Selecting Parameters for Secure McEliece-based Cryptosystems

Robert Niebuhr[1], Mohammed Meziani[2], Stanislav Bulygin[2], and Johannes Buchmann[1]

[1] TU Darmstadt, FB Informatik
Hochschulstrasse 10, 64289 Darmstadt, Germany
{rniebuhr, buchmann}@cdc.informatik.tu-darmstadt.de
[2] CASED – Center for Advanced Security Research Darmstadt,
Mornewegstrasse 32, 64293 Darmstadt, Germany
{mohammed.meziani, stanislav.bulygin}@cased.de

Abstract. In 1994, P. Shor showed that quantum computers will be able to break cryptosystems based on integer factorization and on the discrete logarithm, e.g. RSA or ECC. Code-based crytosystems are promising alternatives to public key schemes based on these problems, and they are believed to be secure against quantum computer attacks. In this paper, we solve the problem of selecting optimal parameters for the McEliece cryptosystem that provide security until a given year and give detailed recommendations. Our analysis is based on the lower bound complexity estimates by Sendrier and Finiasz, and the security requirements model proposed by Lenstra and Verheul.

Key words: Post-quantum cryptography, codes, McEliece, key length, parameters.

1 Introduction

Public key cryptosystems are essential components of IT-security solutions. When public key cryptosystems are used in such solutions, appropriate key sizes must be selected. For RSA, EC-ElGamal, and other factoring and discrete logarithm-based systems that are used today, Lenstra and Verheul [13] proposed a model on how to select appropriate keys that provide security until a given year.

We know from [19] that quantum computers can break all these "classical" systems. This is why post-quantum public key cryptosystems, which remain secure in the presence of quantum computers, must be found, and the problem of selecting cryptographic keys for such systems must be solved.

This paper solves the problem of selecting appropriate parameters for the McEliece cryptosystem based on binary Goppa codes. This cryptosystem is as old as RSA, and it is expected to resist quantum computer attacks. Together with cryptographic schemes based on lattices, multivariate polynomials, or on hash functions, it is one of the most interesting post-quantum candidates.

To instantiate McEliece, a vector space dimension n, a code dimension k, and an error-correcting capability t are chosen. The selection of a parameter set (n, k, t) determines the properties of the cryptosystem instance, e.g. security level, size of the public and private keys, information rate, computation complexity. Two of these parameters can be chosen independently; usually these are n and t, and k is treated as a function of these parameters.

Our main goal for the selection of parameters is to satisfy a minimum security level. Given that the first goal is satisfied, we want to minimize the public key size. The results of our optimization can be found in Table 2 on page 8.

There are, of course, other properties that can be optimized, e.g. encryption or decryption speed. The reason why we focus on the key sizes is the following. While the McEliece cryptosystem is very efficient and does not require special hardware, it suffers from the drawback of having large public and private keys. For example, in a smart card implementation we analyzed[3], the computation time for encryption and decryption accounted for

[3] Implemented by Falko Strenzke, FlexSecure GmbH, Darmstadt, Germany

only 5% of the total time, while the data transfer of the public key required 95% of the total time. Therefore, reducing the public and private key size is an important target.

The paper is organized as follows: In Section 2, we will review relevant details from coding theory and the McEliece cryptosystem, so as to introduce our notations and review the commonly-used relevant terminology. Section 3 describes our security model. Based on this model we explain our approach for selecting optimal parameters and present our results in Section 4. In Section 5, we summarize our findings and formulate topics for further research.

2 Preliminaries

2.1 Coding theory

Error-correcting codes are widely used in practice, especially for information transfer over noisy channels. Applications are: CDs/DVDs, DSL, DVB-TV, mobile phones, satellite communication, and many more. In cryptography, error-correcting codes also have applications. Many code-based cryptosystems make use of the error-correcting capability of an underlying code by intentionally adding errors to the message, such that these errors may only be efficiently corrected using the private key. The extensive research and many hardware developments due to the increasing popularity of error-correcting codes make code-based cryptosystems a promising candidate for cryptography.

In this section we provide a short introduction to error-correcting codes and the McEliece cryptosystem. A more detailed description can be found in [14].

