The failure of McEliece PKC based on Reed-Muller codes.

October 9, 2013

I. V. Chizhov[1], M. A. Borodin[2]
[1] Lomonosov Moscow State University. email: ivchizhov@gmail.com, ichizhov@cs.msu.ru
[2] Lomonosov Moscow State University email: bor1m@mail.ru

Abstract

This paper describes new algorithm for breaking McEliece cryptosystem, built on Reed-Muller binary code $RM(r, m)$, which receives the private key from the public key. The algorithm has complexity $O(n^d + n^4 log_2 n)$ bit operations, where $n = 2^m, d = GCD(r, m-1)$. In the case of $GCD(r, m-1)$ limitation, attack has polynomial complexity. Practical results of implementation show that McEliece cryptosystems, based on the code with length $n = 65536$ bits, can be broken in less than 7 hours on a personal computer.

1 Introduction

McEliece cryptosystem — one of the oldest public-key cryptosystems. It was introduced in 1978 by R. G. McEliece. McEliece Cryptosystem based on $\mathbb{NP}$-hard (non-deterministic polynomial-time hard) problem in coding theory. The main idea of its construction is masking some code with efficient decoding algorithms under the code, which does not have a visible algebraic and combinatorial structure. Such codes are called generic codes. Binary Goppa codes are used for building original McEliece cryptosystem. Digital signature can be constructed based on McEliece cryptosystem [1]. McEliece cryptosystem is an alternative to RSA cryptosystems and ElGamal which is quite common in practice. However, the development of quantum computing may lead to the rejection of the use of these cryptosystems in post-quantum era because they will be not secure. Thus, the study of public-key cryptosystems, which security is not based on the complexity of the discrete logarithm problem and factoring integers is quite important.

In 1994 V.M.Sidelnikov proposed to used Reed-Muller codes $RM(r, m)$ for build McEliece cryptosystem [2].

Reed-Muller code $RM(r,m)$ is called the set of vector values Ω_f of all boolean functions $f(y_1,\ldots,y_m)$, the degree of non-linearity (maximum length monomial in the Zhegalkin polynomial of function f) does not exceed r [3]. That is

$$RM(r,m) = \{\Omega_f = (x_1,\ldots,x_n), n = 2^m \mid f(y_1,\ldots,y_m) = a_0 \oplus \bigoplus_{s=1}^{t} \bigoplus_{1\leqslant i_1<\ldots<i_s\leqslant m} a_{i_1,\ldots,i_s} y_{i_1}\ldots y_{i_s},\ t \leqslant r\}$$

Note that in the future, the word vector, unless otherwise said, refers to a row vector. It is known that the code $RM(r,m)$ has dimension $k = \sum_{i=0}^{r} \binom{m}{i}$, length $n = 2^m$, Hamming distance $d = 2^{m-r}$. Denote R is generating matrix of the Reed-Muller code $RM(r,m)$, which consists of a unit vector and vector-values of all monomials m, variable degree of non-linearity is not exceeding r.

$$R = \begin{pmatrix} G_0 \\ G_1 \\ \vdots \\ G_r \end{pmatrix},$$

where $G_0 = \Omega_1 = (1,1\ldots,1)$,

$$G_1 = \begin{pmatrix} \Omega_{y_m} \\ \vdots \\ \Omega_{y_2} \\ \Omega_{y_1} \end{pmatrix}, G_2 = \begin{pmatrix} \Omega_{y_{m-1}y_m} \\ \vdots \\ \Omega_{y_1y_3} \\ \Omega_{y_1y_2} \end{pmatrix}, G_r = \begin{pmatrix} \Omega_{y_{m-r+1}y_{m-r+2}\ldots y_m} \\ \vdots \\ \Omega_{y_1y_2\ldots y_{r-1}y_{r+1}} \\ \Omega_{y_1y_2\ldots y_{r-1}y_r} \end{pmatrix}$$

Further, for simplicity of presenting we will not make a difference in designation between vector of boolean functions and boolean function expressed as a Zhegalkin polynomial. Let us describe the structure of McEliece cryptosystem constructed on the basis of the Reed-Muller code. Private key cryptosystem is a pair (H,P). Here H — nonsingular $(k\times k)$-matrix over the field $F_2 = \{0,1\}$, chosen randomly and with uniform distribution from the set of nonsingular binary $(k\times k)$-matrices. P is permutation matrix, i.e. at each line and each column is exactly one unit, and has the dimensions $n\times n$. In other words, the matrix P simulates a permutation. Note, that in the private key matrix R is not included (although in the original cryptosystem it is in the private key), because it does not make sense because of the uniqueness of the Reed-Muller $RM(r,m)$ with parameters $[n,k]$.

