Quantum Resistant Random Linear Code Based Public Key Encryption Scheme RLCE

Yongge Wang
Department of SIS, UNC Charlotte, USA.
yongge.wang@uncc.edu

April 14, 2016

Abstract

Lattice based encryption schemes and linear code based encryption schemes have received extensive attention in recent years since they have been considered as post-quantum candidate encryption schemes. Though LLL reduction algorithm has been one of the major cryptanalysis techniques for lattice based cryptographic systems, key recovery cryptanalysis techniques for linear code based cryptographic systems are generally scheme specific. In recent years, several important techniques such as Sidelnikov-Shestakov attack, filtration attacks, and algebraic attacks have been developed to crypt-analyze linear code based encryption schemes. Though most of these cryptanalysis techniques are relatively new, they prove to be very powerful and many systems have been broken using them. Thus it is important to design linear code based cryptographic systems that are immune against these attacks. This paper proposes linear code based encryption scheme RLCE which shares many characteristics with random linear codes. Our analysis shows that the scheme RLCE is secure against existing attacks and we hope that the security of the RLCE scheme is equivalent to the hardness of decoding random linear codes. Example parameters for different security levels are recommended for the scheme RLCE.

Key words: Random linear codes; McEliece Encryption scheme; secure public key encryption scheme; linear code based encryption scheme

MSC 2010 Codes: 94B05; 94A60; 11T71; 68P25

1 Introduction

With rapid development for quantum computing techniques, our society is concerned with the security of current Public Key Infrastructures (PKI) which are fundamental for Internet services. The core components for current PKI infrastructures are based on public cryptographic techniques such as RSA and DSA. However, it has been shown that these public key cryptographic techniques could be broken by quantum computers. Thus it is urgent to develop public key cryptographic systems that are secure against quantum computing.

Since McEliece encryption scheme [24] was introduced more than thirty years ago, it has withstood many attacks and still remains unbroken for general cases. It has been considered as one of the candidates for post-quantum cryptography since it is immune to existing quantum computer algorithm attacks. The original McEliece cryptographic system is based on binary Goppa codes. Several variants have been introduced to replace Goppa codes in the McEliece encryption scheme. For instance, Niederreiter [27] proposed the use of generalized Reed-Solomon codes and later, Berger and Loidreau [5] proposed the use of sub-codes of generalized Reed-Solomon codes. Sidelnikov [32] proposed the use of Reed-Muller codes, Janwa and Moreno [17] proposed the use of algebraic geometry codes, Baldi et al [1] proposed the use of LDPC codes, Misoczki et al [26] proposed the use of MDPC codes, Löndahl and Johansson [20] proposed the use of convolutional codes, and Berger et al [4] and Misoczki-Barreto [25] proposed quasi-cyclic and quasi-dyadic structure based compact variants of McEliece encryption schemes. Most of them have been broken though MDPC/LDPC code based McEliece encryption scheme [1, 26] and the original binary Goppa code based McEliece encryption scheme are still considered secure.

Goppa code based McEliece encryption scheme is hard to attack since Goppa codes share many characteristics with random codes. Motivated by Faugere et al's [15] algebraic attacks against quasi-cyclic and quasi-dyadic structure based compact variants of McEliece encryption schemes, Faugere et al [14] designed an efficient algorithm to distinguish a random code from a high rate Goppa code. Márquez-Corbella and Pellikaan [21] simplified the distinguisher in [14] using Schur component-wise product of codes.

Sidelnikov and Shestakov [31] showed, for the generalized Reed-Solomon code based McEliece encryption scheme, one can efficiently recover the private parameters for the generalized Reed-Solomon code from the public key. Using component-wise product of codes and techniques from [31], Wieschebrink [37] showed that Berger and Loidreau's proposal [5] could be broken efficiently also. Couvreur et al [9] proposed a general distinguisher based filtration technique to recover keys for generalized Reed-Solomon code based McEliece scheme and Couvreur, Márquez-Corbella, and Pellikaan [10] used filtration attacks to break Janwa and Moreno's [17] algebraic geometry code based McEliece encryption scheme. The filtration attack was recently used by Couvreur et al [11] and Faugere et al [16] to attack Bernstein et al's [6] wild Goppa code based McEliece scheme.

General Goppa code based McEliece schemes are still immune from these attacks. However, based on the new development of cryptanalysis techniques against linear code based cryptographic systems in the recent years, it is important to systematically design random linear code based cryptographic systems defeating these attacks. Motivated by this observation, this paper presents a systematic approach of designing public key encryption schemes using any linear code. For example, we can even use Reed-Solomon codes to design McEliece encryption scheme while it is insecure to use Reed-Solomon codes in the original McEliece scheme. Since our design of linear code based encryption scheme embeds randomness in each column of the generator matrix, it is expected that, without the corresponding private key, these codes are as hard as random linear codes for decoding.

The most powerful message recovery attacks (not key recovery attacks) on McEliece cryptosystem is the information-set decoding attack which was introduced by Prange [29]. In an information-set decoding approach, one finds a set of coordinates of a received ciphertext which are error-free and that the restriction of the code's generator matrix to these positions is invertible. The original message can then be computed by multiplying the ciphertext with the inverse of the sub-matrix. Improvements of the information-set decoding attack have been proposed by Lee-Brickell [18], Leon [19], Stern [33], May-Meurer-Thomae [22], Becker-Joux-May-Meurer [3], and May-Ozerov [23]. Bernstein, Lange, and Peters [7] presented an exact complexity analysis on information-set decoding attack against McEliece cryptosystem. The attacks in [3, 7, 18, 19, 22, 23, 33] are against binary linear codes and are not applicable when the underlying field is $GF(p^m)$ for a prime p. Peters [28] presented an exact complexity analysis on information-set decoding attack against McEliece cryptosystem over $GF(p^m)$. These information-set decoding techniques (in particular, the exact complexity analysis in [7, 28]) are used to select example parameters for RLCE scheme in Section 5.

Unless specified otherwise, we will use $q = 2^m$ or $q = p^m$ for a prime p and our discussion are based on the field $GF(q)$ through out this paper. Bold face letters such as $\mathbf{a}, \mathbf{b}, \mathbf{e}, \mathbf{f}, \mathbf{g}$ are used to denote row or column vectors over $GF(q)$. It should be clear from the context whether a specific bold face letter represents a row vector or a column vector.