In general, a linear code is a k-dimensional subspace of an n-dimensional vector space over a finite field $\mathbb{F}_q$, where k and n are positive integers with $k < n$ and q a prime power. The elements of a code are called codewords. The information rate (or rate) is defined as $R = k/n$. The weight of a vector x, denoted by $wt(x)$, is the number of its non-zero entries, and the Hamming distance of two vectors is the weight of their difference. The minimum distance d of a code is the minimum distance between any two distinct codewords; a code with these properties is denoted as an $[n, k, d]$ code.

The error-correcting capability of a linear code is the maximum number $t^* = \lfloor (d-1)/2 \rfloor$ of errors that the code is able to decode. Sometimes we want to add fewer errors than theoretically possible. Therefore, we denote by (n, k, t) an $[n, k, d]$ code with $t \leq t^*$. In this paper, we restrict our attention to the binary case, i.e., $q = 2$. In the rest of this section, we fix n, k, t, and a linear (n, k, t) code $\mathcal{C}$ over $\mathbb{F}_2$.

The security of code-based cryptosystems is based on the difficulty of some classical problems of coding theory. The most relevant in our context is:

Coset weights: *Let r and n be two positive integers such that $r < n$. Given a binary matrix $H \in \mathbb{F}_2^{r \times n}$, a vector $s \in \mathbb{F}_2^r$ and a non-negative integer t. Find a vector $x \in \mathbb{F}_2^n$ of weight $wt(x) \leq t$ such that $Hx^T = s$.*

It was proved in [3] that the respective decisional problem is NP-complete.

2.2 The McEliece cryptosystem

The McEliece public-key encryption scheme was presented by R. McEliece in 1978 ([15]) and is one of the best-studied code-based schemes up to date.

The idea behind this scheme is to first select a particular (linear) code for which an efficient decoding algorithm is known, and then to create a trapdoor function by disguising the code as a general linear code. Since the problem of decoding a linear code is NP-complete, a description of the original code can serve as the private key, while a

description of the transformed code can serve as the public key.

The McEliece encryption scheme using Goppa codes, as originally proposed by McEliece, has resisted cryptanalysis to date. The system was also the first public-key encryption scheme to use randomization in the encryption process. Although computationally very efficient, the McEliece encryption scheme has received little attention in practice because of the very large public keys.

Construction: We provide an overview on the construction of the McEliece cryptosystem. See [17] for more details.
Fix n, k, t as in the previous section. Let $\mathcal{C}$ be a binary (n, k, t)-Goppa code defined by a Goppa polynomial g of degree t, together with a fast decoding algorithm that can correct up to t errors. In the case of Goppa codes, we have the relation $k = n - t \cdot \lceil \log_2(n) \rceil$. Let G be a $k \times n$ generator matrix for $\mathcal{C}$. To create the disguise, choose a random $k \times k$ invertible binary matrix S (the scrambler) and let P be a random $n \times n$ permutation matrix. The matrix,

$$\widehat{G} = SGP$$

is made public, while g, S, and P form the private key (G can be constructed from g and the set of points where g is evaluated).

Encryption: Represent the plaintext as a vector m of length k, choose a random error vector e of weight at most t, and compute the ciphertext $c = m\widehat{G} + e$.

Decryption: To recover the plaintext m from c, first compute $\widehat{c} = cP^{-1} = mSG + eP^{-1}$. As P is a permutation matrix, eP^{-1} has the same weight as e. Therefore, the decoding algorithm for the code generated by G finds and corrects these errors, resulting in mSG, which is then decoded to $\widehat{m} = mS$. Finally, compute $m = \widehat{m}S^{-1}$.

Remark 1. We assume $\widehat{G}$ to be in systematic form, which can be achieved by Gaussian elimination:

$$\widehat{G} = [I_k | G'],$$

where I_k is the identity matrix of size k. This allows to store the public key more efficiently and reduces its size to $k(n - k)$ bits. See [17] for more details.

3 Our Security Model

3.1 Security of the McEliece cryptosystem

Since the introduction of the McEliece cryptosystem, different attacks against it have been proposed in the literature. Some attacks, called structural attacks, aim at recovering the private key from the public key. A detailed overview of these attacks can be found in [7]. Other attacks, called decoding attacks, attempt to recover the plain text from a given cipher text, and most of them are based on Information Set Decoding (ISD) or, in some cases, on the Birthday algorithm, generalizations, and improvements of these two. Other types of attacks has been proposed, like iterative decoding [10] and statistical decoding [11,16], but they achieved little success. The ISD attacks seem to have the lowest complexity. For these reasons, we base our security analysis on the complexity of this kind of attacks.