Definition 1. Public key McEliece cryptosystem is the matrix G :

$$G = H\cdot R\cdot P.$$

Let us describe an algorithm of encryption. To encrypt the message m, with length k it is necessary:

Encryption algorithm.

1. Calculate $c' = mG$.

2. Select random n-dimensional vector e with weight $wt(e) = \lfloor \frac{d-1}{2} \rfloor$.

3. Calculate $c = c' + e$.

Let us describe the algorithm for decrypting the cryptogram c.

Decryption algorithm.

1. Calculate $c' = cP^{-1}$.

2. Due to the decoding algorithm $RM(r, m)$ let us represent c' as $c' = aR + e'$, for some $a \in F_2^k$ and a error vector $e' \in F_2^n$ with weight $\lfloor \frac{d-1}{2} \rfloor$.

3. Calculate $m = aR_J^{-1}H^{-1}$, where R_J — nonsingular submatrix $k \times k$ of matrix R.

2 Known attack

Let us describe a model scheme of attack on the McEliece cryptosystem built on Reed-Muller codes $RM(r, m)$, which was proposed L. Minder and A. Shokrollahi.

Step 1. Build the code $RM^\sigma(1, m)$ from the code $RM^\sigma(r, m)$.

Step 2. Find a permutation σ' such that $RM^{\sigma\cdot\sigma'}(1, m) = RM(1, m)$. Then σ' will be found, it will be satisfy $RM^{\sigma\cdot\sigma'}(r, m) = RM(r, m)$.

Theorem 1. *There is an algorithm with complexity $O(n^3)$ bit operations, which constructs a permutation σ' such that $RM^{\sigma\cdot\sigma'}(1, m) = RM(1, m)$ from the code generator matrix $RM^\sigma(1, m)$, $m \geqslant 3$.*

Proof. Let G — arbitrary generating$(m+1 \times 2^m)$-matrix code $RM^\sigma(1, m)$ containing unit row. Let G' — $(m \times 2^m)$-matrix obtained from G by removing that line. In the matrix G can not be the equal columns, as in this case, the matrix G has two identical columns, or three columns sum is zero. So, in this case, the Hamming distance for $(RM^\sigma(1, m))^\perp = RM^\sigma(m-2, m)$ should not be more, than three. But the Hamming distance $RM^\sigma(m-2, m)$ equals to $2^2 = 4$. This contradicts the assumption. Thus, in the matrix G' there are no two identical columns. Total number of columns this with length m is 2^m, so in the matrix G' there are all columns of length m. Take as σ' permutation that orders the columns of G in lexicographical order of numbers, which binary representation are the columns of G' (high-order bits from the top). Under the action of permutation σ' matrix G obviously transforms into a generator matrix R of code $RM(1, m)$ in standard form (up to accurate within). For find a permutation we need sort the columns of the matrix G'. The complexity of sorting is $O(n \log_2 n)$. Finally, we need to obtain a matrix containing the row of units from an arbitrary generator matrix G of code $RM^\sigma(1, m)$. It is sufficient to solve the system of linear equations:

$$(a_1 \dots a_{m+1}) \cdot G = (1 \dots 1).$$

Then the matrix AG,

$$\text{where } A = \begin{pmatrix} a_1 & \dots & a_{m+1} \\ a_1 \oplus 1 & \dots & a_{m+1} \\ \vdots & \dots & \vdots \\ a_1 & \dots & a_{m+1} \oplus 1 \end{pmatrix},$$

will be looked for. Solving the system of linear equations and calculating matrix multiplication requires $O(n^3)$ operetions.

□

The question is how to write $RM^\sigma(1,m)$ from code $RM^\sigma(r,m)$. The first method, proposed by L. Minder is to accumulate sufficient number of codewords of minimum weight, and after it build the code $RM^\sigma(r-1,m)$ from $RM^\sigma(r,m)$. Next, in the same manner we can build the code $RM^\sigma(r-2,m)$ from the code $RM^\sigma(r-1,m)$. Continuing the construction, we get the code $RM^\sigma(1,m)$ in the end.