2 Goppa codes and McEliece Public Key Encryption scheme

In this section, we briefly review Goppa codes and McEliece scheme. For given parameters $q, n \le q$, and t, let $g(x)$ be a polynomial of degree t over $GF(q)$. Assume that $g(x)$ has no multiple zero roots and $\alpha_0, \cdots, \alpha_{n-1} \in GF(q)$ be pairwise distinct which are not roots of $g(x)$. The following subspace $C_{Goppa}(g)$ defines the code words of an $[n, k, d]$ binary Goppa code where $d \ge 2t+1$. This binary Goppa code $C_{Goppa}(g)$ has dimension $k \ge n - tm$ and corrects t errors.

$$C_{Goppa}(g) = \left\{ c \in \{0,1\}^n : \sum_{i=0}^{n-1} \frac{c_i}{x - \alpha_i} \equiv 0 \bmod g(x) \right\}.$$

Furthermore, if $g(x)$ is irreducible, then $C_{Goppa}(g)$ is called an irreducible Goppa code. The parity check matrix H for the Goppa codes looks as follows:

$$V_t(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_0^1 & \alpha_1^1 & \cdots & \alpha_{n-1}^1 \\ \cdots & \cdots & \ddots & \cdots \\ \alpha_0^t & \alpha_1^t & \cdots & \alpha_{n-1}^t \end{bmatrix} \begin{bmatrix} \frac{1}{g(\alpha_0)} & & \\ & \ddots & \\ & & \frac{1}{g(\alpha_{n-1})} \end{bmatrix} \tag{1}$$

where $\mathbf{x} = [\alpha_0, \ldots, \alpha_{n-1}]$ and $\mathbf{y} = \left[\frac{1}{g(\alpha_0)}, \ldots, \frac{1}{g(\alpha_{n-1})}\right]$.

The McEliece scheme [24] is described as follows. For the given parameters n and t, choose a binary Goppa code based on an irreducible polynomial $g(x)$ of degree t. Let G_s be the $k \times n$ generator matrix for the Goppa code. Select a random dense $k \times k$ nonsingular matrix S and a random $n \times n$ permutation matrix P. Note that the permutation matrix P is required only if the support $\alpha_0, \cdots, \alpha_{n-1}$ is known to the public. Then the public key is $G = SG_sP$ which generates a linear code with the same rate and minimum distance as the code generated by G_s. The private key is G_s.

Encryption. For a k-bit message block $\mathbf{m}$, choose a random row vector $\mathbf{e} \in \{0,1\}^n$ of length n and weight t. Compute the cipher text $\mathbf{y} = \mathbf{m}G + \mathbf{e}$

Decryption. For a received ciphertext $\mathbf{y}$, first compute $\mathbf{y}' = \mathbf{y}P^{-1}$. Next use an error-correction algorithm to recover $\mathbf{m}' = \mathbf{m}S$ and compute the message $\mathbf{m}$ as $\mathbf{m} = \mathbf{m}'S^{-1}$.

3 Random linear code based encryption scheme RLCE

The protocol for the Random Linear Code based Encryption scheme RLCE proceeds as follows:

Key Setup. Let n, $k, d, t > 0$, and $r \ge 1$ be given parameters such that $n - k + 1 \ge d \ge 2t + 1$. Let $G_s = [\mathbf{g}_0, \cdots, \mathbf{g}_{n-1}]$ be a $k \times n$ generator matrix for an $[n, k, d]$ linear code such that there is an efficient decoding algorithm to correct at least t errors for this linear code given by G_s.

1. Let $C_0, C_1, \cdots, C_{n-1} \in GF(q)^{k\times r}$ be $k \times r$ matrices drawn uniformly at random and let

   $$G_1 = [\mathbf{g}_0, C_0, \mathbf{g}_1, C_1 \cdots, \mathbf{g}_{n-1}, C_{n-1}] \tag{2}$$

   be the $k \times n(r+1)$ matrix obtained by inserting the random matrices C_i into G_s.

2. Let $A_0, \cdots, A_{n-1} \in GF(q)^{(r+1)\times(r+1)}$ be dense nonsingular $(r+1) \times (r+1)$ matrices chosen uniformly at random and let

   $$A = \begin{bmatrix} A_0 & & & \\ & A_1 & & \\ & & \ddots & \\ & & & A_{n-1} \end{bmatrix} \tag{3}$$

be an $n(r+1)\times n(r+1)$ nonsingular matrix.

3. Let S be a random dense $k\times k$ nonsingular matrix and P be an $n(r+1)\times n(r+1)$ permutation matrix.

4. The public key is the $k\times n(r+1)$ matrix $G = SG_1AP$ and the private key is (S, G_s, P, A).

Encryption. For a row vector message $\mathbf{m}\in GF(q)^k$, choose a random row vector $\mathbf{e} = [e_0, \ldots, e_{n(r+1)-1}] \in GF(q)^{n(r+1)}$ such that the Hamming weight of $\mathbf{e}$ is at most t. The cipher text is $\mathbf{y} = \mathbf{m}G + \mathbf{e}$.
Decryption. For a received cipher text $\mathbf{y} = [y_0, \ldots, y_{n(r+1)-1}]$, compute

$$\mathbf{y}P^{-1}A^{-1} = [y'_0, \ldots, y'_{n(r+1)-1}] = \mathbf{m}SG_1 + \mathbf{e}P^{-1}A^{-1}$$

where

$$A^{-1} = \left[\begin{array}{cccc} A_0^{-1} & & & \\ & A_1^{-1} & & \\ & & \ddots & \\ & & & A_{n-1}^{-1} \end{array}\right] \tag{4}$$

