Study of McEliece & Niederreiter Cryptosystems

Snehil Suresh Wakchaure

Abstract—**These Most advanced security systems rely on public-key schemes based either on the factorization or discrete logarithm problem. Since both problems are known to be closely related, a major breakthrough in cryptanalysis tackling one of those problems could render a large set of cryptosystems completely useless. Most of today's cryptosystems are based on the trapdoor of the factoring problem (FP) or the discrete logarithm problem (DLP). Cryptosystems based on error correcting codes are a possible alternative. The McEliece public-key scheme is based on the alternative security assumption that decoding unknown linear, binary codes is NP-complete. In this work, McEliece and Niederreiter Cyptosystems which are based on binary Goppa error correcting codes are studied.**

***Index Terms*—Post-quantum Cryptography, McEliece & Niedderreiter cryptosystems, Binary irreducible Goppa codes**

I. Introduction

MOST of the present cryptosystems are either built on the difficulty of factoring large number (FP) or on the discrete logarithm problem (DLP). With a significant breakthrough in cryptanalysis or a major improvement of the best known attacks on these problems (i.e., the Number Field Sieve or Index Calculus), a large number of recently employed cryptosystems may turn out to be insecure overnight. Attacks against these systems are limited by the available computational power in terms of computational speed. Algorithms such as the Shor's algorithm render most of the presently secure cryptosystems useless due to the advent of quantum computers in the future. The McEliece cryptosystem incorporates a linear error-correcting code (namely a Goppa code) which is hidden as a general linear code. For Goppa codes, fast decoding algorithms exist when the code is known, but decoding codewords without knowledge of the coding scheme is proven NP-complete [3]. Contrary to DLP and FP-based systems, this makes this scheme also suitable for post-quantum era since it will remain unbroken when appropriately chosen security parameters are used.

In 1978, McEliece proposed the original McEliece cryptosystem based on classical Goppa codes which remains unbroken for certain system parameters. Several unsuccessful proposals involving various codes like Reed-Muller codes, Reed-Solomon-codes, Algebraic Geometric codes, etc. were made which were broken. The original scheme however remains more secure although few CCA2 secure versions have comparable level of security to the original scheme. Efforts to build a signature scheme had failed until Courtois, Finiaz & Sendrier successfully proposed their signature scheme.

II. Number Theory Basics

A field is a set **F** with distinguished elements 0 and 1 and binary operations + and x such that **F** forms an abelian goup under + with identity element 0. i.e. a+b=b+a; (a+b)+c =a+(b+c), a+0=a; '-a' exists such that –a+a=0. Also, **F** forms an abelian group under x, with identity element 1, i.e. a x b = b x a; (a x b) x c =a x (b x c); a x 1 = a ; there exists a^{-1} such that a^{-1}x a =1.

For finite fields, we know that if q is an integer greater than 1, then a field of order q exists if and only if q is a prime power, and all fields of order q are isomorphic. If q is a prime power, then we refer to the unique field of order q as $\mathbf{F_q}$. For example, if q is actually a prime, then $\mathbf{F_q}$ simply consists of the integers mod q, with the operations of addition and multiplication mod(q). For a finite field operations such as multiplication need a polynomial for the particular field over which the multiplication could be taken a modulus of. This approach is useful to improve space complexity in hardware and software implementations.

A vector space **V** is a set that is closed under finite vector addition and scalar multiplication. The basic example is n-dimensional Euclidean space $\mathbf{R^n}$, where every element is represented by a list of n real numbers, scalars are real numbers, addition is component wise, and scalar multiplication is multiplication on each term separately.
For a general vector space, the scalars are members of a field **F**, in which case $\mathbf{R^n}$ is called a vector space over **F.**
Euclidean n-space $\mathbf{R^n}$ is called a real vector space, and $\mathbf{C^n}$ is called a complex vector space. Vector spaces may involve field F as scalars for various operations such as vector addition, etc. Vector spaces follow the following rules: Commutativity, Associativity or Vector addition, Additive identity, Existence of additive inverse, Distributivity over scalar & vector sums, Scalar multiplication identity, etc.

For each prime p and positive integer m ≥ 1, there exists a finite field $F_p{}^m$ with $_p{}^m$ elements, and there exists no finite field with q elements if q is not a prime power. Any two fields with $_p{}^m$ elements are isomorphic. Under addition, $F_p{}^m$ is isomorphic to the vector space $F_p{}^m$. Under multiplication, the nonzero elements of $F_p{}^m$ form a cyclic group generated by a primitive element which belongs to $F_p{}^{m.}$ represented by a polynomial.

