# Wild McEliece

Daniel J. Bernstein[1], Tanja Lange[2], and Christiane Peters[2]

[1] Department of Computer Science
University of Illinois at Chicago, Chicago, IL 60607–7045, USA
djb@cr.yp.to
[2] Department of Mathematics and Computer Science
Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, Netherlands
tanja@hyperelliptic.org, c.p.peters@tue.nl

**Abstract.** The original McEliece cryptosystem uses length-$n$ codes over $\mathbf{F}_2$ with dimension $\geq n - mt$ efficiently correcting $t$ errors where $2^m \geq n$. This paper presents a generalized cryptosystem that uses length-$n$ codes over small finite fields $\mathbf{F}_q$ with dimension $\geq n - m(q-1)t$ efficiently correcting $\lfloor qt/2 \rfloor$ errors where $q^m \geq n$. Previously proposed cryptosystems with the same length and dimension corrected only $\lfloor (q-1)t/2 \rfloor$ errors for $q \geq 3$. This paper also presents list-decoding algorithms that efficiently correct even more errors for the same codes over $\mathbf{F}_q$. Finally, this paper shows that the increase from $\lfloor (q-1)t/2 \rfloor$ errors to more than $\lfloor qt/2 \rfloor$ errors allows considerably smaller keys to achieve the same security level against all known attacks.

**Keywords:** McEliece cryptosystem, Niederreiter cryptosystem, Goppa codes, wild Goppa codes, list decoding.

## 1 Introduction

Code-based cryptography was proposed in 1978 by McEliece [28] and is one of the oldest public-key cryptosystems. Code-based cryptography has lately received a lot of attention because it is a good candidate for public-key cryptography that remains secure against attacks by a quantum computer. See Overbeck and Sendrier [32] for a detailed overview of the state of the art; see also Bernstein [3] for the fastest known quantum attack.

Encryption in McEliece's system is very efficient (a matrix-vector multiplication) and thanks to Patterson's algorithm [33] decryption is also efficient. However, this system is rarely used in implementations. The main complaint is that the public key is too large.

Obviously, in the post-quantum setting, some secure public-key cryptosystem is better than none, and so one can tolerate the large key sizes. However,

convincing users to already now switch over to code-based systems requires shorter keys.

McEliece's original system uses binary Goppa codes. Several smaller-key variants have been proposed using other codes, such as Reed–Solomon codes [31], generalized Reed–Solomon codes [38], quasi-dyadic codes [30] or geometric Goppa codes [22]. Unfortunately, many specific proposals turned out to be breakable.

The most confidence-inspiring proposal is still McEliece's original proposal to use binary Goppa codes. For these only information-set-decoding attacks apply; these are generic attacks that work against any code-based cryptosystem. In 2008, Bernstein, Lange, and Peters [7] ran a highly optimized information-set-decoding attack to break the specific parameters proposed by McEliece in 1978. After 30 years the system had lost little of its strength; the break would not have been possible with the computation power available in 1978.

The best defense against this attack is to use codes with a larger error-correcting capability. Slightly larger binary Goppa codes are still unbreakable by any algorithm known today.

The disadvantage of binary Goppa codes is that they have a comparably large key size. The construction of the code is over $\mathbf{F}_{2^m}$ but then only codewords with entries in $\mathbf{F}_2$ are considered. Doing a similar construction with $\mathbf{F}_q$, for a prime power $q > 2$, as base field decreases the key size at the same security level against information-set decoding, as shown by Peters [34]. However, this effect appears only with sufficiently big base fields such as $\mathbf{F}_{31}$; codes over $\mathbf{F}_3$ and $\mathbf{F}_4$ look worse than those over $\mathbf{F}_2$. The main reason making $\mathbf{F}_2$ better is that for binary Goppa codes it is well known that the subfield construction almost doubles the error-correcting capability of the code (more precisely, of known fast decoding algorithms), improving the security of the resulting scheme. For codes over other fields no increase in the error-correcting capability was used in the estimates.

In this paper we propose using "wild Goppa codes". These are subfield codes over small $\mathbf{F}_q$ that have an increase in error-correcting capability by a factor of about $q/(q-1)$. McEliece's construction using binary Goppa codes is the special case $q = 2$ of our construction.

These codes were analyzed in 1976 by Sugiyama, Kasahara, Hirasawa, and Namekawa [41] but have not been used in code-based cryptography so far. We explain how to use these codes in the McEliece cryptosystem and how to correct $\lfloor qt/2 \rfloor$ errors where previous proposals corrected only $\lfloor (q-1)t/2 \rfloor$ errors. We also present a list-decoding algorithm that allows even more errors.

In the following sections we give the mathematical background of our proposal and explain where the increase in error-correcting capability comes from. After reviewing structural attacks and their applicability to our proposal we present parameters for different base fields that achieve 128-bit security against information-set-decoding attacks. These show that base fields $\mathbf{F}_q$ with $q \leq 32$ are interesting alternatives to $\mathbf{F}_2$. For $\mathbf{F}_{32}$ the increase factor $q/(q-1)$ is close to 1 and so our results are close to the results of Peters; but for $q = 3, 4$, or 5 the change is significant. Using list decoding further decreases the size of the key and leads to the smallest public keys proposed for subfield Goppa codes.

## 2   The McEliece Cryptosystem

This section gives background on the McEliece cryptosystem in two variants: the classical setup by McEliece as in [28] and Niederreiter's variant [31].

**Codes.** A linear code of length $n$ and dimension $k$ over $\mathbf{F}_q$ is a $k$-dimensional subspace of $\mathbf{F}_q^n$. Such a code $C$ can be represented (usually in many ways) by a *generator matrix*, a $k \times n$ matrix $G$ such that $C = \{mG : m \in \mathbf{F}_q^k\}$; or by a *parity-check matrix*, an $(n-k) \times n$ matrix $H$ such that $C = \{c \in \mathbf{F}_q^n : Hc^t = 0\}$.

Given a generator matrix $G$ for a linear code $C$ one can easily determine a parity-check matrix $H$ for $C$ by linear transformations. In particular, if $G$ has *systematic form*, i.e., $G = (I_k|Q)$ where $Q$ is a $k \times (n - k)$ matrix, then $H = (-Q^t|I_{n-k})$ is a parity-check matrix for the code $\mathbf{F}_q^k G$.

The *Hamming distance* between two words in $\mathbf{F}_q^n$ is the number of coordinates where they differ. The *Hamming weight* of a word is the number of nonzero coordinates in the word. The *minimum distance* of a nonzero linear code $C$ is the smallest Hamming weight of a nonzero codeword in $C$.

A *decoding algorithm* for $C$ receives a vector $y$ in $\mathbf{F}_q^n$ and a positive integer $w$ as inputs. The output is a codeword $c$ in $C$ at distance at most $w$ from $y$ if such $c$ exists. The linear codes that are interesting candidates for the McEliece cryptosystem are codes allowing fast error correction, i.e. fast computation of an error vector $e$ of weight $\leq w$ such that $y - e$ lies in $C$.