Let $\mathbf{y}' = [y'_0, y'_{r+1}, \cdots, y'_{(n-1)(r+1)}]$ be the row vector of length n selected from the length $n(r+1)$ row vector $\mathbf{y}P^{-1}A^{-1}$. Then $\mathbf{y}' = \mathbf{m}SG_s + \mathbf{e}'$ for some error vector $\mathbf{e}' \in GF(q)^n$. Let $\mathbf{e}'' = \mathbf{e}P^{-1} = [e''_0, \cdots, e''_{n(r+1)-1}]$ and $\mathbf{e}''_i = [e''_{i(r+1)}, \ldots, e''_{i(r+1)+r}]$ be a sub-vector of $\mathbf{e}''$ for $i \le n-1$. Then $\mathbf{e}'[i]$ is the first element of $\mathbf{e}''_i A_i^{-1}$. Thus $\mathbf{e}'[i] \neq 0$ only if $\mathbf{e}''_i$ is non-zero. Since there are at most t non-zero sub-vectors $\mathbf{e}''_i$, the Hamming weight of $\mathbf{e}' \in GF(q)^n$ is at most t. Using the efficient decoding algorithm, one can compute $\mathbf{m}' = \mathbf{m}S$ and $\mathbf{m} = \mathbf{m}'S^{-1}$. Finally, calculate the Hamming weight $w = \mathtt{weight}(\mathbf{y} - \mathbf{m}G)$. If $w \le t$ then output $\mathbf{m}$ as the decrypted plaintext. Otherwise, output error.
Comment 1. In the design of RLCE scheme, the permutation matrix P has two purposes. The first purpose is to hide the supports of the underlying encoding scheme generator matrix (this is necessary if the supports of the underlying encoding scheme are unknown). The second purpose is to hide the positions and combinations of the column vectors $\mathbf{g}_i$ and C_i.
Comment 2. In the RLCE decryption process, one checks whether the Hamming weight $w = \mathtt{weight}(\mathbf{y} - \mathbf{m}G)$ is smaller than t. This step is used to defeat chosen ciphertext attacks (CCA). In a CCA atack, an adversary gives a random vector $\mathbf{y} = [y_0, \ldots, y_{n(r+1)-1}]$ (which is not a valid ciphertext) to the decryption oracle to learn a decrypted value. This decrypted value could be used to obtain certain information about the private generator matrix G_s (see Section 4.2 for details). Alternatively, one may use an appropriate padding scheme to pad a message before encryption. Then it is sufficient for the decryption process to verify whether the decrypted message has the correct padding strings to defeat the CCA attacks.

4 Robustness of RLCE codes against existing attacks

4.1 Randomness of generator matrix columns

We first use the following theorem to show that any single column of the underlying generator matrix G_s could be completely randomized in a RLCE public key G.

Theorem 4.1 *Let $G_s = [\mathbf{g}_0, \cdots, \mathbf{g}_{n-1}] \in GF(q)^{k\times(n-1)}$ be a linear code generator matrix. For any randomly chosen full rank $k\times(r+1)$ matrix $R_0 \in GF(q)^{k\times(r+1)}$, there exists a $k\times k$ nonsingular matrix S, a $(r+1)\times(r+1)$ matrix A_0, and a $k\times r$ matrix $C_0 \in GF(q)^{k\times r}$ such that*

$$R_0 = S[\mathbf{g}_0, C_0]A_0 \tag{5}$$

Proof. By the fundamental properties of matrix equivalence, for two $m \times n$ matrices A, B of the same rank, there exist invertible $m \times m$ matrix P and $n \times n$ invertible matrix Q such that $A = PBQ$. The theorem could be proved using this property and the details are omitted here. $\square$

Let $R = [R_0, \ldots, R_{n-1}] \in GF(q)^{k \times n(r+1)}$ be a fixed random linear code generator matrix. Theorem 4.1 shows that for any generator matrix G_s (e.g., a Reed-Solomon code generator matrix), we can choose matrices S and A_0 so that the first $r+1$ columns of the RLCE scheme public key G (constructed from G_s) are identical to R_0. However, we cannot use Theorem 4.1 to continue the process of choosing $A_1, \ldots, A_{n-1}$ to obtain $G = R$ since S is fixed after A_0 is chosen. Indeed, it is straightforward to show that one can use Theorem 4.1 to continue the process of choosing $A_1, \ldots, A_{n-1}$ to obtain $G = R$ if and only if there exists a $k \times k$ nonsingular matrix S such that, for each $i \leq n-1$, the vector $S\mathbf{g}_i$ lies in the linear space generated by the column vectors of R_i. A corollary of this observation is that if R_i generates the full k dimensional space, then each linear code could have any random matrix as its RLCE public key.

Theorem 4.2 *Let $R = [R_0, \ldots, R_{n-1}] \in GF(q)^{k \times n(r+1)}$ and $G_s = [\mathbf{g}_0, \cdots, \mathbf{g}_{n-1}] \in GF(q)^{k \times n}$ be two fixed MDS linear code generator matrices. If $r+1 \geq k$, then there exist $A_0, \cdots, A_{n-1} \in GF(q)^{(r+1)\times(r+1)}$ and $C_0, \cdots, C_{n-1} \in GF(q)^{k \times r}$ such that $R = [\mathbf{g}_0, C_0, \cdots, \mathbf{g}_{n-1}, C_{n-1}]A$ where A is in the format of (3).*

Proof. Without loss of generality, we may assume that $r = k-1$. For each $0 \leq i \leq n-1$, choose a random matrix $C_i \in GF(q)^{k \times r}$ such that $G_i = [\mathbf{g}_i, C_i]$ is an $k \times k$ invertible matrix. Let $A_i = G_i^{-1} R_i$. Then the theorem is proved. $\square$

Theorem 4.2 shows that in the RLCE scheme, we must have $r < k-1$. Otherwise, for a given public key $G \in GF(q)^{k \times n(r+1)}$, the adversary can choose a Reed-Solomon code generator matrix $G_s \in GF(q)^{k \times n}$ and compute $A_0, \cdots, A_{n-1} \in GF(q)^{r \times r}$ and $C_0, \cdots, C_{n-1} \in GF(q)^{k \times r}$ such that $G = [\mathbf{g}_0, C_0, \cdots, \mathbf{g}_{n-1}, C_{n-1}]A$. In other words, the adversary can use the decryption algorithm corresponding to the generator matrix G_s to break the RLCE scheme.

Theorem 4.2 also implies an efficient decryption algorithm for random $[n, k]$ linear codes with sufficiently small t of errors. Specifically, for an $[n, k]$ linear code with generator matrix $R \in GF(q)^{k \times n}$, if $t \leq \frac{n-k^2}{2k}$, then one can divide R into $m = 2t + k$ blocks $R = [R_0, \cdots, R_{m-1}]$. Theorem 4.2 can then be applied to construct an equivalent $[m, k]$ Reed-Solomon code with generator matrix $G_s \in GF(q)^{k \times m}$. Thus it is sufficient to decrypt the equivalent Reed-Solomon code instead of the original random linear code. For McEliece based encryption scheme, Bernstein, Lange, and Peters [7] recommends the use of 0.75 ($= k/n$) as the code rate. Thus Theorem 4.2 has no threat on these schemes.