III. Coding Theory Basics

Linear codes: A linear code of length n over a field *F* is a subspace of $F^{n.}$ Thus the words of the codespace F^n are vectors referred to as codewords. If C is a linear code that, as a vector space over field *F*, has dimension *k* then we say that C is an [n,k] code. If C has minimum distance d, then C is an [n,k,d] code over *F*. 'n' is the length of the code, 'k' is the number if information bits and 'd' is the minimum distance of the code. The minimum distance of the code is the min(d(x,y)) where x and y are codewords lying in the corresponding field. The weight of the codeword is denoted by the number representing the nuber of non-zero entities in the codeword.

The generator matrix of a linear code is a spanning matrix where the rows are linearly independent of each other. We may construct many codes using the generator matrices. Generator matrix is the basis of a linear code, generating all possible codewords. The systematic form for the generator matrix is G = [I_k | P], where I_k is the identity matrix and P is the dimension of k x r. The generator matrix can be used to construct the parity check matrix of the code.

Two codes C1 and C2 are said to be equivalent if one code can be created from the other due to the following transformations: Permute components & scale components. Equivalent codes have the same distance.

The parity check matrix H is the generator matrix of the dual code. As such, a codeword c is in C if and only if the matrix-vector product $\mathbf{Hc^t=0}$. The parity check matrix can be constructed from the generator matrix as follows: H=[$-P^T|I_{n-k}$]. The parity check matrix is important to generate the syndrome vector S.

Syndrome decoding: Suppose Alice and Bob agree to use a linear binary code given by a k-dimensional subspace $C \subset \mathbb{F}^n$. They also have to agree on an encoding map. This is the same as choosing a basis for C, or equivalently, a generator matrix for C. We will assume this choice has been made once and for all. We will also assume that the generating matrix is in standard form, perhaps by changing to an equivalent code if necessary.

Suppose that Bob receives a word y $\in \mathbb{F}^n$. What message should he decode it as?

Syndrome decoding provides the following method.

1. If the generator matrix takes the form G = [Ik|A], work out the parity check matrix H = [A t|In - k].
2. Enumerate the cosets of C in C in $\mathbb{F}^n$and choose coset leaders ε_i for $i = 1, \ldots, 2^{n-k}$
3. Compute the syndromes S(ε_i) for each coset leader and construct the syndrome lookup table.
4. Compute the syndrome S(y) and from the table determine which $\varepsilon_i$satisfies S(y) = S(ε_i).
5. Decode the message as the first k-bits of y + ε_i

Let H be a parity check matrix for C. The syndromes associated with a received word r is s=rH^T. We have: s=rH^t= $(v+e)H^t$=vH^T+eH^T=eH^T i.e. the syndrome depends only on the error pattern e and not the transmitted codeword v. Syndromes and cosets of C are deeply related as two vectors x,y belonging to F^n yield the same syndrome if and only if they are elements of the same coset of C. For any set of codeowords, the cosets table is created and the coset leader are identified. Ususally the coset leaders are the cosets with the minimum weights (due to high probability of occurrence in practical life). The corresponding Syndrome is calculated. Now, the Syndrome calculated from the received word if matched with the Syndrome in the table, and the corresponding coset leader represents the error vector using which the original codeword message can be extracted.

The general decoding problem for linear codes is defined as follows:

1. Let C be an (n,k) linear code over *F* and y belongs to
2. Find x (which belongs to C) where dist(y,x) is minimal.

Let e be a vector of weight less than equal to t=floor(d-1/2) and x belongs to C, then there is unique solution to the general decoding problem for y=x+e. The code C is said to be t error correcting code. Also, there is the problem of finding weights (subspace weights) of a linear codeword which says that if C is an (n,k) linear code over *F* and w belongs to *N*, then find x (which belongs to C) which satisfies dist (0,x)=w. The above two problems of general decoding and finding subspace weights is NP hard. Given 2 generator matrices, the problem is to decide if the codes generated by the matrices are permutation equivalent or not. In the case of where F=F_2, permutation equivalency is the same as equivalency.

IV. Binary Goppa Codes

A. Encoding and decoding using irreducible binary Goppa

The binary Goppa code is an error-correcting code that belongs to the class of general Goppa codes, and have notable advantages over non-binary goppa codes. The advantages of Goppa codes are as follows: The lower bounds for goppa codes are easy to compute 2. The knowledge of generating polynomial allows efficient error correction.3. Without the knowledge of the generating polynomial no efficient algorithm for error correction is known.