**The McEliece public-key cryptosystem.** Choose a linear code $C$ over $\mathbf{F}_q$ of length $n$ and dimension $k$ which can correct $w$ errors. Take a generator matrix $G$ for $C$. Also choose uniformly at random an $n \times n$ permutation matrix $P$ and an invertible $k \times k$ matrix $S$. Compute the matrix $\hat{G} = SGP$ and publish $\hat{G}$ together with the parameters $n$, $k$, and $w$. Make sure to keep $G$, $P$, and $S$ as well as $C$ secret.

Messages suitable for encryption are messages $m \in \mathbf{F}_q^k$. Encryption works as follows: Compute $m\hat{G}$. Compute a random error vector $e$ of weight $w$. Send $y = m\hat{G} + e$.

Decryption: Compute $yP^{-1} = mSG + eP^{-1}$. Apply $C$'s decoding algorithm to find $mSG$ which is a codeword in $C$ from which one obtains the original message $m$.

**The Niederreiter public-key cryptosystem.** Choose $C$ as above. Take a parity-check matrix $H$ of $C$. Choose a random $n \times n$ permutation matrix $P$ and a random invertible $(n - k) \times (n - k)$ matrix $M$. Publish the matrix $\hat{H} = MHP$ and the error weight $w$. Again keep the code and the matrices $H$, $P$, and $M$ secret.

Messages suitable for encryption are vectors $u \in \mathbf{F}_q^n$ of Hamming weight $w$. Encryption works as follows: Encrypt $u$ by multiplication with $\hat{H}$. Send $y = \hat{H}u^t$.

Decryption: Compute $v$ in $\mathbf{F}_q^n$ with $M^{-1}y = Hv^t$ by linear algebra. Note that $v^t - Pu^t$ lies in the kernel of $H$, i.e. is a codeword in $C$. Use the decoding algorithm to retrieve $v^t - Pu^t$, and since $v$ is known get $Pu^t$. Inverting $P$ yields $u$.

**Choice of codes.** Niederreiter proposed his system using generalized Reed–Solomon codes (GRS codes) whereas McEliece proposed to use classical binary

Goppa codes. The use of GRS codes was proven to be insecure in [38]. However, Niederreiter's system with binary Goppa codes has the same security as the McEliece cryptosystem as shown in [26].

## 3   Goppa Codes

This section gives an introduction to classical Goppa codes over $\mathbf{F}_q$.

Fix a prime power $q$; a positive integer $m$; a positive integer $n \leq q^m$; an integer $t < n/m$; distinct elements $a_1, \ldots, a_n$ in $\mathbf{F}_{q^m}$; and a polynomial $g(x)$ in $\mathbf{F}_{q^m}[x]$ of degree $t$ such that $g(a_i) \neq 0$ for all $i$.

The words $c = (c_1, \ldots, c_n)$ in $\mathbf{F}_{q^m}^n$ with

$$\sum_{i=1}^{n} \frac{c_i}{x - a_i} \equiv 0 \pmod{g(x)} \tag{3.1}$$

form a linear code $\Gamma_{q^m}(a_1, \ldots, a_n, g)$ of length $n$ and dimension $n - t$ over $\mathbf{F}_{q^m}$. The *Goppa code* $\Gamma_q(a_1, \ldots, a_n, g)$ with *Goppa polynomial* $g(x)$ and *support* $a_1, \ldots, a_n$ is the restriction of $\Gamma_{q^m}(a_1, \ldots, a_n, g)$ to the field $\mathbf{F}_q$, i.e., the set of elements $(c_1, \ldots, c_n)$ in $\mathbf{F}_q^n$ that satisfy (3.1). As a subfield subcode of $\Gamma_{q^m}(a_1, \ldots, a_n, g)$ the code $\Gamma_q(a_1, \ldots, a_n, g)$ has dimension $\geq n - mt$. Beware that there is a conflicting definition of "support" elsewhere in coding theory.

Let $\Gamma_q(a_1, \ldots, a_n, g)$ be a Goppa code of length $n$, support $a_1, \ldots, a_n$, and Goppa polynomial $g$ of degree $t$. Assume that $\Gamma_q(a_1, \ldots, a_n, g)$ has dimension exactly $n - mt$. Fix a basis of $\mathbf{F}_{q^m}$ over $\mathbf{F}_q$ and write each element of $\mathbf{F}_{q^m}$ with respect to that basis. Then a parity-check matrix for $\Gamma_q(a_1, \ldots, a_n, g)$ is given by the $mt \times n$ matrix

$$H = \begin{pmatrix} \frac{1}{g(a_1)} & \frac{1}{g(a_2)} & \cdots & \frac{1}{g(a_n)} \\ \frac{a_1}{g(a_1)} & \frac{a_2}{g(a_2)} & \cdots & \frac{a_n}{g(a_n)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{a_1^{t-1}}{g(a_1)} & \frac{a_2^{t-1}}{g(a_2)} & \cdots & \frac{a_n^{t-1}}{g(a_n)} \end{pmatrix},$$

over $\mathbf{F}_q$ where each entry is actually a column vector written in the chosen $\mathbf{F}_q$-basis of $\mathbf{F}_{q^m}$.

The code $\Gamma_q(a_1, \ldots, a_n, g)$ is often referred to as a "classical" Goppa code since it is the basic construction of a genus-0 geometric Goppa code which Goppa later generalized for higher-genus varieties.

For the decoding algorithm in Section 5 it is useful to recall that the codewords in $\Gamma_{q^m}(a_1, \ldots, a_n, g)$ can be constructed by evaluating certain functions at $a_1, \ldots, a_n$. Specifically: Define $h(x) = \prod_i (x - a_i)$. Note that $g(x)$ and $h(x)$ are coprime. For each $f \in g\mathbf{F}_{q^m}[x]$ define

$$\mathrm{ev}(f) = \left( \frac{f(a_1)}{h'(a_1)}, \frac{f(a_2)}{h'(a_2)}, \ldots, \frac{f(a_n)}{h'(a_n)} \right),$$

where $h'$ denotes the derivative of $h$.

If $f$ has degree less than $n$ then one can recover it from the the entries of $\text{ev}(f)$ by Lagrange interpolation: namely, $f/h = \sum_i (f(a_i)/h'(a_i))/(x - a_i)$. Consequently $\sum_i (\text{ev}(f))_i/(x - a_i)$ is 0 in $\mathbf{F}_{q^m}[x]/g$, where $(\text{ev}(f))_i$ denotes the $i$-th entry of $\text{ev}(f)$.

Let $(c_1, \ldots, c_n)$ in $\mathbf{F}_{q^m}^n$ be such that $\sum_i c_i/(x - a_i) \equiv 0 \pmod{g(x)}$. Define $f = \sum_i c_i h/(x - a_i)$ in $\mathbf{F}_{q^m}[x]$. Then $f \in g\mathbf{F}_{q^m}[x]$. Since the polynomial $\sum_i c_i h/(x - a_i)$ has degree less than $n$, also $f$ has degree less than $n$. Moreover, $c_j = f(a_j)/h'(a_j) = \text{ev}(f)_j$.