For $t \leq \frac{n-k^2}{2k}$, the adversary is guaranteed to succeed in breaking the system. Since multiple errors might be located within the same block R_i with certain probability, for a given t that is slightly larger than $\frac{n-k^2}{2k}$, the adversary still has a good chance to break the system using the above approach. It is recommended that t is significantly larger than $\frac{n-k^2}{2k}$. For the RLCE scheme, this means that r should be significantly smaller than k. This is normally true since k is very larger for secure RLCE schemes.

In following sections, we list heuristic and experimental evidences that the RLCE public key G shares the properties of random linear codes. We believe that the security of the RLCE scheme is equivalent to decoding a random linear code.

4.2 Chosen ciphertext attacks (CCA)

In this section, we show that certain information about the private generator matrix G_s is leaked if the decryption process does neither include padding scheme validation nor include ciphertext correctness validation. However, it is not clear whether this kind of information leakage would help the adversary to break the RLCE encryption scheme. We illustrate this using the parameter $r = 1$.

Assume that $G_1 = [\mathbf{g}_0, \mathbf{r}_0, \mathbf{g}_1, \mathbf{r}_1, \cdots, \mathbf{g}_{n-1}, \mathbf{r}_{n-1}]$ and $G = SG_1AP$. The adversary chooses a random vector $\mathbf{y} = [y_0, \ldots, y_{2n-1}] \in GF(q)^{2n-1}$ and gives it to the decryption oracle which outputs a vector $\mathbf{x} \in GF(q)^k$. Let $\mathbf{y}P^{-1}A^{-1} = [y'_0, \ldots, y'_{2n-1}]$ and $A_i = \begin{bmatrix} a_{i,00} & a_{i,01} \\ a_{i,10} & a_{i,11} \end{bmatrix}$. Then we have

$$\begin{array}{rcl} \mathbf{x}G - \mathbf{y} & = & \mathbf{x}S[\mathbf{g}_0, \mathbf{r}_0, \mathbf{g}_1, \mathbf{r}_1, \cdots, \mathbf{g}_{n-1}, \mathbf{r}_{n-1}]AP - \mathbf{y} \\ & = & [\cdots, \mathbf{x}S[\mathbf{g}_i, \mathbf{r}_i]A_i, \cdots]P - \mathbf{y} \\ & = & [\cdots, [y'_{2i}, \mathbf{x}S\mathbf{r}_i]A_i, \cdots]P + \mathbf{e} - \mathbf{y} \\ & = & [\cdots, [y'_{2i}, y'_{2i+1}]A_i, \cdots]P + [\cdots, [0, \mathbf{x}S\mathbf{r}_i - y'_{2i+1}]A_i, \cdots]P + \mathbf{e} - \mathbf{y} \\ & = & \mathbf{y} + [\cdots, [0, \mathbf{x}S\mathbf{r}_i - y'_{2i+1}]A_i, \cdots]P + \mathbf{e} - \mathbf{y} \\ & = & [\cdots, [(\mathbf{x}S\mathbf{r}_i - y'_{2i+1})a_{i,10}, (\mathbf{x}S\mathbf{r}_i - y'_{2i+1})a_{i,11}], \cdots]P + \mathbf{e} \end{array} \tag{6}$$

where $\mathbf{e}$ is a row vector of Hamming weight at most t. From the identity (6), one can calculate a list of potential values for $c_i = a_{i,10}/a_{i,11}$. The size of this list is $\binom{2n}{2}$. For each value in this list, one obtains the corresponding two column vectors $[\mathbf{f}_0, \mathbf{f}_1] = S[\mathbf{g}_i, \mathbf{r}_i]A_i$ from the public key G. Then one has

$$[\mathbf{f}_0, \mathbf{f}_1]\begin{bmatrix} 1 & 0 \\ -c_i & 1 \end{bmatrix} = S[\mathbf{g}_i, \mathbf{r}_i]\begin{bmatrix} a_{i,00} & a_{i,01} \\ c_i a_{i,11} & a_{i,11} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ -c_i & 1 \end{bmatrix} = S[\mathbf{g}_i, \mathbf{r}_i]\begin{bmatrix} a_{i,00} - c_i a_{i,01} & a_{i,01} \\ 0 & a_{i,11} \end{bmatrix} \tag{7}$$

That is, $\mathbf{f}_0 - c_i\mathbf{f}_1 = (a_{i,00} - c_i a_{i,01})S\mathbf{g}_i$. Thus, for each candidate permutation matrix P, one can calculate a matrix SG_sB where $B = \mathrm{diag}[a_{0,00} - c_0 a_{0,01}, \cdots, a_{n-1,00} - c_{n-1}a_{n-1,01}]$ is an $n \times n$ diagonal matrix with unknown diagonal elements $a_{0,00} - c_0 a_{0,01}, \cdots$, and $a_{n-1,00} - c_{n-1}a_{n-1,01}$.

On the other hand, for each ciphertext $\mathbf{y} = [y_0, \ldots, y_{2n-1}] \in GF(q)^{2n-1}$, let $\mathbf{y}P^{-1} = [z_0, z_1, \cdots, z_{2n-1}]$. The codeword corresponding to the secret generator matrix SG_s is $[y'_0, y'_2, \ldots, y'_{2n-2}]$ where $\mathbf{y}P^{-1}A^{-1} = [y'_0, \ldots, y'_{2n-1}]$. By the fact that

$$[y'_{2i}, y'_{2i+1}] = [z_{2i}, z_{2i+1}]A_i^{-1} = \frac{1}{|A_i|}[z_{2i}, z_{2i+1}]\begin{bmatrix} a_{i,11} & -a_{i,01} \\ -c_i a_{i,11} & a_{i,00} \end{bmatrix},$$

one has $y'_{2i} = \frac{a_{i,11}}{|A_i|}(z_{2i} - c_i z_{2i+1})$. For each candidate permutation matrix P, one first chooses k independent messages $\mathbf{x}_0, \cdots, \mathbf{x}_{k-1}$ and calculates the corresponding k independent ciphertexts $\mathbf{y}_0, \cdots, \mathbf{y}_{k-1}$. Using P and the above mentioned technique, one obtains a generator matrix $G_a = S'G_s\mathrm{diag}\left[\frac{a_{0,11}}{|A_0|}, \cdots, \frac{a_{n-1,11}}{|A_{n-1}|}\right]$. Thus in order to decode a ciphertext $\mathbf{y}$, it is sufficient to decode the error correcting code given by the generator matrix G_a. This task becomes feasible for certain codes. For example, this task is equivalent to the problem of attacking a generalized Reed-Solomon code based McElience encryption scheme if G_s generates a generalized Reed-Solomon code.