Binary Goppa code is defined by a polynomial g(x) of degree t over a finite field GF(2^m) without multiple zeroes, and a sequence L of n distinct elements from GF(2m) that aren't the roots of the polynomial. The polynomial if irreducible and monic provides better secirurity for McEliece.

$$\forall i,j \in \mathbb{Z}_n : L_i \in GF(2^m) \wedge L_i \neq L_j \wedge g(L_i) \neq 0$$

Codewords belong to the kernel of syndrome function, forming a subspace of $\{0,1\}^n$.

$$\Gamma(g,L) = \left\{ c \in \{0,1\}^n \middle| \sum_{i=0}^{n-1} \frac{c_i}{x - L_i} \equiv 0 \mod g(x) \right\}$$

Code defined by the tuple (g,L) has minimum distance 2t+1, thus it can correct

$$t = \left\lfloor \frac{(2t+1)-1}{2} \right\rfloor$$

errors in a word size n-mt using codewords of size n. It also possesses convenient parity check matrix of the form as shown below. The parity check matrix when multiplied by the transpose of the word to be tested, if zero verifies that the word is a codeword lying in the field GF(2^m). The rows of matrix H generate a vector space V which is a subspace of F_2^n. H is an mtxn matrix, G is a nxk matrix with k>=n-mt for the Goppa code. G can be computed from H by using GH^T=0., so the vectors in nullspace of H modulo 2 form the rowspace of G. In linear algebra, the kernel (also null space or nullspace) of a matrix A, is the set of all vectors x for which Ax = 0.

$$\mathcal{L} = \{\alpha_0, \cdots, \alpha_{n-1}\} \qquad \alpha_i \in \mathbb{F}_{2^m}$$

a tuple of n distinct elements, called support, *such that*

$$g(\alpha_j) \neq 0, \forall 0 \leq j \leq n.$$

For any vector $c = (c_0, \cdots, c_{n-1}) \in \mathbb{F}^n$, *define the* syndrome *of c by*

$$\mathcal{S}_c(z) = -\sum_{i=0}^{n-1} \frac{c_i}{g(\alpha_i)} \frac{g(z) - g(\alpha_i)}{z - \alpha_i} \mod g(z)$$

The binary Goppa code $\Gamma(\mathcal{L}, g(z))$ *over* $\mathbb{F}_2$ *is the set of all* $c = (c_0, \cdots, c_{n-1})$ *such that the indentity*

$$\mathcal{S}_c(z) = 0$$

holds in the polynomial ring $\mathbb{F}_{2^m}(z)$ *or equivalent*

$$\mathcal{S}_c(z) = \sum_{i=0}^{n-1} \frac{c_i}{z - \alpha_i} \equiv 0 \mod g(z)$$

Thus, a parity check matrix of $\mathcal{G}(\mathbf{L}, g(X))$ can be written as

$$\mathsf{H} = \begin{pmatrix} g_t g(\gamma_0)^{-1} & \cdots & g_t g(\gamma_{n-1})^{-1} \\ (g_{t-1} + g_t\gamma_0)g(\gamma_0)^{-1} & \cdots & (g_{t-1} + g_t\gamma_{n-1})g(\gamma_{n-1})^{-1} \\ \vdots & \ddots & \vdots \\ \left(\sum_{j=1}^{t} g_j\gamma_0^{j-1}\right) g(\gamma_0)^{-1} & \cdots & \left(\sum_{j=1}^{t} g_j\gamma_{n-1}^{j-1}\right) g(\gamma_{n-1})^{-1} \end{pmatrix} = \mathsf{XYZ}$$

where

$$\mathsf{X} = \begin{pmatrix} g_t & 0 & 0 & \cdots & 0 \\ g_{t-1} & g_t & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_1 & g_2 & g_3 & \cdots & g_t \end{pmatrix}, \quad \mathsf{Y} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \gamma_0 & \gamma_1 & \cdots & \gamma_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_0^{t-1} & \gamma_1^{t-1} & \cdots & \gamma_{n-1}^{t-1} \end{pmatrix}, \quad \text{and}$$

$$\mathsf{Z} = \begin{pmatrix} \frac{1}{g(\gamma_0)} & & & \\ & \frac{1}{g(\gamma_1)} & & \\ & & \ddots & \\ & & & \frac{1}{g(\gamma_{n-1})} \end{pmatrix}$$

and therefore we have

$$\mathbf{c} \in \mathcal{G}(\mathbf{L}, g(X)), \quad \text{iff} \quad \mathsf{H}\mathbf{c}^T = 0.$$

The minimum distance of a Goppa code G which is generated by an irreducible polynomial of degree t is atleast 2t+1. Therefore, it is always possible to correct upto t errors.

