guess of a linear equation valid for the error vectors with some probability. Suppose the linear relation

$$L(\mathbf{e}_1^j, \ldots, \mathbf{e}_s^j) = 0 \tag{6}$$

holds with probability p_L. Then, if we make a correct guess for some j, we will reduce the number of variables in the system by 1. In general, for p guesses, the complexity of the overall process will be:

$$\left(\frac{1}{p_L}\right)^p \left(\binom{k-p+Dreg_{k-p}-1}{Dreg_{k-p}}^{\omega}\right) \tag{7}$$

How efficient this method is, strongly depends on the parameters of the scheme ℓ, s and the defining polynomial p_d. Therefore, in the next Subsection IV-D, we will analyze the efficiency of this approach for a concrete set of parameters.

D. Concrete parameter sets and their security

We will consider the following parameter sets for practical use:

Parameter Set 1. $l = 3$, $s = 2$, and $E_3 = \{\{0,0,1\}, \{0,1,0\}, \{1,0,0\}, \{0,1,1\}, \{1,0,1\}, \{1,1,0\}\}$. The defining polynomial for E_3 is $p_d = 1 + x_1 + x_2 + x_3 + x_1x_2 + x_1x_3 + x_2x_3$, and the density $\rho_3 = |E_3|^{1/\ell} = 6^{1/3}$. According to Prop. 1, and making a similar analysis as in [6], in order for the decryption process to be efficient, we need to keep the ratio $n_i/k_i \approx 6$, which implies $n \approx 7k$, and the size of the list in the end will be $\rho^n 2^{k-n} \approx 1$.

Parameter Set 2. $l = 4$, $s = 2$, and $E_4 = \{\{0,0,0,1\}, \{0,0,1,0\}, \{0,1,0,0\}, \{1,0,0,0\}, \{0,0,1,1\}, \{0,1,1,0\}, \{0,1,0,1\}, \{1,0,0,1\}, \{1,0,1,0\}, \{1,1,0,0\}\}$. The defining polynomial for E_4 is $p_d = 1 + x_1 + x_2 + x_3 + x_4 + x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + x_3x_4$. Here, the density is $\rho_4 = |E_4|^{1/\ell} = 10^{1/4}$, and efficiency requires $n_i/k_i \approx 5$, i.e. $n \approx 6k$. The list in the end is ≈ 1.

For the parameter sets 1 and 2, the system of equations (5) turns into $p_d(\sigma_1^j(\mathbf{e}_1^j) + \sigma_2^j(\mathbf{e}_2^j)) = 0, j \in \{1, \ldots, m\}$. It is easy to see that we can actually take σ_2^j to be the identity permutation, so

$$p_d(\sigma_1^j(\mathbf{e}_1^j) + \mathbf{e}_2^j) = 0, j \in \{1, \ldots, m\} \tag{8}$$

where σ_1^j is an arbitrary permutation from $\mathcal{S}_\ell$.

For the Par. Set 1, since $|\mathcal{S}_3| = 3! = 6$, for each $j \in \{1, \ldots, m\}$, we obtain 6 equations in the system (8), but a simple check shows that only 5 are linearly independent. So in total, the system has $5m$ equations, and according to the above computations, this is $\approx 12k$ equations.

Similarly, for the Par. Set 2, out of the total $|\mathcal{S}_4| = 4! = 24$ obtained equations, for each $j \in \{1, \ldots, m\}$, only 10 are linearly independent. In total, the system in this case has $10m$ i.e. $\approx 15k$ equations.

In the choice of the parameter sets, we have taken into account the probability of existence of linear relations of the form (6). We exhaustively investigated all such relations and

determined that for both sets, the highest probability for a linear relation is $p_L = 2/3$. This information can be used to combine the system solving with guessing with complexity (7).