Therefore $\Gamma_{q^m}(a_1, \ldots, a_n, g) = \{\text{ev}(f) : f \in g\mathbf{F}_{q^m}[x],\ \deg(f) < n\} = \{(f(a_1)/h'(a_1), \ldots, f(a_n)/h'(a_n)) : f \in g\mathbf{F}_{q^m}[x],\ \deg(f) < n\}$.

## 4    Wild McEliece

We propose using the McEliece cryptosystem, the Niederreiter cryptosystem, etc. with Goppa codes of the form $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$ where $g$ is an irreducible monic polynomial in $\mathbf{F}_{q^m}[x]$ of degree $t$. Note the exponent $q - 1$ in $g^{q-1}$. We refer to these codes as "wild Goppa codes" for reasons explained later in this section.

We further propose to use error vectors of weight $\lfloor qt/2 \rfloor$. The advantage of wild Goppa codes is that they allow us to efficiently correct $\lfloor qt/2 \rfloor$ errors (or slightly more with the help of list decoding). For $q \in \{3, 4, \ldots\}$ this is strikingly better than the performance of an irreducible polynomial of the same degree $(q - 1)t$, namely correcting $\lfloor (q - 1)t/2 \rfloor$ errors. This change does not hurt the code dimension: polynomials of the form $g^{q-1}$ produce codes of dimension at least $n - m(q - 1)t$ (and usually exactly $n - m(q - 1)t$), just like irreducible polynomials of degree $(q - 1)t$.

**Comparison to previous proposals.** For $q = 2$ this proposal is not new: it is exactly McEliece's original proposal to use a binary Goppa code $\Gamma_2(a_1, \ldots, a_n, g)$, where $g$ is an irreducible polynomial of degree $t$, and to use error vectors of weight $t$. McEliece used Patterson's algorithm to efficiently decode $t$ errors.

We also do not claim credit for considering Goppa codes over slightly larger fields $\mathbf{F}_3$, $\mathbf{F}_4$, etc. Peters in [34, Section 8] pointed out that switching from binary Goppa codes to codes of the form $\Gamma_{31}(a_1, \ldots, a_n, g)$, with $t/2$ errors, reduces the key size by a factor of more than 2 while preserving security against all known attacks.

What is new in our cryptosystem is the use of Goppa polynomials of the form $g^{q-1}$ for $q \geq 3$, allowing us to correct more errors for the same field size, the same code length, and the same code dimension.

**Minimum distance of wild Goppa codes.** The following theorem is the main theorem of the 1976 paper [41] by Sugiyama, Kasahara, Hirasawa, and Namekawa. What the theorem states is that, for any monic squarefree polynomial $g$ in $\mathbf{F}_{q^m}[x]$, the code $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$ is the same as $\Gamma_q(a_1, \ldots, a_n, g^q)$. The code therefore has minimum distance at least $qt + 1$. Efficient decoding of $\lfloor qt/2 \rfloor$ errors requires more effort and is discussed in the next section.

The case $q = 2$ of this theorem is due to Goppa, using a different proof that can be found in many textbooks. The case $q \geq 3$ has received less attention. We include a streamlined proof to keep this paper self-contained.

The proof immediately generalizes from the pair $(g^{q-1}, g^q)$ to the pair $(g^{rq-1}, g^{rq})$, and to coprime products of such pairs. These generalizations also appear in [41]. Wirtz in [44], and independently Katsman and Tsfasman in [23], further generalized the results of [41] to geometric Goppa codes. See Janwa and Moreno [22] for discussion of the possibility of using geometric Goppa codes in the McEliece cryptosystem but also Minder's thesis [29] and the paper by Faure and Minder [15] for attacks on the elliptic-curve version and the genus-2 version. We do not consider this possibility further in this paper.

**Theorem 4.1** *Let $q$ be a prime power. Let $m$ be a positive integer. Let $n$ be an integer with $1 \leq n \leq q^m$. Let $a_1, a_2, \ldots, a_n$ be distinct elements of $\mathbf{F}_{q^m}$. Let $g$ be a monic squarefree polynomial in $\mathbf{F}_{q^m}[x]$ coprime to $(x - a_1) \cdots (x - a_n)$. Then $\Gamma_q(a_1, a_2, \ldots, a_n, g^{q-1}) = \Gamma_q(a_1, a_2, \ldots, a_n, g^q)$.*

*Proof.* If $\sum_i c_i/(x - a_i) = 0$ in $\mathbf{F}_{q^m}[x]/g^q$ then certainly $\sum_i c_i/(x - a_i) = 0$ in $\mathbf{F}_{q^m}[x]/g^{q-1}$.

Conversely, consider any $(c_1, c_2, \ldots, c_n) \in \mathbf{F}_q^n$ such that $\sum_i c_i/(x - a_i) = 0$ in $\mathbf{F}_{q^m}[x]/g^{q-1}$. Find an extension $k$ of $\mathbf{F}_{q^m}$ so that $g$ splits into linear factors in $k[x]$. Then $\sum_i c_i/(x-a_i) = 0$ in $k[x]/g^{q-1}$, so $\sum_i c_i/(x-a_i) = 0$ in $k[x]/(x-r)^{q-1}$ for each factor $x - r$ of $g$. The elementary series expansion

$$\frac{1}{x - a_i} = -\frac{1}{a_i - r} - \frac{x - r}{(a_i - r)^2} - \frac{(x - r)^2}{(a_i - r)^3} - \cdots$$

then implies

$$\sum_i \frac{c_i}{a_i - r} + (x - r) \sum_i \frac{c_i}{(a_i - r)^2} + (x - r)^2 \sum_i \frac{c_i}{(a_i - r)^3} + \cdots = 0$$

in $k[x]/(x - r)^{q-1}$; i.e., $\sum_i c_i/(a_i - r) = 0$, $\sum_i c_i/(a_i - r)^2 = 0$, $\ldots$, $\sum_i c_i/(a_i - r)^{q-1} = 0$. Now take the $q$th power of the equation $\sum_i c_i/(a_i - r) = 0$, and use the fact that $c_i \in \mathbf{F}_q$, to obtain $\sum_i c_i/(a_i - r)^q = 0$. Work backwards to see that $\sum_i c_i/(x - a_i) = 0$ in $k[x]/(x - r)^q$.

By hypothesis $g$ is the product of its distinct linear factors $x - r$. Therefore $g^q$ is the product of the coprime polynomials $(x - r)^q$, and $\sum_i c_i/(x - a_i) = 0$ in $k[x]/g^q$; i.e., $\sum_i c_i/(x - a_i) = 0$ in $\mathbf{F}_{q^m}[x]/g^q$. $\square$

**The "wild" terminology.** To explain the name "wild Goppa codes" we briefly review the standard concept of wild ramification.