Encoding a message: for a Goppa(L, g(z)) Goppa code defined by a polynomial of degree t over GF(q^m), and by choosing L (such that g(Li)!=0) of size n, dimension k, minimum distance d a message is encoded by writing it in blocks of k symbols and multiplying every block by the generator matrix G. ie. (m_1, …. m_k). G =(c_1 ….. c_n). The decoding algorithm for binary irreducible Goppa codes is given above.

B. Consequences of Choice of L

One can choose L within certain restrictions for Goppa codes. The number of elements in L i.e. n has both upper and lower bounds . First of all since L can only contain elements of GF(q^m) which are no roots of *g(z),* we know that: n<=q^m-R where R is the number of distinct roots of g(z). Also, we have to make sure that n>=1+mt for the code to be usable. For practical purposes, a long value of L is desired to avoid message expansion (n/k). Since original message is divided into blocks of length k and then encoded into blocks of length n, the message is expanded by a factor of n/k, which we wish to minimize. In order to do so, L (of length n) should be chosen large.

$$\forall i,j \in \mathbb{Z}_n : L_i \in GF(2^m) \wedge L_i \neq L_j \wedge g(L_i) \neq 0$$

Algorithm 2.3.1 Error Correction of Binary Irreducible Goppa Codes

```
Input: A binary irreducible Goppa code $\mathcal{G}(\mathbf{L}, g(X))$, a vector $\mathbf{c} = \mathbf{m} \oplus \mathbf{e}$,
where $\mathbf{m}$ is a codeword and $\mathbf{e}$ is an error vector.
Output: The message $\mathbf{m}$ and the error vector $\mathbf{e}$.

/* Compute the syndrome of c */
$S_\mathbf{c}(X) = \sum_{i=0}^{n-1} \frac{c_i}{X-\gamma_i} \mod g(X)$ (or use the parity check matrix H)

if $S_\mathbf{c}(X) \equiv 0 \mod g(X)$ then
  /* there is no error, c is a codeword */
  return(c, 0)
else
  /* there are errors, c is not a codeword */
  $T(X) \equiv S_\mathbf{c}^{-1}(X) \mod g(X)$
  $\tau(X) \equiv \sqrt{T(X) + X} \mod g(X)$

  /* extended Euclidean algorithm */
  $i = 0$; $r_{-1}(X) = \alpha_{-1}(X) = g(X)$; $r_0(X) = \alpha_0(X) = \tau(X)$; $\beta_{-1}(X) = 0$;
  $\beta_0(X) = 1$
  while $\deg(r_i(X)) \geq \lfloor (t+1)/2 \rfloor$ do
    $i = i + 1$
    Determine $q_i(X)$ and $r_i(X)$, s.t. $r_i(X) = r_{i-2}(X) - q_i(X)r_{i-1}(X)$
      and $\deg(r_i(X)) < \deg(r_{i-1}(X))$
    $\beta_i(X) = \beta_{i-2}(X) + q_i(X)\beta_{i-1}(X)$
    $\alpha_i(X) = r_i(X)$

  $\sigma(X) = c^2((\alpha_i(X))^2 + X(\beta_i(X))^2)$ with $c \in \mathbb{F}_{2^m}$, s.t. $\sigma(X)$ is monic

  /* Determination of zeroes of $\sigma_\mathbf{e}(X)$ */
  for $i = 0$ to $n - 1$ do
    if $\sigma(\gamma_i) = 0$ then
      $\mathbf{e}_i = 1$
    else
      $\mathbf{e}_i = 0$
  $\mathbf{m} = \mathbf{c} \oplus \mathbf{e}$
  return(m, e)
```

The above figure shows the decoding scheme for the binary irreducible Goppa codes.