The working principle of an ISD attack is as follows: Consider an (n, k, t)-linear code $\mathcal{C}$ of a generator matrix G of rank k. Denote by $c \in \mathbb{F}_2^n$ a received word, i.e. $c = x + e$ with $x = m\widehat{G} \in \mathcal{C}$ and $e \in \mathbb{F}_2^n$ is the added error of weight t, where $m \in \mathbb{F}_2^k$ is the plain text. To recover m from c, an attacker randomly selects a subset $I \subset \{1, \ldots, n\}$ of size k, called an *Information Set*, in the hope that the sub-matrix $\widehat{G}_I$ formed by the I-indexed columns of $\widehat{G}$ is invertible and c_I is error-free, i.e. $e_I = \mathbf{0}$, where z_I denotes the I-indexed positions of word z. If this is the case,

then $x = c_I \cdot \widehat{G}_I^{-1}$ and the plain text is $m = x \cdot \widehat{G}$.

Over the years, several ISD algorithms have been described and improved. The most important of these are listed in Table 1, together with their respective binary work factor to decode a $(1024, 524, 50)$ Goppa code (these are the original McEliece parameters). An explicit bounds for some of these algorithms can be found in [8].

Table 1. Complexity of ISD algorithms against $(1024, 524, 50)$ McEliece cryptosystem

Year	Algorithm	Log. of binary work factor
1986	Adams-Mejier [1]	80.7
1988	Lee-Brickell [12]	70.89
1989	Stern [20]	66.21
1994	Canteaut-Chabanne [5]	65.5
1998	Canteaut-Chabaud [6]	64.1
2008	Bernstein-Lange-Peters [4]	60.4
2009	Finiasz-Sendrier [9]	59.9

In 2009, Finiasz and Sendrier developed lower bounds for ISD algorithms [9] by analyzing an idealized ISD algorithm. Their work can be used to calculate conservative estimates for the security of arbitrary parameters. Therefore, we will use this algorithm to determine the security level of a given Goppa code.

3.2 Lenstra-Verheul Model

In [13], Lenstra and Verheul developed a formal model to find appropriate parameters for symmetric and some asymmetric cryptosystems. Their model is based on a set of assumptions that combine the impact of cryptanalytic progress and the effect of changes in computing environment. The key points of their model on which the choice of parameters depends are the following:

1. **Security margin:** is the year s in which attacks on a certain cryptographic primitive were infeasible. It defines the term of "adequate security". In order to determine this, Lenstra and Verheul evaluate a function IMY(s). This abbreviation stands for "Infeasible number of MIPS-years[4] for year s", and it refers to the minimum computational effort that is expected to be infeasible to do in year s. In [13] the default value of s is 1982 which represents the last year for which it is assumed that a 56-bit key DES cryptosystem provides adequate security for commercial use. The computational effort for breaking the 56-bit DES system was estimated to be $5 \cdot 10^5$ MIPS-years.

2. **Computing environment:** estimates the changes in computational power available to attackers. This estimation is based on a slight variation of Moore's law by introducing three three variables a, b, and c that specify the changes in hardware speed, IT budget, and price over time. The definitions of these variables and their default values are as follows:
   - a is the expected average number of months in which processor speed and memory size increase by a factor of two. The default value is $a = 18$, which is the value specified by Moore's law and is so far in line with current hardware developments. In this paper we are going to use the same value due to the fact that over the last years, hardware development has resulted in a doubling of transistors (for a fixed price) every 12-24 months[5]. Thus, a default of 18 is a compromise of this historic data. Also, opinions differ in whether hardware development will slow down or new technologies will further accelerate it;
   - $c \in \{0, 1\}$ indicates how to interpret the variable a: For $c = 0$, the amount of computing power and memory *which is available to an attacker* doubles every a months, while for $c = 1$, the computing power

[4] MIPS = million instructions per second
[5] See http://wi-fizzle.com/compsci/

and RAM *for a given price* double every a months. We will use $c = 1$ since the historic trend mentioned above refers to a fixed price.
- b is defined as the average number of years it takes for IT budgets to double. According to historic data[6], the US Gross National Product has doubled approx. every 10.5 years over the last 30 years. Since the exact growth varies every year, we will use an average value to extrapolate over a larger period of time. Our default setting for b is 10.