The theorem, which is a consequence of the work of L. Minder and A. Shokrollahi, is holds.

Theorem 2. *There is an algorithm with complexity $O(n^r)$ bit operations, which building code $RM^\sigma(r-1,m)$ from the code $RM^\sigma(r,m)$.*

3 The theoretical results

This paper proposes a different approach to building code $RM^\sigma(1,m)$.

Theorem 3. *There is an algorithm which builds code $RM^\sigma(r_1+r_2,m)$ from the codes $RM^\sigma(r_1,m)$ and $RM^\sigma(r_2,m)$ and requires $O(n^4)$ bit operations, where n is the length of the viewed codes.*

Proof. Let us see the code $RM^\sigma(r_1,m)$ basis $\{f_1,f_2,\dots,f_{k_1}\}$ and the code $RM^\sigma(r_2,m)$ basis $\{g_1,g_2,\dots,g_{k_2}\}$. View the code C, which is linear span of the vectors $\{f_1g_1,f_1g_2,\dots,f_1g_{k_2},f_2g_1,\dots,f_2g_{k_2},\dots,f_{k_1}g_1,\dots,f_{k_1}g_{k_2}\}$. Prove that the resulting code equal the code $RM^\sigma(r_1+r_2,m)$. On the one hand, since the degree of multiplication $(f_i)^{\sigma^{-1}}(g_j)^{\sigma^{-1}}, 1 \leqslant i \leqslant k_1, 1 \leqslant j \leqslant k_2$, does not exceed the sum of the degrees of each function, entered in, i.e.:

$$\deg((f_i)^{\sigma^{-1}}(g_j)^{\sigma^{-1}}) \leqslant \deg(f_i^{\sigma^{-1}}) + \deg(g_j^{\sigma^{-1}}) \leqslant r_1+r_2, \;\; 1 \leqslant i \leqslant k_1, 1 \leqslant j \leqslant k_2,$$

then $C \subseteq RM^\sigma(r_1+r_2,m)$.

Prove the reverse inclusion. For this we give the lower bound of the dimension of the code C. Due to the fact that the dimension of the code is not changed by being subjected to a permutation, then we estimate the dimension of the code $C^{\sigma^{-1}}$. This code is a linear span of the vectors $(f_i)^{\sigma^{-1}}(g_j)^{\sigma^{-1}} = f'_i \cdot g'_j, 1 \leqslant i \leqslant k_1, 1 \leqslant j \leqslant k_2$. Herewith, $\{f'_1,f'_2,\dots,f'_{k_1}\}$ — basis of the code $RM(r_1,m)$, and $\{g'_1,g'_2,\dots,g'_{k_2}\}$ — basis of the code

$RM(r_2, m)$. Without loss of generality, we assume that $r_1 \leqslant r_2$. We prove that the code $C^{\sigma^{-1}}$ will contain the following linearly independent vectors

$$\{\Omega_1, \{\Omega_{y_{j_1} y_{j_2} \ldots y_{j_s}}\}_{s=1,\ldots,r_1+r_2, 1 \leqslant j_1 < j_2 < \ldots < j_s \leqslant m}\}.$$

This will be sufficient to prove the theorem. Indeed, the number of these vectors is

$$\sum_{i=0}^{r_1+r_2} \binom{m}{i} = \dim RM(r_1 + r_2, m).$$

So $\dim C = \dim C^{\sigma^{-1}} \geqslant \dim RM(r_1 + r_2, m) = \dim RM^{\sigma}(r_1 + r_2, m)$ and $C = RM^{\sigma}(r_1 + r_2, m)$. View the vector $f \in \{1, y_{i_1} y_{i_2} \ldots y_{i_t}, 1 \leqslant t \leqslant r_1, 1 \leqslant i_1 < i_2 < \ldots < i_t \leqslant m\}$. It lies in the code $RM(r_1, m)$, and in the code $RM(r_2, m)$, so it can be expanded in the basis of these codes:

$$f = \sum_{i=1}^{k_1} \alpha_i f'_i = \sum_{j=1}^{k_2} \beta_j g'_j.$$

Then we have the chain of equalities

$$f = f \cdot f = \sum_{i=1}^{k_1} \alpha_i f'_i \cdot \sum_{j=1}^{k_2} \beta_j g'_j = \sum_{i,j} \alpha_i \beta_j f'_i g'_j.$$

As $f'_i g'_j \in C^{\sigma^{-1}}$, so the vector f also belongs to the code $C^{\sigma^{-1}}$.