C. Run time of decoding algorithm

TABLE I
DECODING ALGORITHM TIME COMPLEXITY

Operation	Time
Compute Syndrome $S_c(X)$	(n-k)n binary operations
Find T(X) by Extended Euclid's algorithm	$O(t^2m^2)$
Sqrt(T(X)+X)	$O(t^2m^2)$
Find roots of error locator polynomial	n(tm^2+tm) operations
Total decoding time	$O(n.t.m^2)$

V. The McEliese cryptosystem

The McEliece cryptosystem uses the decoding hardness of binary irreducible Goppa codes as follows:

- **System Parameters:** $n, t \in \mathbb{N}$, where $t \ll n$.
- **Key Generation:** Given the parameters n, t generate the following matrices:

G' :	$k \times n$ generator matrix of a binary irreducible (n, k) Goppa code $\mathcal{G}$ which can correct up to t errors, where k is chosen maximal.
S :	$k \times k$ random binary non-singular matrix
P :	$n \times n$ random permutation matrix

  Then, compute the $k \times n$ matrix $\mathsf{G} = \mathsf{S}\mathsf{G}'\mathsf{P}$.

- **Public Key:** (G, t)
- **Private Key:** $(\mathsf{S}, D_\mathcal{G}, \mathsf{P})$, where $D_\mathcal{G}$ is an efficient decoding algorithm for $\mathcal{G}$ (see e.g. algorithm 2.3.1).

- **Encryption:** To encrypt a plaintext $\mathbf{m} \in \{0,1\}^k$ choose a vector $\mathbf{z} \in \{0,1\}^n$ of weight t randomly and compute the ciphertext $\mathbf{c}$ as follows:

$$\mathbf{c} = \mathbf{m}\mathsf{G} \oplus \mathbf{z} \ .$$

- **Decryption:** To decrypt a ciphertext $\mathbf{c}$ calculate

$$\mathbf{c}\mathsf{P}^{-1} = (\mathbf{m}\mathsf{S})\,\mathsf{G}' \oplus \mathbf{z}\mathsf{P}^{-1}$$

first, and apply the decoding algorithm $\mathcal{D}_{\mathsf{G}}$ for $\mathcal{G}$ to it. Since $\mathbf{c}\mathsf{P}^{-1}$ has a hamming distance of t to the Goppa code we obtain the codeword

$$\mathbf{m}\mathsf{S}\mathsf{G}' = \mathcal{D}_{\mathcal{G}}\left(\mathbf{c}\mathsf{P}^{-1}\right) \ .$$

Let $J \subseteq \{1,\cdots,n\}$ be a set, such that $\mathsf{G}_{\cdot J}$ is invertible, then we can compute the plaintext $\mathbf{m} = (\mathbf{m}\mathsf{S}\mathsf{G}')_J\,(\mathsf{G}'_{\cdot J})^{-1}\,S^{-1}$

A. *Trapdoor to McEliece cryptosystem*

The trapdoor for the McEliece cryptosystem is the knowledge of the generator polynomial g(x). The general decoding problem is as follows:

Problem 1.2.3 *The general decoding problem for linear codes is defined as follows:*

- *Let $\mathcal{C}$ be an (n,k) linear code over $\mathbb{F}$ and $\mathbf{y} \in \mathbb{F}^n$.*
- *Find $\mathbf{x} \in \mathcal{C}$ where $\operatorname{dist}(\mathbf{y},\mathbf{x})$ is minimal.*

Let $\mathbf{e}$ be a vector of weight $\le t := \lfloor \frac{d-1}{2} \rfloor$ and $\mathbf{x} \in \mathcal{C}$. Then there is a unique solution to the general decoding problem for $\mathbf{y} = \mathbf{x} + \mathbf{e}$. The code $\mathcal{C}$ is said to be an t-error correcting code.

Thus, we see that given x, it is highly infeasible to find the other codewords i.e. y. This is the general decoding problem due to which the Goppa code can be used as a trapdoor to the McEliece Cryptosystem. If anyone solves the general decoding problem, McEliece shall break but the reverse is not true. Also, the scrambler function provides security to the McEliece system.

B. *Security of McEliece cryptosystem*

The following would break the system:

- Knowledge of the generator polynomial
- Solving of the general decoding problem
- Attacks on private key: Not every error code is a good choice for McEliece.
- If P is secret, L may be revealed without any problems
- If g(x) is unknown, and the attacker successfully computes P and M, then g(x) can be found out and the system breaks. (both McEliece & Niederreiter)
- If P is revealed, it is easy to recover g(x)
- If the attacker finds G, he can find g(x). He can then use the decryption algorithm to break the code and find the transmitted message. Therefore, a possible attack is to guess S & P.

C. *Types of Attacks*

There are 2 types of attacks: try to decode given encrypted message (direct attack) & tying to decode the structure of the original code from the public key (structural attack).