3. **Cryptanalysis:** the expected cryptanalytic progress. It is measured by the number of months r it is expected for cryptanalytic attacks to become twice as effective. We estimate this number by attacks against code-based cryptosystems only, since the cryptanalytic development can be very different for other cryptosystems. Lenstra and Verheul's default value is $r = 18$. In code-based cryptography, we find it reasonable to assume that the pace of future cryptanalytic developments and their impact will be relatively close to what we have seen from 1988 until 2009. By applying a linear regression on data points listed in Table 1, we get a line whose slope roughly equals -0.41 meaning that a twofold attack efficiency improvement will happen in each $1/0.41 \approx 2,44$ years. Also the value of r is $r = 2,44 \cdot 12 \approx 29.27$. In this paper, we take $r = 30$, which corresponds to 2.5 years.

Based on these variables, Lenstra and Verheul present a formula which can be used to derive lower bounds for cryptographic key sizes that offer at least a specified security margin until year y in the future (independent of the concrete asymmetric cryptosystem). To do this, they show how IMY(y) is derived from the parameters above. Given that breaking the DES system takes $5 \cdot 10^5$ MIPS-years, which was infeasible in the year s, the function IMY(y) is defined by:

$$\text{IMY}(y) = 5 \cdot 10^5 \cdot 2^{12(y-s)/a} \cdot 2^{c(y-s)/b} \quad \text{MIPS-years.} \tag{1}$$

With our default settings, it follows that in year y a computational complexity of

$$\text{IMY}(y) = 5 \cdot 10^5 \cdot 2^{\frac{23}{30}(y-1982)} \quad \text{MIPS-years} \tag{2}$$

provides an acceptable level of security. Furthermore, if we use as a data point the result that approximately $2^{60.4}$ binary operations are needed to break the original McEliece with parameters $(1024, 524, 50)$ [4], and expect cryptanalytic developments by a factor $2^{12(y-2008)/r}$ (with $r = 30$), we claim that a sufficient condition for security level, denoted by $S(n, k, t)$, of a McEliece instance with parameter set (n, k, t) providing an adequate security until a given year y is the following:

$$S(n, k, t) \geq \frac{\text{IMY}(y) \cdot 2^{12(y-2008)/30} \cdot 2^{60.4}}{1.7 \cdot 10^5}. \tag{3}$$

The value $1.7 \cdot 10^5$ is expressed in MIPS-years and obtained from the fact that the attack by Bernstein et al. [4] required 1400 CPU days on Q6600 quad processors. Assuming that a Q6600 processor [4] does approximately $44,000$ MIPS (SiSoft Sandra benchmark and [2]), this corresponds to $1.7 \cdot 10^5$ MIPS-years.

Therefore, the inequality (3) becomes:

$$S(n, k, t) \geq 2.9412 \cdot 2^{\frac{23}{30}(y-1982)+\frac{12}{30}(y-2008)+60.4} \tag{4}$$

4 Parameters selection

4.1 Our methodology

The problem of estimating secure parameters for the McEliece cryptosystem for a given year consists in obtaining, for the security level S calculated as above, a set of parameters that achieves this security level and provides the smallest key size among all other such sets. To solve this problem, we use the following methodology:

[6] See http://www.bea.gov

1. Based on simplified theoretical arguments we show that there exists an optimal rate R^* with $R^* \approx 0.8$ such that for a given key size the maximum of security is achieved at this rate.
2. We show how an instance attaining maximum security for a given key size can be used to solve the problem of finding the optimal key size for a given security level.
3. We present an algorithm that we use to find optimal instances that experimentally have a rate of ≈ 0.74, corresponding to the arguments from 1.