In order for the attacks in the preceding paragraphs to work, the adversary needs to have the knowledge of the permutation matrix P. Since the number of candidate permutation matrices P is huge, this kind of attacks is still infeasible in practice.

4.3 Niederreiter's scheme and Sidelnikov-Shestakov's attack

Sidelnikov and Shestakov's cryptanalysis technique [31] was used to analyze Niederreiter's scheme which is based on generalized Reed-Solomon codes. Let $\alpha = (\alpha_0, \ldots, \alpha_{n-1})$ be n distinct elements of $GF(q)$ and let $v = (v_0, \ldots, v_{n-1})$ be nonzero (not necessarily distinct) elements of $GF(q)$. The generalized Reed-Solomon (GRS) code of dimension k, denoted by $GRS_k(\alpha, v)$, is defined by the following subspace.

$$GRS_k(\alpha, v) = \{(v_0 f(\alpha_0), \ldots, v_{n-1} f(\alpha_{n-1})) : f(x) \in GF(q)[x]_k)\}$$

where $GF(q)[x]_k$ is the set of polynomials in $GF(q)[x]$ of degree less than k. $GF(q)[x]_k$ is a vector space of dimension k over $GF(q)$. For each code word $c = (v_0 f(\alpha_0), \ldots, v_{n-1} f(\alpha_{n-1}))$, $f(x) = f_0 + f_1 x + \ldots + f_{k-1} x^{k-1}$ is called the associate polynomial of the code word c that encodes the message $(f_0, \ldots, f_{k-1})$. $GRS_k(\alpha, v)$ is an $[n, k, d]$ MDS code where $d = n - k + 1$.

Niederreiter's scheme [27] replaces the binary Goppa codes in McEliece scheme using GRS codes. The first attack on Niederreiter scheme is presented by Sidelnikov and Shestakov [31]. Wieschebrink [36] revised Niederreiter's scheme by inserting random column vectors into random positions of G_s before obtaining the public key G. Couvreur et al [9] showed that Wieschebrink's revised scheme is insecure under the product code attacks.

Berger and Loidreau [5] recommend the use of sub codes of Niederreiter's scheme to avoid Sidelnikov and Shestakov's attack. Specifically, in Berger and Loidreau's scheme, one uses a random $(k-l) \times k$ matrix S' of rank $k - l$ instead of the $k \times k$ matrix S to compute the public key $G = S' G_s$.

For smaller values of l, Wieschebrink [37] shows that a private key (α, v) for Berger and Loidreau scheme [5] could be recovered using Sidelnikov-Shestakov algorithm. For larger values of l, Wieschebrink used Schur product code to recover the secret values for Berger-Loidreau scheme. Let $G = SG_s$ be the $(k-l) \times n$ public key generator matrix for Berger-Loidreau scheme, $\mathbf{r}_0, \cdots, \mathbf{r}_{k-l-1}$ be the rows of G, and $f_0, \cdots, f_{k-l-1}$ be the associated polynomials to those rows. For two row vectors $\mathbf{a}, \mathbf{b} \in GF(q)^n$, the component wise product $\mathbf{a} * \mathbf{b} \in GF(q)^n$ is defined as

$$\mathbf{a} * \mathbf{b} = (a_0 b_0, \cdots, a_{n-1} b_{n-1}) \tag{8}$$

By the definition in (8), it is straightforward to observe that

$$\mathbf{r}_i * \mathbf{r}_j = (v_0^2 f_i(\alpha_0) f_j(\alpha_0), \cdots, v_{n-1}^2 f_i(\alpha_{n-1}) f_j(\alpha_{n-1})). \tag{9}$$

For $2k - 1 \leq n - 2$, if the code generated by $\mathbf{r}_i * \mathbf{r}_j$ equals $GRS_{2k-1}(\alpha, \mathbf{v}')$ for $\mathbf{v}' = (v_0^2, \cdots, v_{n-1}^2)$, then the Sidelnikov-Shestakov algorithm could be used to recover the values α and v. For $2k - 1 \leq n - 2$, if the code generated by $\mathbf{r}_i * \mathbf{r}_j$ does not equal $GRS_{2k-1}(n, \mathbf{v}')$, then the attack fails. Wieschebrink claimed that the probability that the attack fails is very small. For the case of $2k - 1 > n - 2$, Wieschebrink applied Sidelnikov-Shestakov algorithm on the component wise product code of a shortened code of the original $GRS_k(\alpha, \mathbf{v})$.

The crucial step in Sidelnikov and Shestakov attack is to use the echelon form $E(G) = [I|G']$ of the public key to get minimum weight codewords that are co-related to each other supports. In the encryption scheme RLCE, each column of the public key G contains mixed randomness. Thus the echelon form $E(G) = [I|G']$ obtained from the public key G could not be used to build any useful equation system. In other words, it is expected that Sidelnikov and Shestakov attack does not work against the RLCE scheme.

4.4 Filtration attacks

Using distinguisher techniques [14], Couvreur et al. [9] designed a filtration technique to attack GRS code based McEliece scheme. The filtration technique was further developed by Couvreur et al [11] to attack wild Goppa code based McEliece scheme. In the following, we briefly review the filtration attack in [11]. For two codes C_1 and C_2 of length n, the star product code $C_1 * C_2$ is the vector space spanned by $\mathbf{a} * \mathbf{b}$ for all pairs $(\mathbf{a}, \mathbf{b}) \in C_1 \times C_2$ where $\mathbf{a} * \mathbf{b}$ is defined in (8). For $C_1 = C_2$, $C_1 * C_1$ is called the square code of C_1. It is showed in [11] that

$$\dim C_1 \times C_2 \leq \left\{ n, \dim C_1 \dim C_2 - \binom{\dim(C_1 \cap C_1)}{2} \right\}. \tag{10}$$

Furthermore, the equality in (10) is attained for most randomly selected codes C_1 and C_2 of a given length and dimension. Note that for $C = C_1 = C_2$ and $\dim C = k$, the equation (10) becomes $\dim C^{*2} \leq \min\left\{n, \binom{k+1}{2}\right\}$.