A prime $p$ "ramifies" in a number field $L$ if the unique factorization $p\mathcal{O}_L = Q_1^{e_1} Q_2^{e_2} \cdots$ has an exponent $e_i$ larger than 1, where $\mathcal{O}_L$ is the ring of integers of $L$ and $Q_1, Q_2, \ldots$ are distinct maximal ideals of $\mathcal{O}_L$. Each $Q_i$ with $e_i > 1$ is "ramified over $p$"; this ramification is "wild" if $e_i$ is divisible by $p$.

If $\mathcal{O}_L/p$ has the form $\mathbf{F}_p[x]/f$, where $f$ is a monic polynomial in $\mathbf{F}_p[x]$, then the maximal ideals $Q_1, Q_2, \ldots$ correspond naturally to the irreducible factors

of $f$, and the exponents $e_1, e_2, \ldots$ correspond naturally to the exponents in the factorization of $f$. In particular, the ramification corresponding to an irreducible factor of $f$ is wild if and only if the exponent is divisible by $p$.

Similar comments apply to more general extensions of global fields. Ramification corresponding to an irreducible factor $\varphi$ of a monic polynomial $f$ in $\mathbf{F}_{p^m}[x]$ is wild if and only if the exponent is divisible by $p$, i.e., the local component of $f$ is a power of $\varphi^p$. We take the small step of referring to $\varphi^p$ as being "wild", and referring to the corresponding Goppa codes as "wild Goppa codes". Of course, if the Goppa code for $\varphi^p$ is wild, then the Goppa code for $\varphi^{p-1}$ must also be wild, since (by Theorem 4.1) it is the same code.

The traditional concept of wild ramification is defined by the characteristic of the base field. We find it more useful to allow a change of base from $\mathbf{F}_p$ to $\mathbf{F}_q$, generalizing the definition of wildness to use the size of $\mathbf{F}_q$ rather than just the characteristic of $\mathbf{F}_q$.

## 5    Decrypting Wild-McEliece Ciphertexts

The main problem faced by a wild-McEliece receiver is to decode $\lfloor qt/2 \rfloor$ errors in the code $\Gamma = \Gamma_q(a_1, \ldots, a_n, g^{q-1})$: i.e., to find a codeword $c = (c_1, \ldots, c_n) \in \Gamma$, given a received word $y = (y_1, \ldots, y_n) \in \mathbf{F}_q^n$ at Hamming distance $\lfloor qt/2 \rfloor$ from $c$. This section presents an asymptotically fast algorithm that decodes $\lfloor qt/2 \rfloor$ errors, and then a "list decoding" algorithm that decodes even more errors.

**Classical decoding.** Recall from Theorem 4.1 that

$$
\begin{aligned}
\Gamma &= \Gamma_q(a_1, \ldots, a_n, g^q) \\
&\subseteq \Gamma_{q^m}(a_1, \ldots, a_n, g^q) \\
&= \left\{ \left( \frac{f(a_1)}{h'(a_1)}, \ldots, \frac{f(a_n)}{h'(a_n)} \right) : f \in g^q \mathbf{F}_{q^m}[x], \deg f < n \right\}
\end{aligned}
$$

where $h = (x-a_1) \cdots (x-a_n)$. We thus view the target codeword $c = (c_1, \ldots, c_n) \in \Gamma$ as a sequence $(f(a_1)/h'(a_1), \ldots, f(a_n)/h'(a_n))$ of function values, where $f$ is a multiple of $g^q$ of degree below $n$. We are given $y$, the same sequence with $\lfloor qt/2 \rfloor$ errors, or more generally with $\leq \lfloor qt/2 \rfloor$ errors. We reconstruct $c$ from $y$ as follows:

- Interpolate $y_1 h'(a_1)/g(a_1)^q, \ldots, y_n h'(a_n)/g(a_n)^q$ into a polynomial $\varphi$: i.e., construct the unique $\varphi \in \mathbf{F}_{q^m}[x]$ such that $\varphi(a_i) = y_i h'(a_i)/g(a_i)^q$ and $\deg \varphi < n$.
- Compute the continued fraction of $\varphi/h$ to degree $\lfloor qt/2 \rfloor$: i.e., apply the Euclidean algorithm to $h$ and $\varphi$, stopping with the first remainder $v_0 h - v_1 \varphi$ of degree $< n - \lfloor qt/2 \rfloor$.
- Compute $f = (\varphi - v_0 h/v_1) g^q$.
- Compute $c = (f(a_1)/h'(a_1), \ldots, f(a_n)/h'(a_n))$.

This algorithm uses $n^{1+o(1)}$ operations in $\mathbf{F}_{q^m}$ if multiplication, evaluation, interpolation, and continued-fraction computation are carried out by standard FFT-based subroutines; see [5] for a survey of those subroutines.

To see that this algorithm works, observe that $\varphi$ has many values in common with the target polynomial $f/g^q$: specifically, $\varphi(a_i)=f(a_i)/g(a_i)^q$ for all but $\lfloor qt/2 \rfloor$ values of $i$. In other words, the error-locator polynomial

$$\epsilon = \prod_{i:\frac{f(a_i)}{g(a_i)^q} \neq \varphi(a_i)} (x - a_i)$$

has degree at most $\lfloor qt/2 \rfloor$. The difference $\varphi - f/g^q$ is a multiple of $h/\epsilon$, say $\delta h/\epsilon$. Now the difference $\delta/\epsilon - \varphi/h = -(f/g^q)/h$ is smaller than $1/x^{qt}$ and therefore smaller than $1/\epsilon^2$, so $\delta/\epsilon$ is a "best approximation" to $\varphi/h$, so $\delta/\epsilon$ must appear as a convergent to the continued fraction of $\varphi/h$, specifically the convergent at degree $\lfloor qt/2 \rfloor$. Consequently $\delta/\epsilon = v_0/v_1$; i.e., $f/g^q = \varphi - v_0 h/v_1$.

More generally, one can use any Reed–Solomon decoder to reconstruct $f/g^q$ from the values $f(a_1)/g(a_1)^q, \ldots, f(a_n)/g(a_n)^q$ with $\lfloor qt/2 \rfloor$ errors. This is an illustration of the following sequence of standard transformations:

$$\text{Reed–Solomon decoder} \Rightarrow \text{generalized Reed–Solomon decoder}$$
$$\Rightarrow \text{alternant decoder} \Rightarrow \text{Goppa decoder.}$$

The resulting decoder corrects $\lfloor (\deg g)/2 \rfloor$ errors for general Goppa codes $\Gamma_q(a_1, \ldots, a_n, g)$; in particular, $\lfloor q(\deg g)/2 \rfloor$ errors for $\Gamma_q(a_1, \ldots, a_n, g^q)$; and so $\lfloor q(\deg g)/2 \rfloor$ errors for $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$, by Theorem 4.1.