Already in [18] it was pointed out that the ISD-family of algorithms has an approximate complexity of

$$C(n, R) = p(n)2^{-t \log_2(1-R)}, \tag{5}$$

where $p(n)$ is some polynomial in n. For the classical ISD the degree of p is 3 and it gets lower for improvements. In the case of a t-error correcting Goppa code of length n and dimension $k = n - t\lceil \log_2 n \rceil$, the above formula becomes

$$C_G(n, R) = p(n)2^{c(R)n/\log_2 n(1+o(1))}, \tag{6}$$

where $c(R) = -(1-R)\log_2(1-R)$ is the complexity coefficient. In [18] it is also mentioned that, neglecting $p(n)$ and concentrating only on the exponential part, the following can be shown: for a given code length n, the highest complexity is achieved at an information rate of $1 - e^{-1} \approx 0.63$. Although we will compute our table using the lower bounds from [9], we would first like to provide some theoretical evidence that the optimal rate exists also for the problem of the smallest key size. Considering that the numerous improvements of the ISD enhance only the polynomial part significantly, the reasoning seems to be sound. In the following lemma we simplify $C_G(n, R)$ to

$$C_G(n, R) = 2^{(c(R)n/\log_2 n(1+o(1)))}, \tag{7}$$

similarly to [18].

Lemma 1. *Given the key size K, the maximum complexity of an ISD-like algorithm as per (7) is achieved at an information rate $R^* \approx 0.8$.*

Proof. The proof is quite elementary, but we still provide it for completeness. First, from the formula $K = R(1-R)n^2$ we have that $n = \sqrt{K/(R(1-R))}$. Now if we substitute this expression on n in (7) we have that

$$C_G(K, R) = 2^{(c_K(R)\sqrt{K}/\log_2 \sqrt{\frac{K}{R(1-R)}})(1+o(1))},$$

where $c_K(R) = -\log_2(1-R)\sqrt{\frac{1-R}{R}}$. So in order to maximize $C_G(K, R)$ for a given K we need to maximize $c_K(R)\sqrt{K}/\log_2 \sqrt{\frac{K}{R(1-R)}}$ for this K. By taking a derivative for R we have the following equation for obtaining the point of maximum:

$$c'_K(R)\frac{K_s}{\log_2 \frac{K_s}{\sqrt{(1-R)R}}} + c_K(R)\frac{K_s(1-2R)}{2\log_2 2\frac{K_s}{\sqrt{(1-R)R}}R(1-R)} = 0,$$

where $K_s = \sqrt{K}$. This simplifies to

$$c'_K(R) + c_K(R)\frac{1-2R}{2\log_2 \frac{K_s}{\sqrt{(1-R)R}}R(1-R)} = 0.$$

Considering that $K_s \to \infty$ for $K \to \infty$, the second summand tends to 0, and we are left with the equation

$$c'_K(R) = 0.$$

The solution of the above equation is a root of the equation

$$\frac{\ln(1-R)}{R} = -2$$

and numerically this root is $R^* \approx 0.8$.

Now let us show that, given the fact that the maximum complexity for the given key size is attained at some R^*, the minimum key size for the given security level is achieved for a code with the same rate R^*.

Proposition 1. *Let the security level S^* be given. Let $C(K, R)$ be the complexity of an algorithm A for decoding up to half the minimum distance a Goppa code with the key size K and rate R. We impose the following formal assumptions on $C(K, R)$:*

(a) $C(K, R)$ is continuous on $]0, \infty[\times[0, 1]$.
(b) C is unbounded in K for all R: $\forall R \in]0, 1[: C(K, R) \to \infty, K \to \infty$.
(c) C is increasing in K: $\forall K_2 > K_1 > 0 \ \forall R \in]0, 1[: C(K_1, R) < C(K_2, R)$.

Further, assume that for given K the maximum complexity of A is achieved at R^*:

(d) $\forall K > 0 \ \forall R \neq R^* : C^*(K) := C(K, R^*) > C(K, R)$.

Then the McEliece cryptosystem that satisfies the security level S^* w.r.t A with the smallest possible key size has an underlying Goppa code of rate R^*.

Proof. Due to (a) the function $C^*(K)$ is continuous and due to (c) is strictly increasing. Now because of (b) there exists a solution to $C^*(K) = S^*$. And because of the above mentioned properties of C^* this solution is unique: $C^*(K^*) = S^*$. Finally, the claim of the proposition follows from $S^* = C(K^*, R^*) > C(K^*, R) \ \forall R \neq R^*$.

Remark 2. Conditions (a)-(c) are natural for any complexity function of a decoding algorithm. The property (d) is true at least for ISD-like algorithms as we have seen in Lemma 1.