Guessing S & P:

- If P is revealed, it is easy to recover g(x)
- If the attacker finds G, he can find g(x). He can then use the decryption algorithm to break the code and find the transmitted message. Therefore, a possible
- The number of possible S matrices for the attacker is:
$$\prod_{i=0}^{k-1}(2^k - 2^i).$$
-
- The number of possible P matrices are n!
- As dimensions of S & P increase, it becomes more & more difficult to guess S & P.

Exhaustive codewords comparison:

This is another brute force approach is where all 2^k codewords in the code defined by G'=SGP are generated to find the unique codeword c closest to y, thus c=mG' and d(c,y)<=t. m is then extracted using Gaussian elimination method.

Syndrome decoding:

The cryptanalyst could compute the parity check matrix H' corresponding to G', using $G'(H')^T=0$. Next, he could compute the syndrome of the transmitted word y , which is:

$$y(H^*)^T = (mG^* + e)(H^*)^T = mG^*(H^*)^T + e(H^*)^T = e(H^*)^T.$$

If he finds the error vector e, he discovers mG' and can easily find m by applying Gaussian elimination. Therefore he can generate all possible error vectors of length n and weight <=t, compute $e(H^*)^T$ and compare this to the syndrome.

Information Set Decoding:

This is another kind of attack in which some attacking the system can select k random positions hoping that they are not in error. If the restriction of G' to these k positions still has rank k, one can find a candidate m' for the transmitted message m by applying Gaussian e
limitation. This is the most effective attack so far, but is still considered infeasible. The expected workload is of the order of 10^{19}.

Side channel attacks:

The susceptibility of McEliece system to side channel attacks has not been extensively studied yet. This is due to low number of systems employing McEliece type systems. A successful timing attack on Patterson's algorithm was demonstrated which recovers the error vector instead of the key

D. *Multiple Encryptions of the same message (McEliece)*

One of the disadvantages of the systems is that it is not safe for the use of sending the same message several times with the same encryption matrix G'.

E. *Advantages & Disadvantages of McEliece cryptosystem*

Advantages:

- The system is elegant and easy to understand
- Its security has been well studied since 1978 with various other codes too
- Implementation achieve relatively high encryption/decryption speeds

Disadvantages:

- The system is easily broken when multiple encryptions of the same message are used
- The Cipher text messages are n/k times longer than the plain text messages. Bad for space complexity & security (Cipher text only attacks)
- Size of the keys used in the system are very large as compared to the cryptosystems like RSA
- There is no obvious way to use the scheme for signatures as there is with RSA

G. *Available Implementations*

- Hardware Implementation of McEliece scheme on reconfigurable hardware targeting Spartan 3 devices.
- I386 assembler implementation
- A C implementation for 32-bit architectures
- Java API provided by Flexiprovider for both McEliece & Niederreiter

F. *Parameter selection*

The original parameters suggested by McEliece are m=10, t=50, while some of the authors suggest t=38 as a better choice for computational complexity of the algorithm while not reducing the security level. These parameters give 2^{60} bit security.

Table 2.1.: Security of McEliece Depending on Parameters

Security Level	**Parameters** (n,k,t), errors added	**Size** K_{pub} in KBits	**Size** K_{sec} $(G(z),P,S)$ in KBits
Short-term (60 bit)	(1024, 644, 38), 38	644	(0.38, 10, 405)
Mid-term (80 bit)	(2048, 1751, 27), 27	3,502	(0.30, 22, 2994)
Long-term (256 bit)	(6624, 5129, 115), 117	33,178	(1.47, 104, 25690)

G. *Key sizes*

Parameter Sizes and Costs

parameters	n	2^{15}	2^{16}		2^{17}		
	t	10	9	10	8	9	10
size public key in MB	$k(n-k)/(8\cdot 1024^2)$	0.58	1.12	1.12	2.38	2.38	2.38
signature cost	$t!t^2m^3$	2^{40}	2^{37}	2^{40}	2^{34}	2^{38}	2^{41}
verification cost	t column operations[2]	2^{18}	2^{19}	2^{19}	2^{20}	2^{20}	2^{20}
signature length	$\log_2(n^t)$	150	144	160	136	153	170