View the vector $f \in \{y_{i_1} y_{i_2} \ldots y_{i_t}, r_1 + 1 \leqslant t \leqslant r_1 + r_2, 1 \leqslant i_1 < i_2 < \ldots < i_t \leqslant m\}$. Then f the following expansion in the basis holds:

$$f = y_{i_1} y_{i_2} \ldots y_{i_{r_1}} y_{i_{r_1+1}} \ldots y_{i_t}.$$

Herewith, $y_{i_1} y_{i_2} \ldots y_{i_{r_1}} \in RM(r_1, m)$, and $y_{i_{r_1+1}} \ldots y_{i_t} \in RM(r_2, m)$, as $1 \leqslant t - r_1 \leqslant r_2$. So the expansions hold:

$$y_{i_1} y_{i_2} \ldots y_{i_{r_1}} = \sum_{i=1}^{k_1} \alpha_i f'_i; \;\; y_{i_{r_1+1}} \ldots y_{i_t} = \sum_{j=1}^{k_2} \beta_j g'_j.$$

Hence we obtain

$$f = \sum_{i=1}^{k_1} \alpha_i f'_i \cdot \sum_{j=1}^{k_2} \beta_j g'_j = \sum_{i,j} \alpha_i \beta_j f'_i g'_j.$$

As $f'_i g'_j \in C^{\sigma^{-1}}$, the vector f also belongs to the code $C^{\sigma^{-1}}$. It remains to estimate the number of operations required to build the code $RM^{\sigma}(r_1 + r_2, m)$. Let $\{f_1, f_2, \ldots, f_{k_1}\}$ — arbitrary basis of code $RM(r_1, m)$, $\{g_1, g_2, \ldots, g_{k_2}\}$ — arbitrary basis of code $RM(r_2, m)$, and let $L = \{h_1, h_2, \ldots, h_k\}$ consists of all linearly independent multiplication $f_i \cdot g_j, 1 \leqslant$

$i \leqslant k_1, 1 \leqslant j \leqslant k_2$. From the construction the linear span L coincides with C. To build the code C is enough to construct a set L, that is its basis.

We form L as follows.Take a vector $f_1 g_1$. Add to it the vector $f_1 g_2$. And we reduce $f_1 g_2$ to upper triangular form. This requires one bit-wise addition of two vectors. If as a result we obtain a linearly dependent system, then we discard the vector $f_1 g_2$ and move on to the next vector, repeating the reducing to upper triangular form. Suppose that we have tested x_1 vectors to obtain a linearly independent system of two vectors. Then there will be at most x_1 addition of vectors of length n. Similarly construct a system of three linearly independent vectors. Given that the system of the first two vectors is reduced to upper triangular form, the process of bringing the system of three vectors to upper triangular form we need 2 additions of vectors of length n. Suppose that we have tested x_2 candidate for a third vector, then only need to $2x_2$ addition of vectors of length n. Continuing the arguments, we conclude that the complexity of building a complete code base will require

$$N = \sum_{i=1}^{k(1,2)-1} i \cdot x_i$$

addition of vectors of length n, since the total number of tested vectors can not be greater than the number of multiplication $f_i g_j$. $\sum_{j=1}^{k(1,2)-1} x_j = k_1 \cdot k_2$, where $k(1,2)$ — the dimension of the code $RM^{\sigma}(r_1 + r_2, m)$. The estimation of Nis :

$$N \leqslant k(1,2) \cdot \sum_{i=1}^{k(1,2)-1} x_i = k(1,2) \cdot k_1 \cdot k_2.$$

As $k(1,2), k_1, k_2 \leqslant n$, then $N \leqslant n^3$. Then the number of bit operations for the construction of basis will be equal $O(N \cdot n) = O(n^4)$, where n is the length of code.