So now we may expect the following to happen in our table (Table 2): Although we use more advanced lower bounds from [9] we still expect that for given K the maximum security will be achieved at some R^*, the same for all K. As we have mentioned this is due to the fact that the improvements of the ISD algorithm do not seem to improve much on the exponential part. Moreover, because of the same reason we expect this R^* not to differ significantly from the value 0.8 predicted by Lemma 1. Having this, we will then use Proposition 1 to construct an algorithm that with arbitrary precision finds an instance with the smallest key possible that achieves the given security level S. This algorithm is depicted below (see Algorithm 1). In this algorithm, the value of S is calculated via the inequality (4), the interval $[R_{start}, R_{end}]$ is chosen large enough and contains 0.8, so we take an information rate which ranges from $R_{start} = 0.6$ to $R_{end} = 0.85$. All other parameters are chosen so that it is feasible to complete the algorithm in a reasonable time. For the key size, we set $K_{upper} = 200$ kB as an upper bound and use the step size $K_{step} = 1$ kB. Moreover, we use the lower bound formula from [9] as a function C.

4.2 Proposed Parameters

Our results are presented in Table 2 which shows the following information:

- Year: the year until which data security is required. Historic data is given mainly to allow comparison with other sources.
- Symmetric key size: the symmetric key size required to ensure data security, calculated in accordance with Lenstra and Verheul's approach.
- Lower bound for $\log_2(S(n, k, t))$: the $\log_2$ of the minimum number of binary operations (required to break a McEliece cryptosystem) that are infeasible in the respective year.
- RSA and EC parameters: the original parameters proposed by Lenstra and Verheul. They allow for easy comparison between "classical" and code-based cryptosystems.
- The last two columns are a translation of the required symmetric key size into parameters relevant in practice, i.e. the number of MIPS years that render a cryptosystem infeasible to break, and the corresponding number of years on a modern Quad core CPU.

Table 2. Proposed parameters for the McEliece cryptosystem – optimized for public key size