McEliece & Niederreiter private Key size:

$$(n-k)n + (n-k+1+2\cdot\log_2 n) + k^2 + n\cdot\log_2 n$$

McEliece & Niederreiter public Key size:
K(n-k) bits

McEliece system parameters	Size public key in bytes		Workfactor (binary operations)	
$(n,k,d=2t+1)$	plain	CCA2-secure	encryption	decryption
$(1024, 524, 101)$	67,072	32,750	2^{18}	2^{22}
$(2048, 1608, 81)$	411,648	88,440	$2^{20.5}$	2^{23}
$(2048, 1278, 141)$	327,168	123,008	2^{20}	2^{24}
$(2048, 1025, 187)$	262,400	131,072	2^{20}	$2^{24.5}$
$(4096, 2056, 341)$	1,052,672	524,280	2^{22}	$2^{26.5}$

Table 2: Performance of the McEliece PKC

McEliece system parameters	Workfactor (binary operations)		
	GISD	LEON-LWCW	CC-LWCW [3]
$(n,k,d=2t+1)$	$p=2$	$p=3,\ l=m$	$p=2,\ l=2m-1$
$(1024, 524, 101)$	2^{70}	2^{69}	2^{64}
$(2048, 1608, 81)$	2^{110}	2^{107}	2^{98}
$(2048, 1278, 141)$	2^{120}	2^{118}	2^{110}
$(2048, 1025, 187)$	2^{115}	2^{112}	2^{106}
$(4096, 2056, 341)$	2^{195}	2^{193}	2^{184}

Table 3: Attacking the McEliece PKC

VI. NIEDERREITER CRYPTOSYSTEM

A. *Encryption & Decryption*

As size of the keys is less of a concern today due to huge memory capacity, main handicap for McEliece has been that it could not be used for signature. Only recently, Courtoius, Finiasz & Sendrier showed it is possible to construct a signature scheme based on the Niederreiter cryptosystem. The Niederreiter cryptosystem is a variation of the McEliece Cryptosystem developed in 1986 by Harald Niederreiter. It applies the same idea to the parity check matrix H of a linear code. Niederreiter is equivalent to McEliece from a security point of view. It uses a syndrome as ciphertext and the message is an error pattern. The encryption of Niederreiter is about ten times faster than the encryption of McEliece. Niederreiter can be used to construct a digital signature scheme.
A special case of Niederreiter's original proposal was broken[2] but the system is secure when used with a Binary Goppa code.

Key generation

1. Alice selects a binary (n, k)-linear Goppa code G capable of correcting t errors. This code possesses an efficient decoding algorithm.
2. Alice generates a $(n - k) \times n$ parity check matrix H for the code G.
3. Alice selects a random $(n - k) \times (n - k)$ binary non-singular matrix S.
4. Alice selects a random $n \times n$ permutation matrix P.
5. Alice computes the $(n - k) \times n$ matrix $H_{pub} = SHP$.
6. Alice's public key is (H_{pub}, t); her private key is (S, H, P).

Message encryption

Suppose Bob wishes to send a message m to Alice whose public key is (H_{pub}, t):

1. Bob encodes the message m as a binary string of length n and weight at most t.
2. Bob computes the ciphertext as $c = H_{pub}m^T$.

Message decryption

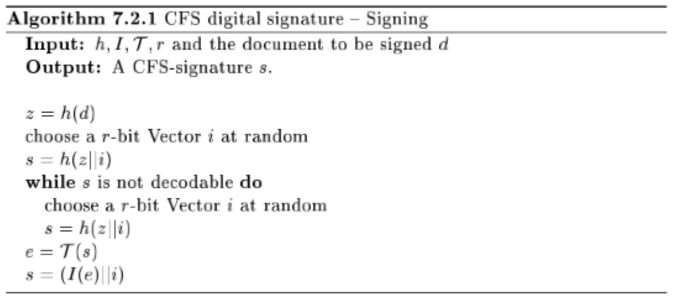
Upon receipt of $c = H_{pub}m^T$ from Bob, Alice does the following to retrieve the message m.

1. Alice computes $S^{-1}c = HPm^T$.
2. Alice applies a syndrome decoding algorithm for G to recover PmT.
3. Alice computes the message m via $mT = P^{-1}Pm^T$.

Recommended values for these parameters are n = 1024, t = 38, k = 644.