Couvreur et al [11] showed that the square code of an alternant code of extension degree 2 may have an unusually low dimension when its actual rate is larger than its designed rate. Specifically, Couvreur et al created a family of nested codes (called a filtration) defined as follows:

$$C^a(0) \supseteq C^a(1) \supseteq \cdots \supseteq C^a(q+1). \tag{11}$$

where $a \in \{0, \cdots, n-1\}$. Roughly speaking, $C^a(j)$ consists of codewords of C corresponding to polynomials which have a zero of order j at position a. The first two elements of this filtration are just punctured and shortened versions of C and the rest of them can be computed from C by computing star products and solving linear systems. The support values $\alpha_0, \cdots, \alpha_{n-1}$ (the private key) for the Goppa code could be recovered using this nested family of codes efficiently.

The crucial part of the filtration technique is the efficient algorithm to compute the nested family of codes in (11). For our RLCE scheme, the public key generator matrix G contains random columns. Thus linear equations constructed in Couvreur et al [11] could not be solved and the nested family (11) could not be computed correctly. Furthermore, the important characteristics for a code C to be vulnerable is that one can find a related code C_1 of dimension k such that the dimension of the square code of C_1 has a dimension significantly less than $\min\left\{n, \binom{k+1}{2}\right\}$.

To get experimental evidence that RLCE codes share similarity with random linear codes with respect to the above mentioned filtration attacks, we carried out several experiments using Shoup's NTL library [30]. The source code for our experiments is available at [35]. In the experiments, we used Reed-Solomon codes over $GF(2^{10})$. The RLCE parameters are chosen as the 80-bit security parameter $n = 560$, $k = 380$, $t = 90$, and $r = 1$ (see Section 5 for details). For each given 380×560 generator matrix G_s of Reed-Solomon code, we selected another random 380×560 matrix $C \in GF(2^{10})^{380\times 560}$ and selected 2×2 matrices $A_0, \ldots, A_{559}$. Each column $\mathbf{c}_i$ in C is inserted in G_s after the column $\mathbf{g}_i$. The extended generator matrix is multiplied by $A = \mathrm{diag}[A_0, \ldots, A_{559}]$ from the right hand side to obtain the public key matrix $G \in GF(2^{10})^{380\times 1120}$. For each $i = 0, \cdots, 1119$, the matrix G_i is used to compute the product code, where G_i is obtained from G by deleting the ith column vector. In our experiments, all of these product codes have dimension 1119. We repeated the above experiments 100 times for 100 distinct Reed-Solomon generator matrices and the results remained the same. Since $\min\left\{1119, \binom{381}{2}\right\} = 1119$, the experimental results meet our expectation that RLCE behaves like a random linear code. We did the same experiments for the dual code of the above code. That is, for a 180×560 generator matrix G_s of the dual code, the same procedure has been taken. In this time, after deleting one column from the resulting public key matrix, the product code always had dimension 1119 which is the expected dimension for a random linear code.

4.5 Algebraic attacks

Faugere, Otmani, Perret, and Tillich [15] developed an algebraic attack against quasi-cyclic and dyadic structure based compact variants of McEliece encryption scheme. In a high level, the algebraic attack from [15] tries to find $\mathbf{x}^*, \mathbf{y}^* \in GF(q)^n$ such that $V_t(\mathbf{x}^*, \mathbf{y}^*)$ is the parity check matrix for the underlying alternant codes of the compact variants of McEliece encryption scheme. $V_t(\mathbf{x}^*, \mathbf{y}^*)$ can then be used to break the McEliece scheme. Note that this $V_t(\mathbf{x}^*, \mathbf{y}^*)$ is generally different from the original parity check matrix $V_t(\mathbf{x}, \mathbf{y})$ in (1). The parity check matrix $V_t(\mathbf{x}^*, \mathbf{y}^*)$ was obtained by solving an equation system constructed from

$$V_t(\mathbf{x}^*, \mathbf{y}^*)G^T = 0, \tag{12}$$

where G is the public key. The authors of [15] employed the special properties of quasi-cyclic and dyadic structures (which provide additional linear equations) to rewrite the equation system obtained from (12) and then calculate $V_t(\mathbf{x}^*, \mathbf{y}^*)$ efficiently.

Faugere, Gauthier-Umaña, Otmani, Perret, and Tillich [14] used the algebraic attack in [15] to design an efficient Goppa code distinguisher to distinguish a random matrix from the matrix of a Goppa code whose rate is close to 1. For instance, [14] showed that the binary Goppa code obtained with $m = 13$ and $r = 19$ corresponding to a 90-bit security key is distinguishable.

It is challenging to mount the above mentioned algebraic attacks on the RLCE encryption scheme. Assume that the RLCE scheme is based on a Reed-Solomon code. Let G be the public key and (S, G_s, A, P) be the private key. The parity check matrix for a Reed-Solomon code is in the format of

$$V_t(\alpha) = \left[\begin{array}{ccccc} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{t+1} & \alpha^{2(t+1)} & \cdots & \alpha^{(t+1)(n-1)} \end{array}\right]. \tag{13}$$

The algebraic attack in [14, 15] requires one to obtain a parity check matrix $V_t(\alpha^*)$ for the underlying Reed-Solomon code from the public key G, where α^* may be different from α. Assume that $V_t(\alpha^*) = [\mathbf{v}_0, \cdots, \mathbf{v_{n-1}}] \in GF(q)^{(t+1)\times n}$ is a parity check matrix for the underlying Reed-Solomon code. Let $V'_t(\alpha^*) \in GF(q)^{(t+1)\times n(r+1)}$ be a $(t+1) \times n(r+1)$ matrix obtained from $V_t(\alpha^*)$ by inserting r column vectors $\mathbf{0}$ after each column of $V_t(\alpha^*)$. That is,

$$V'_t(\alpha^*) = [\mathbf{v}_0, \mathbf{0}, \mathbf{v}_1, \mathbf{0}, \cdots, \mathbf{v_{n-1}}, \mathbf{0}]. \tag{14}$$