Year	Sym-metric Key Size	Lower bound for $\log_2 S(n,k,t)$	McEliece para-meters (n,k,t) and public key size (kB)	RSA Key Size and SDL[a] Field Size	Elliptic Curve Key Size (bits) for $c=0$	$c=18$	Infeasible number of MIPS-years	Corresponding number of years on a 2.4 GHz Intel Core 2 Quad Q6600
2009	77	83	(1634, 1217, 39) 62	1323 1024	145	157	$8.52 \cdot 10^{11}$	$1.94 \cdot 10^{7}$
2010	78	84	(1635, 1197, 41) 64	1369 1056	146	160	$1.45 \cdot 10^{12}$	$3.30 \cdot 10^{7}$
2011	79	85	(1652, 1203, 42) 66	1416 1088	148	163	$2.47 \cdot 10^{12}$	$5.61 \cdot 10^{7}$
2012	80	87	(1687, 1226, 43) 69	1464 1120	149	165	$4.19 \cdot 10^{12}$	$9.52 \cdot 10^{7}$
2013	80	88	(1702, 1219, 45) 72	1513 1184	151	168	$7.14 \cdot 10^{12}$	$1.62 \cdot 10^{8}$
2014	81	89	(1770, 1306, 43) 74	1562 1216	152	172	$1.21 \cdot 10^{13}$	$2.75 \cdot 10^{8}$
2015	82	90	(1823, 1368, 42) 76	1613 1248	154	173	$2.07 \cdot 10^{13}$	$4.70 \cdot 10^{8}$
2016	83	91	(1833, 1356, 44) 79	1664 1312	155	177	$3.51 \cdot 10^{13}$	$7.98 \cdot 10^{8}$
2017	83	92	(1845, 1356, 45) 81	1717 1344	157	180	$5.98 \cdot 10^{13}$	$1.36 \cdot 10^{9}$
2018	84	93	(1877, 1387, 45) 83	1771 1376	158	181	$1.02 \cdot 10^{14}$	$2.32 \cdot 10^{9}$
2019	85	95	(1951, 1481, 43) 85	1825 1440	160	185	$1.73 \cdot 10^{14}$	$3.93 \cdot 10^{9}$
2020	86	96	(1955, 1463, 45) 88	1881 1472	161	188	$2.94 \cdot 10^{14}$	$6.68 \cdot 10^{9}$
2021	86	97	(1983, 1479, 46) 91	1937 1536	163	190	$5.01 \cdot 10^{14}$	$1.14 \cdot 10^{10}$
2022	87	98	(2013, 1508, 46) 93	1995 1568	164	193	$8.52 \cdot 10^{14}$	$1.94 \cdot 10^{10}$
2023	88	99	(2018, 1491, 48) 96	2054 1632	166	197	$1.45 \cdot 10^{15}$	$3.30 \cdot 10^{10}$
2024	89	101	(2104, 1596, 46) 99	2113 1696	167	198	$2.47 \cdot 10^{15}$	$5.61 \cdot 10^{10}$
2025	89	102	(2106, 1576, 48) 102	2174 1728	169	202	$4.20 \cdot 10^{15}$	$9.55 \cdot 10^{10}$
2026	90	103	(2135, 1604, 48) 104	2236 1792	170	205	$7.14 \cdot 10^{15}$	$1.62 \cdot 10^{11}$
2027	91	104	(2157, 1614, 49) 107	2299 1856	172	207	$1.21 \cdot 10^{16}$	$2.75 \cdot 10^{11}$
2028	92	105	(2198, 1654, 49) 110	2362 1888	173	210	$2.07 \cdot 10^{16}$	$4.70 \cdot 10^{11}$
2029	93	106	(2220, 1664, 50) 113	2427 1952	175	213	$3.52 \cdot 10^{16}$	$8.00 \cdot 10^{11}$
2030	93	108	(2241, 1673, 51) 116	2493 2016	176	215	$5.98 \cdot 10^{16}$	$1.36 \cdot 10^{12}$
2032	95	110	(2344, 1784, 50) 122	2629 2144	179	222	$1.73 \cdot 10^{17}$	$3.93 \cdot 10^{12}$
2034	96	112	(2440, 1877, 50) 129	2768 2272	182	227	$5.01 \cdot 10^{17}$	$1.14 \cdot 10^{13}$
2036	98	115	(2496, 1920, 51) 135	2912 2400	185	232	$1.45 \cdot 10^{18}$	$3.30 \cdot 10^{13}$
2038	99	117	(2440, 1776, 59) 144	3061 2528	188	239	$4.20 \cdot 10^{18}$	$9.55 \cdot 10^{13}$
2040	101	119	(2521, 1854, 59) 151	3214 2656	191	244	$1.22 \cdot 10^{19}$	$2.77 \cdot 10^{14}$
2042	103	122	(2623, 1964, 58) 158	3371 2784	194	248	$3.52 \cdot 10^{19}$	$8.00 \cdot 10^{14}$
2044	104	124	(2662, 1979, 60) 165	3533 2944	197	255	$1.02 \cdot 10^{20}$	$2.32 \cdot 10^{15}$
2046	106	126	(2691, 1973, 63) 173	3700 3072	200	260	$2.95 \cdot 10^{20}$	$6.70 \cdot 10^{15}$
2048	107	129	(2798, 2088, 62) 181	3871 3232	203	265	$8.53 \cdot 10^{20}$	$1.94 \cdot 10^{16}$
2050	109	131	(2804, 2048, 66) 189	4047 3392	206	272	$2.47 \cdot 10^{21}$	$5.61 \cdot 10^{16}$

[a] Subgroup Discrete Logarithm

Algorithm 1 Search$(S, C, K_{step}, K_{upper}, R_{start}, R_{end})$

Require:
- A security level S
- The complexity function $C(n, R)$ satisfying (a)-(d) of Proposition 1
- Step for the key size search K_{step}
- Search upper bound for the key size K_{upper}
- Rate search interval bounds R_{start}, R_{end}

Ensure: n_{out} and R_{out} such that
- $C(n_{out}, R_{out}) \geq S$
- The key size is the smallest possible up to steps K_{step} and R_{step}

Begin
for $K = K_{step}$ to K_{upper} **do**
$n := \lfloor \sqrt{\frac{K}{R_{start}(1-R_{start})}} \rfloor$
while $(k < n \cdot R_{end})$ **do**
$R := \frac{k}{n}$
if $C(n, R) \geq S$ **then**
return n and R
end if
$k := k + 1$
$n := \lfloor \frac{K}{k} + k \rfloor$
end while
end for
return "NO solution found"
End

5 Conclusion

In this work we have addressed the problem of selecting optimal parameters for the McEliece cryptosystem based on binary Goppa codes. This problem was to find instances of McEliece cryptosystem that are secure in a given year and providing the smallest key sizes. For this problem, we have presented detailed parameter recommendations. This allows (potential) users of the McEliece cryptosystem to optimize the parameter choice, thereby improving the applicability of code-based cryptography. We have also shown the fact that all such optimal instances have a certain rate close to 0.74.