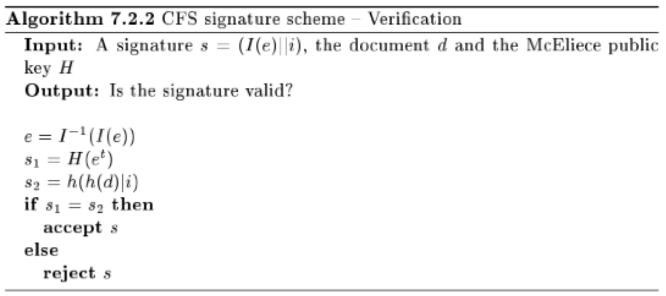
B. *CFS Signature Scheme*

__Courtois, Finiasz and Sendrier showed how the Niederreiter cryptosystem can be used to derive a signature scheme.
1. Hash the document d to be signed (with a public hash algorithm).
2. Decrypt this hash value as if it were an instance of ciphertext.
3. Append the decrypted message to the document as a signature.



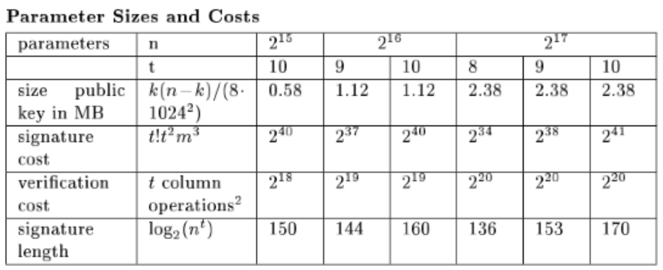
Algorithm 7.2.1 CFS digital signature – Signing

```
Input: $h, I, \mathcal{T}, r$ and the document to be signed $d$
Output: A CFS-signature $s$.

$z = h(d)$
choose a $r$-bit Vector $i$ at random
$s = h(z||i)$
while $s$ is not decodable do
    choose a $r$-bit Vector $i$ at random
    $s = h(z||i)$
$e = \mathcal{T}(s)$
$s = (I(e)||i)$
```



Algorithm 7.2.2 CFS signature scheme – Verification

```
Input: A signature $s = (I(e)||i)$, the document $d$ and the McEliece public key $H$
Output: Is the signature valid?

$e = I^{-1}(I(e))$
$s_1 = H(e^t)$
$s_2 = h(h(d)|i)$
if $s_1 = s_2$ then
    accept $s$
else
    reject $s$
```



Parameter Sizes and Costs

parameters	n	2^{15}	2^{16}		2^{17}		
	t	10	9	10	8	9	10
size public key in MB	$k(n-k)/(8\cdot 1024^2)$	0.58	1.12	1.12	2.38	2.38	2.38
signature cost	$t!t^2m^3$	2^{40}	2^{37}	2^{40}	2^{34}	2^{38}	2^{41}
verification cost	t column operations^2	2^{18}	2^{19}	2^{19}	2^{20}	2^{20}	2^{20}
signature length	$\log_2(n^t)$	150	144	160	136	153	170

VII. McEliece & Niederreiter cryptosystems

- The Niederreiter scheme is derived from the McEliece scheme
- Niederreiter scheme uses the parity check matrix H, while the McEliece scheme uses the Generator matrix G
- Niederreiter scheme was developed to implement a signature scheme such as CFS since McEliece could not be used to implement a signature scheme
- The Encryption in Niederreiter is faster than that of McEliece scheme
- Security of McEliece and Niederreiter schemes are equivalent. Breaking one will also break the other.

VIII. Conclusion

The trapdoor for McEliece & Niederreiter functions is the general decoding problem for error correcting codes. It is hard to find the minimum distance of a C(n,k) linear code over the field F. These systems use this fact along with fast decoding algorithms to achieve code based cryptographic applications. Although highly secure, McEliece systems, although as old as RSA, have not replaced the RSA scheme due to large key sizes which may be an issue for embedded devices. However, with the advent of quantum computing the need for such systems which are more secure than the traditional factoring problem (FP) or Discrete logarithm problem (DLP) based systems.

IX. References

[1] D. Engelbert, “A summary of McEliece type cryptosystems & their security”, TU-Darmsdadt, 2006

[2] Ellen Jochemsz, “Goppa codes & the McEliece cryptosystem”, TU-Darmsdadt, Amsterdam, 2002

[3] Estefan Heyse, “Code based Cryptography, Implementing the McEliece scheme on reconfigurable hardware”, Ruhr-University Bochum, 2009

[4] Siddhartha Biswas,”Introduction to Coding theory, basic codes and shanon’s theorem”

[5] Marco Baldi,”Enhanced public key security for McEliece system”

[6] Bradley J Lucier,”Cryptography, finite fileds & AltiVac”

[7] F. J. Macwilliams, “The theory of error correcting codes”

[8]

[9] W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.

[10] H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.