Then we have

$$\begin{array}{rcl} V'_t(\alpha^*)G_1^T & = & V'_t(\alpha^*)[\mathbf{g}_0, C_0, \cdots, \mathbf{g}_{n-1}, C_{n-1}]^T \\ & = & V_t(\alpha^*)[\mathbf{g}_0, \cdots, \mathbf{g}_{n-1}]^T \\ & = & V_t(\alpha^*)G_s^T \\ & = & \mathbf{0}. \end{array} \tag{15}$$

We cannot build an equation system for the unknown $V'_t(\alpha^*)$ from the public key $G = SG_1AP$ directly since the identity (15) only shows the relationship between $V'_t(\alpha^*)$ and G_1. In other words, in order to build an equation system for $V'_t(\alpha^*)$, one also needs to use unknown variables for the non-singular matrix A and the permutation matrix P. That is, we have

$$V'_t(\alpha^*)(A^{-1})^T(P^{-1})^TG^T = V'_t(\alpha^*)(GP^{-1}A^{-1})^T = V'_t(\alpha^*)G_1^TS^T = \mathbf{0}. \tag{16}$$

with an unknown α^*, an unknown permutation matrix P, and an unknown matrix $A = \mathrm{diag}[A_0, \cdots, A_{n-1}]$ which consists of n dense nonsingular $(r+1)\times(r+1)$ matrices $A_i \in GF(q)^{(r+1)\times(r+1)}$ as defined in (3). In order to find a solution α^*, one first needs to take a potential permutation matrix P^{-1} to reorganize columns of the public key G. Then, using the identity $V'_t(\alpha^*)(A^{-1})^T(P^{-1})^TG^T = \mathbf{0}$, one can build a degree $(t+1)(n-1)+1$ equation system of $k(t+1)$ equations in $n(r+1)^2+1$ unknowns. In case that $k(t+1) \geq n(r+1)^2+1$, one may use Buchberger's Gröbner basis algorithms as in [15] to find a solution α^*. However, this kind of algebraic attacks are infeasible due to the following two challenges. First the number of permutation matrices P is too large to be handled practically. Secondly, even if one can manage to handle the large number of permutation matrices P, the Gröbner basis (or the improved variants such as F_4 or F_5 in Faugere [13, 12]) are impractical for such kind of equation systems.

The Gröbner basis algorithm eliminates top order monomial (in a given order such as lexicographic order) by combining two equations with appropriate coefficients. This process continues until one obtains a univariate polynomial equation. The resulting univariate polynomial equation normally has a very high degree and Buchberger's algorithm runs in exponential time on average (the worst case complexity is double exponential time). Thus Buchberger's algorithm cannot solve nonlinear multicariate equation systems with more than 20 variables in practice (see, e.g., Courtois et al [8]). But it should also be noted that though the worst-case Gröbner basis algorithm is double exponential, the generic behavior is generally much better. In particular, if the algebraic system has only a finite number of common zeros at infinity, then Gröbner basis algorithm for any ordering stops in a polynomial time in d^n where $d = \max\{d_i : d_i \text{ is the total degree of } f_i\}$ and n is the number of variables (see, e.g., [2]).

5 Practical considerations

In order to reduce the message expansion ratio which is defined as the rate of ciphertext size and corresponding plaintext size, it is preferred to use a smaller r for the RLCE encryption scheme. Indeed, the experimental results show that $r = 1$ is sufficient for RLCE to behave like a random linear code. As mentioned in the introduction section, the most powerful message recovery attack (not private key recovery attack) on McEliece encryption schemes is the information-set decoding attack. For the RLCE encryption scheme, the information-set decoding attack is based on the number of columns in the public key G instead of the number of columns in the private key G_s. For the same error weight t, the probability to find error-free coordinates in $(r+1)n$ coordinates is different from the probability to find error-free coordinates in n coordinates. Specifically, the cost of information-set decoding attacks on an $[n, k, t; r]$-RLCE scheme is equivalent to the cost of information-set decoding attacks on a standard $[(r + 1)n, k; t]$-McEliece scheme.

Taking into account of the cost of recovering McEliece encryption scheme secret keys from the public keys and the cost of recovering McEliece encryption scheme plaintext messages from ciphertexts using the information-set decoding methods, we generated a recommended list of parameters for RLCE scheme in Table 1 using the PARI/GP script by Peters [28]. For the recommended parameters, the default underlying linear code is taken as the Reed-Solomon code over $GF(q)$ and the value of r is taken as 1. For the purpose of comparison, we also list the recommended parameters from [7] for the binary Goppa code based McEliece encryption scheme. The authors in [7, 28] proposed the use of semantic secure message coding approach so that one can store the public key as a systematic generator matrix. For the binary Goppa code based McEliece encryption scheme, the systematic generator matrix public key is $k(n-k)$ bits. For RLCE encryption scheme over $GF(q)$, the systematic generator matrix public key is $k(n(r+1)-k)\log q$ bits. It is observed that RLCE scheme generally has larger but acceptable public key size. Specifically, for the same security level, the public key size for the RLCE scheme is approximately four to five times larger than the public key size for binary Goppa code based McEliece encryption scheme. For example, for the security level of 80 bits, the binary Goppa code based McEliece encryption scheme has a public key of size 56.2KB, and the RLCE-MDS scheme has a public key of size $267 \approx 5 \times 56.2$KB.

Table 1: Parameters for RLCE: n, k, t, q, key size ($r = 1$ for all parameters), where "360, 200, 80, 101KB" under column "RLCE-MDS code" represents $n = 360, k = 200, t = 80$.

Security	RLCE-MDS code	binary Goppa code [7]
60	360,200, 80, 101KB	1024, 524, 50, 19.8KB
80	560, 380, 90, 267KB	1632, 1269, 34, 56.2KB
128	1020, 660, 180, 0.98MB	2960, 2288, 57, 187.7KB
192	1560, 954, 203, 2.46MB	4624, 3468, 97, 489.4KB
256	2184, 1260, 412, 4.88MB	6624, 5129, 117, 0.9MB

6 Conclusions

In this paper, we presented techniques for designing general random linear code based public encryption schemes using any linear code. Heuristics and experiments encourages us to think that the proposed schemes are immune against existing attacks on linear code based encryption schemes such as Sidelnikov-Shestakov attack, filtration attacks, and algebraic attacks. In addition to being a post-quantum cryptographic technique, our scheme RLCE has recently been used by Wang and Desmedt [34] to design fully homomorphic encryption schemes.