We do not claim that the particular algorithm stated above is the fastest possible decoder, and in particular we do not claim that it is quite as fast as Patterson's algorithm [33] for $q = 2$. However, it has essentially the same scalability in $n$ as Patterson's algorithm, works for general $q$, and is obviously fast enough to be usable.

An example implementation of a wild-Goppa-code decoder in the Sage computer-algebra system [39] can be found at `http://pqcrypto.org/users/christiane/wild.html`.

**List decoding.** By switching from a classical Reed–Solomon decoding algorithm to the Guruswami–Sudan list-decoding algorithm [19] we can efficiently correct $n - \sqrt{n(n - qt)} > \lfloor qt/2 \rfloor$ errors in the function values $f(a_1)/g(a_1)^q, \ldots, f(a_n)/g(a_n)^q$. This algorithm is not as fast as a classical decoder but still takes polynomial time. Consequently we can handle $n - \sqrt{n(n - qt)}$ errors in the wild Goppa code $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$.

This algorithm can, at least in theory, produce several possible codewords $c$. This does not pose a problem for the CCA2-secure variants of the McEliece cryptosystem introduced by Kobara and Imai in [25]: those variants automatically reject all codewords that do not include proper labels cryptographically protected by an "all-or-nothing transform".

As above, we do not claim that this algorithm is the fastest possible decoder. In particular, for $q = 2$ the same error-correcting capacity was obtained by Bernstein in [4] using a more complicated algorithm, analogous to Patterson's algorithm; we do not claim that the $\Gamma(a_1, \ldots, a_n, g^2)$ approach is as fast as that algorithm.

With more decoding effort we can handle a few additional errors by the standard idea of combinatorially guessing those errors. Each additional error produces a noticeable reduction of key size, as shown later in this paper. In many applications, the McEliece decoding time is unnoticeable while the McEliece key size is a problem, so allowing extra errors at the expense of decoding time is a good tradeoff.

## 6   Attacks

This section discusses several attacks against the wild McEliece cryptosystem. All of the attacks scale poorly to large key sizes; Section 7 presents parameters that are safe against all of these attacks. We do not claim novelty for any of the attack ideas.

We emphasize that the wild McEliece cryptosystem includes, as a special case, the original McEliece cryptosystem. A complete break of the wild McEliece cryptosystem would therefore imply a complete break of the original McEliece cryptosystem, a system that has survived scrutiny for 32 years. It is of course possible that there is a magical dividing line between $q = 2$ and $q = 3$, an attack that breaks every new case of our proposal while leaving the original cryptosystem untouched, but we have not found any such line.

We focus on inversion attacks, i.e., attacks against the one-wayness of wild McEliece encryption. There are several well-known chosen-ciphertext attacks that break semantic security without breaking one-wayness, but all of those attacks are stopped by standard conversions; see [25].

**Information-set decoding.** The top threat against the original McEliece cryptosystem, the attack algorithm that has always dictated key-size recommendations, is information-set decoding, which as mentioned in the introduction is a generic decoding method that does not rely on any particular code structure. The same attack also appears to be the top threat against the wild McEliece cryptosystem for $\mathbf{F}_3$, $\mathbf{F}_4$, etc.

The exact complexity of information-set decoding is not easy to state concisely. We rely on, and refer the reader to, the recent analysis of state-of-the-art $\mathbf{F}_q$ information-set decoding by Peters in [34], combining various improvements from [40], [11], [7], and [16]. To find the parameters in Section 7 we searched various $(n, k, t)$ and applied the complexity formulas from [34] to evaluate the security level of each $(n, k, t)$.

**Generalized birthday attacks.** Wagner's "generalized birthday attacks" [43] can also be used as a generic decoding method. The Courtois–Finiasz–Sendrier signature system [13] was attacked by Bleichenbacher using this method. However, information-set decoding is always more efficient than generalized birthday attacks as an attack against code-based encryption. See [16] for further discussion; the analysis is essentially independent of $q$.

**Polynomial-searching attacks.** There are approximately $q^{mt}/t$ monic irreducible polynomials $g$ of degree $t$ in $\mathbf{F}_{q^m}[x]$, and therefore approximately $q^{mt}/t$

choices of $g^{q-1}$. One can marginally expand the space of polynomials by considering more general squarefree polynomials $g$, but we focus on irreducible polynomials to avoid any unnecessary security questions.

An attacker can try to guess the Goppa polynomial $g^{q-1}$ and then apply Sendrier's "support-splitting algorithm" [37] to compute the support $(a_1, \ldots, a_n)$. We combine two defenses against this attack:

- We keep $q^{mt}/t$ extremely large, so that guessing $g^{q-1}$ has negligible chance of success. Parameters with $q^{mt}/t$ smaller than $2^{128}$ are marked with the international biohazard symbol ☣ in Section 7.
- We keep $n$ noticeably lower than $q^m$, so that there are many possible subsets $\{a_1, \ldots, a_n\}$ of $\mathbf{F}_{q^m}$. The support-splitting algorithm takes $\{a_1, \ldots, a_n\}$ as an input along with $g$.

The second defense is unusual: it is traditional, although not universal, to take $n = 2^m$ and $q = 2$, so that the only possible set $\{a_1, \ldots, a_n\}$ is $\mathbf{F}_{2^m}$. The strength of the second defense is unclear: we might be the first to ask whether the support-splitting idea can be generalized to handle many sets $\{a_1, \ldots, a_n\}$ simultaneously, and we would not be surprised if the answer turns out to be yes. However, the first defense is well known for $q = 2$ and appears to be strong.

**Algebraic attacks.** In a recent paper [14], Faugère, Otmani, Perret, and Tillich broke many (but not all) of the "quasi-cyclic" and "quasi-dyadic" variants of the McEliece cryptosystem that had been proposed in the papers [2] and [30] in 2009. Gauthier Umana and Leander in [17] independently broke some of the same systems.

These variants have highly regular support structures allowing very short public keys. The attacks set up systems of low-degree algebraic equations for the code support, taking advantage of the fact that there are not many variables in the support.

The paper [14] indicates that the same attack strategy is of no use against the original McEliece cryptosystem because there are "much more unknowns" than in the broken proposals: for example, 1024 variables in $\mathbf{F}_{1024}$, totalling 10240 bits. Our recommended parameters also have very large supports, with no particular structure, so algebraic attacks do not appear to pose any threat.