As a next step, we suggest a comprehensive analysis of concrete application scenarios. As we have illustrated above, in these scenarios constraints, as well as the trade-offs between the code properties, strongly depend on the details of the application, e.g. available bandwidth, acceptable response times, or (typical) message size. This analysis would provide further insights into the current strengths and limitations of code-based cryptography, thereby also suggesting new research foci for the future.

References

1. C.M. Adams and H. Meijer. Security-related Comments Regarding McEliece Public-key Cryptosystem. *IEEE Trans. Inform. Theory*, 35(2):454–455, 1989.
2. Au-Ja.de. Intel core 2 Quad Q6600, May 2007. http://www.au-ja.de/review-core2quad6600-5.phtml.
3. E. Berlekamp, R. McEliece, and H. van Tilborg. On the Inherent Intractability of Certain Coding Problems. *IEEE Trans. Inform. Theory*, 24(3):384–386, 1978.
4. D. J. Bernstein, T. Lange, and C. Peters. Attacking and defending the McEliece cryptosystem. In J. Buchmann and J. Ding, editors, *PQCrypto*, volume 5299 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2008.
5. A. Canteaut and H. Chabanne. A further improvement of the work factor in an attempt at breaking McEliece's cryptosystem. Research Report RR-2227, INRIA, 1994.

6. A. Canteaut and F. Chabaud. A New Algorithm for Finding Minimum-Weight Words in a Linear Code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.
7. R. Overbeck D. Engelbert and A. Schmidt. A Summary of McEliece-Type Cryptosystems and their Security. Cryptology ePrint Archive, Report 2006/162, 2006. `http://eprint.iacr.org/`.
8. C. Peters D. J. Bernstein, T. Lange and H. van Tilborg. Explicit Bounds for Generic Decoding Algorithms for Code-based Cryptography. In *Proc. of the International Workshop on Coding and Cryptography, WCC'2009*, pages 68–180, 2009.
9. M. Finiasz and N. Sendrier. Security Bounds for the Design of Code-based Cryptosystems. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, number 5912 in LNCS, pages 88–105. Springer, 2009.
10. M. P. C. Fossorier, K. Kobara, and H. Imai. Modeling Bit Flipping Decoding Based on Nonorthogonal Check Sums With Application to Iterative Decoding Attack of McEliece Cryptosystem. *IEEE Transactions on Information Theory*, 53(1):402–411, 2007.
11. A. Kh. Al Jabri. A Statistical Decoding Algorithm for General Linear Block Codes. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, pages 1–8. Springer-Verlag, 2001.
12. P.J. Lee and E.F. Brickell. An Observation on the Security of McEliece's Public-key Cryptosystem. In *EUROCRYPT '88, Lect. Notes in CS*, pages 275–280, 1988.
13. A. K. Lenstra and Eric R. Verheul. Selecting Cryptographic Key Sizes. *Journal of Cryptology: the journal of the International Association for Cryptologic Research*, 14(4):255–293, 2001.
14. F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error Correcting Codes*. North-Holland, 1977.
15. R.J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DNS Progress Report*, pages 114–116, 1978.
16. R. Overbeck. Statistical Decoding Revisited. In *ACISP*, pages 283–294, 2006.
17. R. Overbeck and N. Sendrier. Code-based Cryptography. In *Post Quantum Cryptography*, pages 95–146. Springer-Verlag, 2008.
18. N. Sendrier. On the Security of the McEliece Public-key Cryptosystem. In M. Blaum, P.G. Farrell, and H. van Tilborg, editors, *Information, Coding and Mathematics*, pages 141–163. Kluwer, 2002. Proceedings of Workshop honoring Prof. Bob McEliece on his 60th birthday.
19. P. W. Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *SFCS '94: Proc. of the 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. IEEE Computer Society, 1994.
20. J. Stern. A method for finding codewords of small weight. In *Proc. of Coding Theory and Applications*, pages 106–113, 1989.