□

Theorem 4. *There is an algorithm which builds the code $RM^{\sigma}(m - r - 1, m)$ from code $RM^{\sigma}(r, m)$ with complexity $O(n^3)$, where n — length of $RM(r, m)$.*

Therefore, to build the code $RM^{\sigma}(1, m)$ from code $RM^{\sigma}(r, m)$ we can use the following operation:

1. Multiplication $\odot$ of codes $RM^{\sigma}(r_1, m)$ and $RM^{\sigma}(r_2, m)$ such that $r_1 + r_2 \leqslant m - 2$: $RM^{\sigma}(r_1, m) \odot RM^{\sigma}(r_2, m) = RM^{\sigma}(r_1 + r_2, m)$.

2. Obtaining the orthogonal code $\perp$ from code $RM^{\sigma}(r, m)$: $(RM^{\sigma}(r, m))^{\perp} = RM^{\sigma}(m - r - 1, m)$.

Also, we can consider the superposition $\circ$ of these operations. For simplicity, instead of $RM^{\sigma}(r, m)$ will use the notation (r, m).

Definition 2. Let $U = \{x \odot y, x^{\perp}\}$. Introduce the concept of the formula U.

1. Element $u \in U$ is a formula over U with depth 1.

2. Let v_1, v_2 — formulas over U with depth s. Then $v_1 \odot v_2$ and $v_1^{\perp}$ are formulas over U with depth $s+1$.

3. There is no other formula.

Definition 3. Closure $[(r,m)]$ of code (r,m) is the set of all such codes (t,m), which can be obtained from (r,m) with application to it of all possible formulas of U.

We denote

$$t_{min}((r,m)) = \min_{(t,m)\in[(r,m)]} t.$$

Let us study set $[(r,m)]$.

Proposition 1. *Code* (t,m), $1 \leqslant t \leqslant m-2$, *belongs* $[(r,m)]$, *if and only if there are integers* a *and* $b \neq 0$ *such that* $t = a\cdot(m-1) + b\cdot r$.

Proof. Prove, that if $(t,m) \in [(r,m)]$, then there are integers a and $b \neq 0$ such that $t = a\cdot(m-1) + b\cdot r$. We use induction on the depth of the formula. If (t,m) obtained from (r,m) with depth 1, then either $(t,m) = (r,m)^{\perp} = (m-r-1,m)$, or $(t,m) = (r,m)\odot(r,m) = (2r,m)$. Thus, the proposition for the codes obtained by applying the formulas of depth 1 is executed.

Suppose, that the proposition hold for all (t,m), obtained from (r,m) with application of the formulas of depth $\leqslant s-1$. We prove the proposition for all (t,m), obtained from (r,m) with application of the formulas of depth s. There are two cases:

1. Code $(t,m) = (t',m)\odot(t'',m)$, where code (t',m) and (t'',m) are obtained from (r,m) with the aid of formula with depth $\leqslant s-1$. By the induction hypothesis there are integers $a', b' \neq 0, a'', b'' \neq 0$, that $t' = a'(m-1) + b'r$ и $t'' = a''(m-1) + b''r$. From the definition of the operation $\odot$ we find that $t = (a'+a'')(m-1) + (b'+b'')r$. Let $b'+b'' = 0$. Then $t = (a'+a'')(m-1)$ and $t > m-2$, or $t \leqslant 0$, and the operation $\odot$ could not be applied to the codes (t',m) and (t'',m). The statement in this case is true.

2. Code $(t,m) = (t',m)^{\perp}$, where code (t',m) obtained from (r,m) with the aid of formula with depth $\leqslant s-1$. By the induction hypothesis there are integers $a', b' \neq 0$, that $t' = a'(m-1) + b'r$. From the definition of the operation $\perp$ we find that $t = m-1-a'(m-1) - b'r = (-a'+1)(m-1) + (-b')r$, that is, in this case, the statement is also true.

Let us prove the opposite, that is, if there are integers a и $b \neq 0$ such that $t = a\cdot(m-1) + b\cdot r$, then $(t,m) \in [(r,m)]$. There are four cases:

1. $a \geqslant 0, b > 0$. Given that $1 \leqslant t \leqslant m-2$, we obtain $a(m-1) + br \leqslant m-2$. So $br \leqslant (1-a)(m-1) - 1$. If $a \geqslant 1$, then

$$br \leqslant -((a-1)(m-1)+1) \leqslant 0,$$

so $b = 0$, which is impossible by assumption. So, $a = 0$. Then t has the form $t = br \leqslant m-2$. Hence, (t, m) obtained by the b-time application of $\odot$ to code (r, m) :

$$(t, m) = \underbrace{(r, m) \odot (r, m) \odot \ldots \odot (r, m)}_{b}.$$

In this case, proposition is proved.