## 7    Parameters

The public key in Kobara and Imai's CCA2-secure variant [25] of the McEliece cryptosystem can be stored in systematic form as $(n-k)k$ entries in $\mathbf{F}_q$. The same is true for the Niederreiter variant; see, e.g., [32, Algorithm 2.3]. The simplest representation of an element of $\mathbf{F}_q$ takes $\lceil \log_2 q \rceil$ bits (e.g., 3 bits for $q = 5$), but a sequence of elements can be compressed: one stores a batch of $b$ elements of $\mathbf{F}_q$ in $\lceil \log_2 q^b \rceil$ bits, at the expense of some easy computation to recover the individual elements. As $b$ grows the storage per field element drops to approximately $\log_2 q$ bits, so $(n-k)k$ elements can be stored using about $\lceil (n-k)k \log_2 q \rceil$ bits.
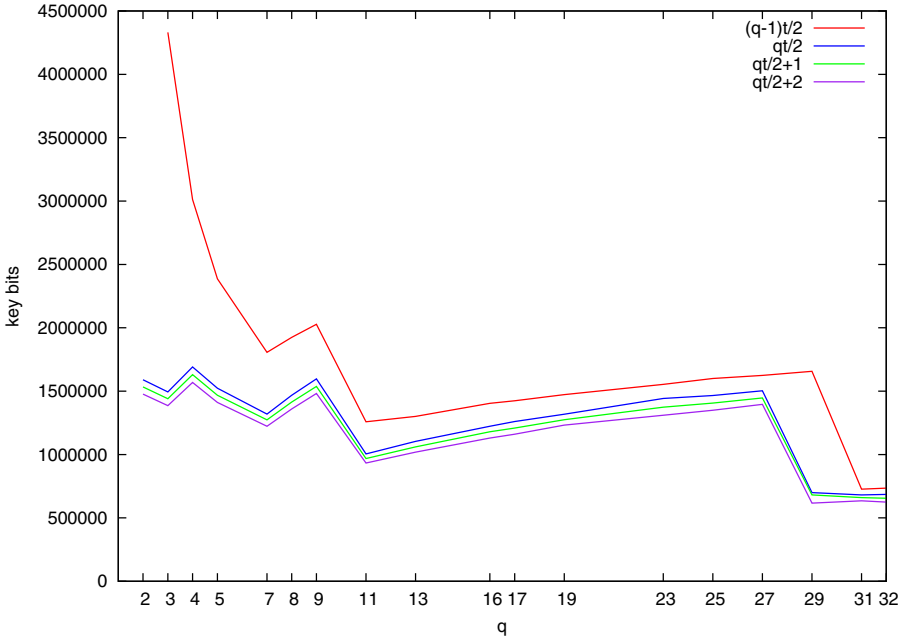
**Fig. 7.1.** Decrease in key sizes when correcting more errors (128-bit security). See Table 7.1.

Table 7.1 gives parameters $(n, k, t)$ for the McEliece cryptosystem using a code $\Gamma = \Gamma_q(a_1, \ldots, a_n, g^{q-1})$ that provides 128-bit security against the attack in [34]. We chose the code length $n$, the degree $t$ of $g$ and the dimension $k = n - \lceil \log_q n \rceil t(q-1)$ of $\Gamma$ to minimize the key size $\lceil (n-k)k \log_2 q \rceil$ for 128-bit security when $w$ errors are added. We compare four cases:

- $w = \lfloor (q-1)t/2 \rfloor$ added errors using classical decoding techniques,
- $w = \lfloor qt/2 \rfloor$ added errors using Theorem 4.1,
- $w = \lfloor qt/2 \rfloor + 1$ added errors, and
- $w = \lfloor qt/2 \rfloor + 2$ added errors,

where the last two cases use Theorem 4.1 together with list decoding as in Section 5. See Figure 7.1 for a graph of the resulting key sizes.

In [7] a Goppa code $\Gamma_2(a_1, \ldots, a_n, g)$ with length 2960, dimension 2288, and $g$ of degree $t = 56$ is proposed for 128-bit security when 57 errors are added by the sender. A key in this setting has 1537536 bits. This is consistent with our table entry for $q = 2$ with $w = \lfloor qt/2 \rfloor + 1$ added errors.

Small $q$'s larger than 2 provide astonishingly good results. For larger $q$'s one has to be careful: parameters optimized against information-set decoding have $q^{mt}/t$ dropping as $q$ grows, reducing the number of suitable polynomials $g$ in

$\mathbf{F}_{q^m}[x]$ significantly. For example, there are only about $2^{28}$ monic irreducible polynomials $g$ of degree 3 over $\mathbf{F}_{31^2}[x]$, while there are about $2^{227}$ monic irreducible polynomials $g$ of degree 20 in $\mathbf{F}_{5^5}[x]$. The smallest $q$ for which the $g$ possibilities can be enumerated in less time than information-set decoding is $q = 11$: the parameters $(n, k, t) = (1199, 899, 10)$ satisfy $q^{\lceil \log_q n \rceil t}/t \approx 2^{100}$, so there are about $2^{100}$ monic irreducible polynomials $g$ in $\mathbf{F}_{11^3}[x]$ of degree $t = 10$. This is one of the cases marked by ✿ in Table 7.1. The security of these cases depends on the strength of the second defense discussed in Section 6.

The ✿ symbol is omitted from the $\lfloor (q-1)t/2 \rfloor$ column because that relatively low error-correcting capability, and relatively high key size, can be achieved by non-wild codes with many more choices of $g$.

**Table 7.1.** Decrease in key sizes when correcting more errors (128-bit security). Each entry in the first column states $q$. Each entry in the subsequent columns states key size, $(n, k, t)$ and the number of errors.

| $q$ | $\lfloor (q-1)t/2 \rfloor$ | $\lfloor qt/2 \rfloor$ | $\lfloor qt/2 \rfloor + 1$ | $\lfloor qt/2 \rfloor + 2$ |
|---|---|---|---|---|
| 2 | — | 1590300 bits: (3009, 2325, 57) 57 errors | 1533840 bits: (2984, 2324, 55) 56 errors | 1477008 bits: (2991, 2367, 52) 54 errors |
| 3 | 4331386 bits: (3946, 3050, 56) 56 errors | 1493796 bits: (2146, 1530, 44) 66 errors | 1439876 bits: (2133, 1545, 42) 64 errors | 1385511 bits: (2121, 1561, 40) 62 errors |
| 4 | 3012336 bits: (2886, 2202, 38) errors 57 | 1691424 bits: (2182, 1678, 28) errors 56 | 1630044 bits: (2163, 1677, 27) 55 errors | 1568700 bits: (2193, 1743, 25) 52 errors |
| 5 | 2386014 bits: (2395, 1835, 28) 56 errors | 1523278 bits: (1931, 1491, 22) 55 errors | 1468109 bits: (1877, 1437, 22) 56 errors | 1410804 bits: (1919, 1519, 20) 52 errors |
| 7 | 1806298 bits: (1867, 1411, 19) 57 errors | 1319502 bits: (1608, 1224, 16) 56 errors | 1273147 bits: (1565, 1181, 16) 57 errors | 1223423 bits: (1633, 1297, 14) 51 errors |
| 8 | 1924608 bits: (1880, 1432, 16) 56 errors | 1467648 bits: (1640, 1248, 14) 56 errors | 1414140 bits: (1659, 1295, 13) 53 errors | 1359540 bits: (1609, 1245, 13) 54 errors |
| 9 | 2027941 bits: (1876, 1428, 14) 56 errors | 1597034 bits: (1696, 1312, 12) 54 errors | 1537389 bits: (1647, 1263, 12) 55 errors | 1481395 bits: (1601, 1217, 12) 56 errors |
| 11 | 1258265 bits: (1286, 866, 14) 70 errors | 1004619 bits: (1268, 968, 10)✿ 55 errors | 968295 bits: (1233, 933, 10)✿ 56 errors | 933009 bits: (1199, 899, 10)✿ 57 errors |
| 13 | 1300853 bits: (1409, 1085, 9) 54 errors | 1104093 bits: (1324, 1036, 8)✿ 52 errors | 1060399 bits: (1283, 995, 8)✿ 53 errors | 1018835 bits: (1244, 956, 8)✿ 54 errors |