We performed experiments using the F_4 algorithm [22] implemented in MAGMA [25], to test the behavior of the obtained system (8) of equations. Since, for both parameter sets, the system is well overdefined, the degree of regularity $Dreg_k$, as expected, grows slower than for systems with equal number of variables and equations. Unfortunately, due to memory constrains, we were not able to obtain an evident trend of the growth of $Dreg_k$ and at the moment, the behavior of these systems remains an open problem. Nevertheless, the results of our experiments strongly indicate that for practical values of k, $Dreg_k$ should grow more than 5, and even for these very conservative projections, we can obtain security of 80 or 128 bits against the described algebraic attack.

Regarding the ISD attacks, for the two parameter sets, we have that:

- for Par. Set 1, the highest probability (3) is obtained for $t = 2$, and is $Pr(k, 2) = 2^{-0.5k}$,
- for Par. Set 2, the highest probability (3) is obtained for $t = 3$, and is: $Pr(k, 3) = 2^{-0.86k}$.

Hence for any of the two parameter sets, $k > 256$ is enough for security of 128 bits against ISD attacks.

At the end of this section we provide four concrete codes from the two parameter sets, two of each set, providing security of 80 and 128 bits, respectively. We denote by $K = (k_1, \ldots, k_w)$ and $N = (n_1, \ldots, n_w)$ the vectors of values used in the definition of concrete generator matrices as defined in equation (1). The following are concrete codes from Parameter Set 1:

◦ Code $(5514, 762)$: $w = 250$, $K = (15, 3, 3, \ldots, 3)$, $N = (21, 21, 18, 18, 21, 18, 18 \ldots, 21, 18, 18)$.

◦ Code $(7956, 1095)$: $w = 361$, $K = (15, 3, 3, \ldots, 3)$, $N = (21, 21, 18, 18, 21, 18, 18 \ldots, 21, 18, 18)$.

The following are concrete codes from Parameter Set 2:

◦ Code $(4600, 776)$: $w = 191$, $K = (16, 4, 4, \ldots, 4)$, $N = (24, 20, \ldots, 20, 20)$.

◦ Code $(7000, 1776)$: $w = 291$, $K = (16, 4, 4, \ldots, 4)$, $N = (24, 20, \ldots, 20, 20)$.

Note that in the suggested parameter sets, the value of k_1 is much smaller than in the scheme proposed in [6]. This is because, the splitting technique prevents the ISD attacks, and the motive for the large k_1 in [6], were exactly ISD type of attacks.

V. Conclusion

We introduced a novel idea of splitting the public generator matrix into s randomly generated matrices, and showed that the split strategy thwarts the ISD attacks that showed fatal for the encryption scheme from [6], [8]. As a result of our security analysis, we provide concrete parameters and instances with conjectured security levels in the range from 2^{80} to 2^{128}.

While the introduction of the concept of "Valid Error Split" makes possible to transform the St-Gen encryption scheme from [6], [8] into a randomly split St-Gen scheme, it remains **an open problem** whether this splitting technique can be applied for the signature scheme introduced in [6], [8], that at this point is considered broken.

VI. Acknowledgements

We would like to thank Nicolas Sendrier and Jean-Pierre Tillich for their long and fruitful discussions. We also thank Jean-Charles Faugère and Ludovic Perret for their hospitality at LIP6 and long discussions about algebraic attacks on these systems. Also, we thank Dustin Moody and Ray Perlner for sharing their initial results and for their excellent analysis of the schemes defined in [6], [8] done in [5].

References

[1] E. N. of Excellence for Cryptology (ECRYPT), "PQCrypto 2006: International Workshop on Post-Quantum Cryptography," http://postquantum.cr.yp.to, [Retrieved: Dec. 2015].

[2] NIST, Workshop on Cybersecurity in a Post-Quantum World, www.nist.gov/itl/csd/ct/post-quantum-crypto-workshop-2015.cfm, [Retrieved: Dec. 2015].

[3] ETSI, "ETSI 2nd Quantum-Safe Crypto Workshop in partnership with the IQC," www.etsi.org/news-events/events/770-etsi-crypto-workshop-2014, [Retrieved: Dec. 2015].

[4] R. J. McEliece, A Public-Key System Based on Algebraic Coding Theory. Jet Propulsion Lab, 1978, pp. 114–116, dSN Progress Report 44.