2. $a \geqslant 0, b < 0$. We represent $b' = -b > 0$ as $b' = q \cdot a - s$, where q, s — non-negative integers and $0 \leqslant s \leqslant a - 1$. Then

$$t = a(m-1) - b'r = a(m - 1 - qr) + sr.$$

As $s, r \geqslant 0$ and $t \leqslant m-2$, then $0 < a(m-1-qr) \leqslant m-2$. So $0 < m-1-qr \leqslant m-2$ and $1 \leqslant qr \leqslant m-2$. So, by the operation $\odot$ we can get the code (qr, m). By applying to the operation $\perp$, which we can get the code $(m - 1 - qr, m)$. Now this code, we can apply the operation $\odot$ to this code. Then, we get the code $(a(m - 1 - qr), m)$. It remains to prove that it is possible to get code (sr, m) from (r, m). Then the required code (t, m) can be obtained with the operation $\odot$:

$$(a(m - 1 - qr), m) \odot (sr, m) = (t, m).$$

As $m - 1 - qr, a > 0$ и $t \leqslant m - 2$, we obtained $0 \leqslant sr \leqslant m - 2$. i.e. code (sr, m) can be obtained as follow:

$$(sr, m) = \underbrace{(r, m) \odot (r, m) \odot \ldots \odot (r, m)}_{s}.$$

3. $a \leqslant 0, b < 0$. As $t = a(m - 1) + br > 0$, this case is impossible.

4. $a \leqslant 0, b > 0$. Consider the code $(m - 1 - t, m) = ((1 - a)(m - 1) - br, m)$. For it $(1 - a) \geqslant 0$ и $(-b) < 0$. Since the case 2 we can get the code $(m - 1 - t, m)$, by applying to the operation $\perp$, we can get the source code.

□

Proposition 2. *Equality* $t_{min}(r, m) = GCD(r, m - 1)$. holds.

Proof. By proposition 1 $[(r, m)]$ consists of all codes (t, m), for which there exist integers a and b, $b \neq 0$, that

$$t = a(m - 1) + br.$$

It is clear that $d = \mathrm{GCD}(r, m - 1)$ has this property, then $(d, m) \in [(r, m)]$. Let $(t, m) \in [(r, m)]$ and $t < d$. As d divides $m-1$ and divides r, then d divides t, that it is impossible.

□

Proposition 3. *Let GCD$(r, m-1) = d$. Then there is an algorithm that builds code (d, m) from the code (r, m) with complexity $O(n^4 \log_2 n)$.*

Proof. By Proposition 2 we can use operations $\odot$ and $\perp$ to the code (r, m) to get code (d, m), where $d = \mathrm{GCD}(r, m-1)$. We calculate the bit complexity of such system. As $(d, m) \in [(r, m)]$ then from the proposition 1 follows that there exist two integers a and $b \neq 0$, that

$$d = a(m-1) + br.$$

We can assume that $|b| \leqslant m-1$. If not present $|b|$ as $b = a'(m-1)+b'$, where $|b'| \leqslant m-1$ and move on to a new concept: $d = (a + a'r)(m-1) + b'r$. In proposition 1 we consider the 3 cases.

1. $a \geqslant 0, b > 0$. As follows from the proof of proposition 1 in this case $d = br$. We conclude that d divided by r, so the source code (r, m) was a source (d, m).

2. $a \geqslant 0, b < 0$. As follows from the proof proposition 1 in this case $d = a(m-1-qr) + sr, 0 < m-1-qr \leqslant m-2, 1 \leqslant sr \leqslant m-2$. So, by the operation $\odot$ we can get the code (qr, m), by applying the operation $\perp$, we can get the code $(m-1-qr, m)$. By theorem 3 and theorem 4 this requires $O((q-1)n^4 + n^3)$ operations. Now, using the operation $\odot$ to the resulting code, we get the code $(a(m-1-qr), m))$ for $O((q-1+a-1)n^4 + n^3)$ operations. Code (sr, m) can be obtained for $O((s-1)n^4)$ operations, so the required code (d, m) can be obtained for $O((q+a+s-2)n^4 + n^3)$ operations. Given that $s < a$ and $q \leqslant |b|$, we find that the total number of operations is equal to $O((|b|+2a)n^4)$. As $b \leqslant m-1 < m$ and $a = \frac{d+|b|r}{m-1} \leqslant \frac{r+(m-1)r}{m-1} = \frac{rm}{m-1} < m$, we finally find that the complexity equals $O(mn^4) = O(n^4 \log_2 n)$.