**Table 7.1.** *(continued)*

| $q$ | $\lfloor (q-1)t/2 \rfloor$ | $\lfloor qt/2 \rfloor$ | $\lfloor qt/2 \rfloor + 1$ | $\lfloor qt/2 \rfloor + 2$ |
|---|---|---|---|---|
| 16 | 1404000 bits: (1335, 975, 8) 60 errors | 1223460 bits: (1286, 971, 7) 56 errors | 1179360 bits: (1251, 936, 7) 57 errors | 1129680 bits: (1316, 1046, 6) 50 errors |
| 17 | 1424203 bits: (1373, 1037, 7) 56 errors | 1260770 bits: (1359, 1071, 6) 51 errors | 1208974 bits: (1315, 1027, 6) 52 errors | 1160709 bits: (1274, 986, 6) 53 errors |
| 19 | 1472672 bits: (1394, 1070, 6) 54 errors | 1318523 bits: (1282, 958, 6) 57 errors | 1274481 bits: (1250, 926, 6) 58 errors | 1231815 bits: (1219, 895, 6) 59 errors |
| 23 | 1553980 bits: (1371, 1041, 5) 55 errors | 1442619 bits: (1472, 1208, 4) 46 errors | 1373354 bits: (1414, 1150, 4) 47 errors | 1310060 bits: (1361, 1097, 4) 48 errors |
| 25 | 1599902 bits: (1317, 957, 5) 60 errors | 1465824 bits: (1384, 1096, 4) 50 errors | 1405640 bits: (1339, 1051, 4) 51 errors | 1349468 bits: (1297, 1009, 4) 52 errors |
| 27 | 1624460 bits: (1407, 1095, 4) 52 errors | 1502811 bits: (1325, 1013, 4) 54 errors | 1446437 bits: (1287, 975, 4) 55 errors | 1395997 bits: (1253, 941, 4) 56 errors |
| 29 | 1656766 bits: (1351, 1015, 4) 56 errors | 699161 bits: (794, 514, 5) 72 errors | 681478 bits: (781, 501, 5) 73 errors | 617003 bits: (791, 567, 4) 60 errors |
| 31 | 726484 bits: (851, 611, 4) 60 errors | 681302 bits: (813, 573, 4) 62 errors | 659899 bits: (795, 555, 4) 63 errors | 634930 bits: (892, 712, 3) 48 errors |
| 32 | 735320 bits: (841, 593, 4) 62 errors | 685410 bits: (923, 737, 3) 48 errors | 654720 bits: (890, 704, 3) 49 errors | 624960 bits: (858, 672, 3) 50 errors |

# References

[1] —- (No Editor), Eleventh International Workshop on Algebraic and Combinatorial Coding Theory, Pamporovo, Bulgaria, June 16–22 (2008), http://www.moi.math.bas.bg/acct2008/acct2008.html, See [15]

[2] Berger, T.P., Cayrel, P.-L., Gaborit, P., Otmani, A.: Reducing Key Length of the McEliece Cryptosystem. In: AFRICACRYPT 2009 [35], pp. 77–97 (2009), Citations in This Document: §6

[3] Bernstein, D.J.: Grover vs. McEliece. In: PQCrypto 2010 [36], pp. 73–80 (2010), http://cr.yp.to/papers.html#grovercode, Citations in This Document: §1

[4] Bernstein, D.J.: List Decoding for Binary Goppa Codes (2008), http://cr.yp.to/papers.html#goppalist, Citations in This Document: §5

[5] Bernstein, D.J.: Fast Multiplication and Its Applications. In: Algorithmic Number Theory [10], pp. 325–384 (2008), `http://cr.yp.to/papers.html#multapps`, Citations in This Document: §5

[6] Bernstein, D.J., Buchmann, J., Dahmen, E. (eds.): Post-Quantum Cryptography. Springer, Heidelberg (2009) ISBN 978-3-540-88701-0, See [32]

[7] Bernstein, D.J., Lange, T., Peters, C.: Attacking and Defending the McEliece Cryptosystem. In: PQCrypto 2008 [9], pp. 31–46 (2008), `http://eprint.iacr.org/2008/318`, Citations in This Document: §1, §6, §7

[8] Boyd, C. (ed.): Advances in Cryptology — ASIACRYPT 2001, Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security Held on the Gold Coast, December 9-13, 2001. LNCS, vol. 2248. Springer, Heidelberg (2001) ISBN 3-540-42987-5, See [13]

[9] Buchmann, J., Ding, J. (eds.): Proceedings of Post-Quantum Cryptography, Second International Workshop, PQCrypto 2008, Cincinnati, OH, USA, October 17-19, 2008. LNCS, vol. 5299. Springer, Heidelberg (2008), See [7]

[10] Buhler, J., Stevenhagen, P. (eds.): Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography. Cambridge University Press, Cambridge (2008) ISBN 978-0521808545, See [5]

[11] Canteaut, A., Chabaud, F.: 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, 367–378 (1998), `http://hal.inria.fr/inria-00074006/en/`, MR 98m:94043, Citations in This Document: §6

[12] Cohen, G.D., Wolfmann, J. (eds.): Coding Theory and Applications. LNCS, vol. 388. Springer, Heidelberg (1989), See [40]

[13] Courtois, N., Finiasz, M., Sendrier, N.: How to Achieve a McEliece-Based Digital Signature Scheme. In: ASIACRYPT 2001 [8], pp. 157–174 (2001), `http://hal.inria.fr/docs/00/07/25/11/PDF/RR-4118.pdf`, MR 2003h:94028, Citations in This Document: §6

[14] Faugère, J.-C., Otmani, A., Perret, L., Tillich, J.-P.: Algebraic Cryptanalysis of McEliece Variants with Compact Keys. In: EUROCRYPT 2010 [18], pp. 279–298 (2010), Citations in This Document: §6, §6

[15] Faure, C., Minder, L.: Cryptanalysis of the McEliece Cryptosystem over Hyperelliptic Codes. In: ACCT 2008 [1], pp. 99–107 (2008), `http://www.moi.math.bas.bg/acct2008/b17.pdf` , Citations in This Document: §4