[5] D. Moody and R. Perlner, "Vulnerabilities of "McEliece in the World of Escher"," Cryptology ePrint Archive, Report 2015/966, 2015, http://eprint.iacr.org/.

[6] D. Gligoroski, S. Samardjiska, H. Jacobsen, and S. Bezzateev, "McEliece in the world of Escher," Cryptology ePrint Archive, Report 2014/360, 2014, http://eprint.iacr.org/.

[7] S. Samardjiska and D. Gligoroski, "Approaching maximum embedding efficiency on small covers using staircase-generator codes," in Information Theory (ISIT), 2015 IEEE International Symposium on, June 2015, pp. 2752–2756.

[8] D. Gligoroski, S. Samardjiska, H. Jacobsen, and S. Bezzateev, "A new code based public key encryption and signature scheme based on list decoding," Workshop on Cybersecurity in a Post-Quantum World, NIST, Gaithersburg MD, USA, 2015.

[9] N. Sendrier and J.-P. Tillich, Private communication, Oct. 2014.

[10] P. Elias, "List decoding for noisy channels, technical report 335," Research Laboratory of Electronics, MIT, 1957.

[11] J. M. Wozencraft, "List decoding. quarterly progress report," Research Laboratory of Electronics, MIT, Tech. Rep., 1958.

[12] E. Prange, "The use of information sets in decoding cyclic codes," IRE Trans. Inf. Theor., vol. 8, 1962, pp. 5–9.

[13] P.J. Lee and E.F. Brickell, "An observation on the security of McEliece's public-key cryptosystem," in Advances in Cryptology-EUROCRYPT'88. Springer, 1988, pp. 275–280.

[14] J. S. Leon, "A probabilistic algorithm for computing minimum weights of large error-correcting codes," IEEE Trans. Inf. Theor., vol. 34, no. 5, Sep. 2006, pp. 1354–1359.

[15] J. Stern, "A method for finding codewords of small weight," in Proceedings of the 3rd International Colloquium on Coding Theory and Applications, Springer-Verlag, 1989, pp. 106–113.

[16] M. Finiasz and N. Sendrier, "Security bounds for the design of code-based cryptosystems," in Advances in Cryptology, ASIACRYPT '09, Springer-Verlag, 2009, pp. 88–105.

[17] D. J. Bernstein, T. Lange, and C. Peters, "Smaller decoding exponents: ball-collision decoding," in Advances in cryptology, CRYPTO'11. Springer-Verlag, 2011, pp. 743–760.

[18] A. May, A. Meurer, and E. Thomae, "Decoding random linear codes in Õ(20.054n)," in Advances in cryptology, ASIACRYPT'11. Springer-Verlag, 2011, pp. 107–124.

[19] A. Becker, A. Joux, A. May, and A. Meurer, "Decoding random binary linear codes in $2^{n/20}$: how 1 + 1 = 0 improves information set decoding," in Advances in cryptology, EUROCRYPT'12. Springer-Verlag, 2012, pp. 520–536.

[20] G. Landais and J.-P. Tillich, “An efficient attack of a McEliece cryptosystem variant based on convolutional codes.” Post-Quantum Cryptography, 2013, pp. 102–117.

[21] A. Otmani and J.-P. Tillich, “An efficient attack on all concrete kks proposals.” in PQCrypto, LNCS, vol. 7071. Springer, 2011, pp. 98–116.

[22] J.-C. Faugère, “A new efficient algorithm for computing GrÃűbner bases (F4).” Journal of Pure and Applied Algebra, vol. 139, no. 1–3, June 1999, pp. 61–88.

[23] J.-C. Faugère, “A new efficient algorithm for computing Gröbner bases without reduction to zero (F5),” in ISSAC 2002. ACM Press, 2002, pp. 75–83.

[24] N. Courtois, E. Klimov, J. Patarin, and A. Shamir, “Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations,” in In Advances in Cryptology, Eurocrypt’00, LNCS 1807. Springer-Verlag, 2000, pp. 392–407.

[25] MAGMA, “High performance software for algebra, number theory, and geometry — a large commercial software package.” [Online]. Available: http://magma.maths.usyd.edu.au