3. $a \leqslant 0, b > 0$. This case differs from the case 2 by additional application of the operation $\perp$, as well as the complexity of the operation is $O(n^3)$. The overall complexity equals $O(n^4 \log_2 n)$.

□

A consequence of the proposition 3 and theorem 1 will be the next main theorem.

Theorem 5. *Suppose GCD$(r, m-1) = 1$. Then there is the algorithm with the complexity of $O(n^4 \log_2 n)$ bit operations, which find a permutation σ' such that $RM^{\sigma\cdot\sigma'}(r, m) = RM(r, m)$ from the code generator matrix $RM^{\sigma}(r, m)$.*

Theorem 6. *Suppose GCD$(r, m-1) = d > 1$. Then there is the algorithm with the complexity $O(n^d + n^4 \log_2 n)$ bit operations, which find a permutation σ' such that $RM^{\sigma\cdot\sigma'}(r, m) = RM(r, m)$ from generator code matrix $RM^{\sigma}(r, m)$.*

Proof. From the code (r, m) we can build the code (d, m) this the complexity $O(n^4 \log_2 n)$. Next, from the code (d, m) we can build the code $(d-1, m)$ with complexity $O(n^d)$ operations and the code $(m-d-1, m)$ with $O(n^3)$ operations. Next, build the code $(m-d-1, m) \odot (d-1, m) = (m-2, m)$. This requires $O(n^3)$ bit operations. Total obtain the required number of operations.

□

4 Practical results

To get practical results we implement the algorithm in software and run it multiple times on PC with a 2.1GHz Intel Centrino processor and 2Gb RAM. The average time of the algorithm for different parameters is given in the table. "M" — Denotes that our algorithm is not better than Minder's algorithm.

If the algorithm reduces the original problem (r, m) to the problem with less complexity (d, m), it is noted in the table with symbols (d, m).

(r,m)	$m = 8$	$m = 9$	$m = 10$	$m = 11$	$m = 12$	$m = 13$	$m = 14$	$m = 15$	$m = 16$
$r = 2$	0.007s	M	0.48s	M	6s	M	3m13s	M	2h30м
$r = 3$	0.01s	0.2s	M	1.35s	19s	M	5m29s	30m31s	M
$r = 4$	0.043s	M	0.43s	(2,11)	15s	M	7m10s	(2,15)	3h28м
$r = 5$	0.042s	0.4s	0.8	M	16.5s	2m1s	14м12s	53m	M
$r = 6$		(2,9)	(3,10)	(2,11)	23s	M	9m28s	14m16s	(3,16)
$r = 7$			0.86s	3.2s	25s	3м16с	10m54s	M	6h43m

5 Conclusion

The article describes a new algorithm for the attack on the McEliece cryptosystem, based on the Reed-Muller code (r, m). Article provides theoretical proof of the method and bit complexity of the algorithm. Comparing the practical results obtained during the implementation of the new method, with the already published in the paper [4], it is easy to see that for the set of parameters satisfying $\mathrm{GCD}(r, m) = 1$, the proposed algorithm is significantly more effective. In particular, the proposed attack allows to carry out breaking cryptosystems of McEliece, based on the code with length $n = 65536$ bits, in less than 7 hours on a personal computer.

6 Bibliography

References

[1] Courtois, Nicolas T., Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. Advances in Cryptology–ASIACRYPT 2001. Springer Berlin Heidelberg, 2001. 157-174.

[2] V. M. Sidelnikov, Open coding based on Reed-Muller binary codes, Diskr. Mat., 1994, 6:2, 3-20.

[3] MacWilliams F. J. NJ a. Sloane, The Theory of Error-Correcting Codes. - 1977.

[4] Minder L. and Shokrollahi A. Cryptanalysis of the Sidelnikov cryptosystem LNCS. 2007. V. 4515. P. 347-360.