[16] Finiasz, M., Sendrier, N.: Security Bounds for the Design of Code-Based Cryptosystems. In: ASIACRYPT 2009 [27], pp. 88–105 (2009), `http://eprint.iacr.org/2009/414` , Citations in This Document: §6, §6

[17] Gauthier Umana, V., Leander, G.: Practical Key Recovery Attacks on Two McEliece Variants (2009), `http://eprint.iacr.org/2009/509`, Citations in This Document: §6

[18] Gilbert, H. (ed.): Proceedings of Advances in Cryptology — EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30-June 3, 2010. LNCS, vol. 6110. Springer, Heidelberg (2010), See [14]

[19] Guruswami, V., Sudan, M.: Improved Decoding of Reed-Solomon and Algebraic-Geometry Codes. IEEE Transactions on Information Theory 45, 1757–1767 (1999), `http://theory.lcs.mit.edu/~madhu/bib.html`, ISSN 0018–9448, MR 2000j:94033, Citations in This Document: §5

[20] Isaacs, I.M., Lichtman, A.I., Passman, D.S., Sehgal, S.K., Sloane, N.J.A., Zassenhaus, H.J. (eds.): Representation Theory, Group Rings, and Coding Theory: Papers in Honor of S. D. Berman. Contemporary Mathematics, vol. 93. American Mathematical Society, Providence (1989), See [23]

[21] Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.): Selected Areas in Cryptography, 16th Annual International Workshop, SAC 2009, Calgary, Alberta, Canada, August 13-14, 2009. LNCS, vol. 5867. Springer, Heidelberg (2009), See [30]

[22] Janwa, H., Moreno, O.: McEliece Public Key Cryptosystems Using Algebraic-Geometric Codes. Designs, Codes and Cryptography 3, 293–307 (1996), Citations in This Document: §1, §4

[23] Katsman, G.L., Tsfasman, M.A.: A Remark on Algebraic Geometric Codes. In: Representation Theory, Group Rings, and Coding Theory [20], pp. 197–199, Citations in This Document: §4

[24] Kim, K. (ed.): Public Key Cryptography: Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptosystems (PKC 2001) Held on Cheju Island, February 13-15, 2001. LNCS, vol. 1992. Springer, Heidelberg (2001), See [25]

[25] Kobara, K., Imai, H.: Semantically Secure McEliece Public-Key Cryptosystems — Conversions for McEliece PKC. In: PKC 2001 [24], pp. 19–35 (2001), MR 2003c:94027, Citations in This Document: §5, §6, §7

[26] Li, Y.X., Deng, R.H., Wang, X.M.: On the Equivalence of McEliece's and Niederreiter's Public-Key Cryptosystems. IEEE Transactions on Information Theory 40, 271–273 (1994), Citations in This Document: §2

[27] Matsui, M. (ed.): Proceedings of Advances in Cryptology — ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. LNCS, vol. 5912. Springer, Heidelberg (2009), See [16]

[28] McEliece, R.J.: A Public-Key Cryptosystem Based on Algebraic Coding Theory, JPL DSN Progress Report, pp. 114–116 (1978), `http://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF`, Citations in This Document: §1, §2

[29] Minder, L.: Cryptography Based on Error-Correcting Codes, Ph.D. Thesis, EPFL, PhD thesis 3846 (2007), Citations in This Document: §4

[30] Misoczki, R., Barreto, P.S.L.M.: Compact McEliece Keys from Goppa Codes. In: SAC 2009 [21], pp. 376–392 (2009), Citations in This Document: §1, §6

[31] Niederreiter, H.: Knapsack-Type Cryptosystems and Algebraic Coding Theory. Problems of Control and Information Theory 15, 159–166 (1986), Citations in This Document: §1, §2

[32] Overbeck, R., Sendrier, N.: Code-Based Cryptography. In: Post-Quantum Cryptography [6], pp. 95–145 (2009), Citations in This Document: §1, §7

[33] Patterson, N.J.: The Algebraic Decoding of Goppa Codes. IEEE Transactions on Information Theory 21, 203–207 (1975), Citations in This Document: §1, §5

[34] Peters, C.: Information-Set Decoding for Linear Codes over $\mathbf{F}_q$. In: PQCrypto 2010 [36], pp. 81–94 (2010), `http://eprint.iacr.org/2009/589`, Citations in This Document: §1, §4, §6, §6, §7

[35] Preneel, B. (ed.): Progress in Cryptology —- AFRICACRYPT 2009, Second International Conference on Cryptology in Africa, Gammarth, Tunisia, June 21-25, 2009. LNCS, vol. 5580. Springer, Heidelberg (2009), See [2]

[36] Sendrier, N. (ed.): Post-Quantum Cryptography, Third International Workshop, PQCrypto, Darmstadt, Germany, May 25-28, 2010. LNCS, vol. 6061. Springer, Heidelberg (2010), See [3], [34]

[37] Sendrier, N.: Finding the Permutation between Equivalent Linear Codes: The Support Splitting Algorithm. IEEE Transactions on Information Theory 46, 1193–1203 (2000), `http://hal.inria.fr/docs/00/07/30/37/PDF/ RR-3637.pdf` , MR 2001e:94017, Citations in This Document: §6

[38] Sidelnikov, V.M., Shestakov, S.O.: On an Encoding System Constructed on the Basis of Generalized Reed-Solomon Codes. Discrete Mathematics and Applications 2, 439–444 (1992), MR 94f:94009, Citations in This Document: §1, §2

[39] Stein, W. (ed.): Sage Mathematics Software (Version 4.4.3). The Sage Group (2010), `http://www.sagemath.org`, Citations in This Document: §5

[40] Stern, J.: A Method for Finding Codewords of Small Weight. In: [12], pp. 106–113 (1989), Citations in This Document: §6

[41] Sugiyama, Y., Kasahara, M., Hirasawa, S., Namekawa, T.: Further Results on Goppa Codes and Their Applications to Constructing Effcient Binary Codes. IEEE Transactions on Information Theory 22, 518–526 (1976), Citations in This Document: §1, §4, §4, §4

[42] Wagner, D.: A Generalized Birthday Problem (Extended Abstract). In: [45], pp. 288–303 (2002); See Also Newer Version [43], `http://www.cs.berkeley.edu/ daw/papers/genbday.html`

[43] Wagner, D.: A Generalized Birthday Problem (Extended Abstract) (Long Version) (2002); See Also Older Version [42], `http://www.cs.berkeley.edu/∼daw/ papers/genbday.html`, Citations in This Document: §6

[44] Wirtz, M.: On the Parameters of Goppa Codes. IEEE Transactions on Information Theory 34, 1341–1343 (1988), Citations in This Document: §4

[45] Yung, M. (ed.): Proceedings of Advances in Cryptology — CRYPTO 2002: 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 2002. LNCS, vol. 2442. Springer, Heidelberg (2002) ISBN 3-540-44050-X, See [42]