Acknowledgments

I would like to thank several colleagues for very detailed comments and suggestionsto improve the presentation of this paper.

References

[1] M. Baldi, M. Bodrato, and F. Chiaraluce. A new analysis of the mceliece cryptosystem based on qc-ldpc codes. In *Security and Cryptography for Networks*, pages 246–262. Springer, 2008.

[2] M. Bardet, J.-C. Faugere, and B. Salvy. On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In *Proc. Int. Conference on Polynomial System Solving*, pages 71–74, 2004.

[3] 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 *EUROCRYPT 2012*, pages 520–536. Springer, 2012.

[4] T.P. Berger, P.-L. Cayrel, P. Gaborit, and A. Otmañi. Reducing key length of the mceliece cryptosystem. In *Progress in Cryptology–AFRICACRYPT 2009*, pages 77–97. Springer, 2009.

[5] T.P Berger and P. Loidreau. How to mask the structure of codes for a cryptographic use. *Designs, Codes and Cryptography*, 35(1):63–79, 2005.

[6] D. Bernstein, T. Lange, and C. Peters. Wild McEeliece. In *Selected Areas in Cryptography*, pages 143–158. Springer, 2011.

[7] D.J. Bernstein, T. Lange, and C. Peters. Attacking and defending the McEliece cryptosystem. In *Post-Quantum Cryptography*, pages 31–46. Springer, 2008.

[8] N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *EUROCRYPT 2000*, pages 392–407. Springer, 2000.

[9] A. Couvreur, P. Gaborit, V. Gauthier-Umaña, A. Otmani, and J.-P. Tillich. Distinguisher-based attacks on public-key cryptosystems using Reed–Solomon codes. *Designs, Codes and Cryptography*, pages 1–26, 2013.

[10] A. Couvreur, I. Márquez-Corbella, and R. Pellikaan. A polynomial time attack against algebraic geometry code based public key cryptosystems. In *Proc. ISIT*, pages 1446–1450. IEEE, 2014.

[11] A. Couvreur, A. Otmañi, and J.-P. Tillich. Polynomial time attack on wild McEliece over quadratic extensions. In *Advances in Cryptology–EUROCRYPT 2014*, pages 17–39. Springer, 2014.

[12] J.-C. Faugere. A new efficient algorithm for computing Gröbner bases without reduction to 0 (F5). In *Proc. ISSAC*, pages 75–83.

[13] J.-C. Faugere. A new efficient algorithm for computing Gröbner bases (F4). *J. Pure and Applied Algebra*, 139(1):61–88, 1999.

[14] J.-C. Faugere, V. Gauthier-Umaña, A. Otmani, L. Perret, and J.-P. Tillich. A distinguisher for high-rate mceliece cryptosystems. *Information Theory, IEEE Transactions on*, 59(10):6830–6844, 2013.

[15] J.-C. Faugere, A. Otmani, L. Perret, and J.-P. Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In *Eurocrypt 2010*, pages 279–298. Springer, 2010.

[16] J.-C. Faugere, L. Perret, and F. De Portzamparc. Algebraic attack against variants of mceliece with goppa polynomial of a special form. In *Advances in Cryptology–ASIACRYPT 2014*, pages 21–41. Springer, 2014.

[17] H. Janwa and O. Moreno. Mceliece public key cryptosystems using algebraic-geometric codes. *Designs, Codes and Cryptography*, 8(3):293–307, 1996.

[18] P. J. Lee and E. F. Brickell. An observation on the security of McEliece's public-key cryptosystem. In *EUROCRYPT'88*, pages 275–280. Springer, 1988.

[19] J. Leon. A probabilistic algorithm for computing minimum weights of large error-correcting codes. *IEEE Trans. Information Theory*, 34(5):1354–1359, 1988.

[20] C. Löndahl and T. Johansson. A new version of mceliece pkc based on convolutional codes. In *Information and Communications Security*, pages 461–470. Springer, 2012.

[21] I. Márquez-Corbella and R. Pellikaan. Error-correcting pairs for a public-key cryptosystem. *arXiv preprint arXiv:1205.3647*, 2012.

[22] A. May, A. Meurer, and E. Thomae. Decoding random linear codes in $\tilde{O}(2^{0.054n})$. In *ASIACRYPT 2011*, pages 107–124. Springer, 2011.

[23] A. May and I. Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In *EUROCRYPT 2015*, pages 203–228. Springer, 2015.

[24] R.J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN progress report*, 42(44):114–116, 1978.

[25] R. Misoczki and P. Barreto. Compact mceliece keys from goppa codes. In *Selected Areas in Cryptography*, pages 376–392. Springer, 2009.

[26] R. Misoczki, J.-P. Tillich, N. Sendrier, and P. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2069–2073. IEEE, 2013.

[27] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Prob. Control and Information Theory*, 15(2):159–166, 1986.

[28] C. Peters. Information-set decoding for linear codes over F_q. In *Post-Quantum Cryptography*, pages 81–94. Springer, 2010.

[29] E. Prange. The use of information sets in decoding cyclic codes. *IRE Trans. Information Theory*, 8(5):5–9, 1962.

[30] Victor Shoup. NTL: A library for doing number theory, 2001.

[31] V. M Sidelnikov and S. Shestakov. On insecurity of cryptosystems based on generalized Reed-Solomon codes. *Discrete Mathematics and Applications*, 2(4):439–444, 1992.

[32] V.M. Sidelnikov. A public-key cryptosystem based on binary reed-muller codes. *Discrete Mathematics and Applications*, 4(3):191–208, 1994.

[33] J. Stern. A method for finding codewords of small weight. In *Coding theory and applications*, pages 106–113. Springer, 1989.

[34] Y. Wang and Y. Desmedt. Towards fully homomorphic encryption schemes from codes. In *Submitted*, pages 1–1. Springer Press, 2015.

[35] Yongge Wang. Random linear code based public key encryption implementation http://webpages.uncc.edu/yonwang/rlce.html, 2015.

[36] C. Wieschebrink. Two NP-complete problems in coding theory with an application in code based cryptography. In *Proc. IEEE ISIT*, pages 1733–1737. IEEE Press, 2006.

[37] C. Wieschebrink. Cryptanalysis of the Niederreiter public key scheme based on GRS subcodes. In *Post-Quantum Cryptography*, pages 61–72. Springer, 2010.