A graduate introduction to classic McEliece

Ganyu (Bruce) Xu

November, 2024

1 Introduction

Classic McEliece is an IND-CCA secure key encapsulation mechanism (KEM) submitted to NIST's post-quantum cryptography (PQC) standardization project. Its security is based on the conjectured intractability of the **Syndrome Decoding Problem**.

This document provides an introduction to the mathematics behind classic McEliece with a particular emphasis on the details of binary Goppa code. We assume a graduate level of mathematical maturity, though most of the results can be reasoned about using first or second year undergraduate math.

Classic McEliece is an appealing candidate for PQC because its cryptanalysis has been remarkably stable since its earliest conception in 1978 [McE78] (whereas lattice-based cryptography saw enormous advance in its cryptanalysis in the last decade alone, forcing many latticed-based schemes to revise their parameters). Classic McEliece also has the following advantages:

- Encryption is faster
- Ciphertext is smaller
- There is no decryption failure that can leak information about secret key

On the other hand, Classic McEliece has larger public key size and slower decryption routine.

1.1 A summary of binary Goppa code

Binary Goppa code is a linear error-correcting code. Each instance is parameteried by:

- some base field $K = \mathbb{F}_{2^m}$
- n distinct field elements $\alpha_1, \alpha_2, \ldots, \alpha_n \in K$
- An irreducible polynomial $g \in K[x]$ such that $\deg(g) = t$

1.2 An overview of Classic McEliece

Each instance of Classic McEliece is parameterized by three integers:

- m is the size of the base field $K = \mathbb{F}_{2^m}$
- n is the size of code words: codewords $\subseteq K^n$
- t is the error-correcting capacity of the underlying Goppa code

Key generation. Begin by randomly generating the parameters of a Goppa code instance, which include:

- n distinct field elements $\alpha_1, \alpha_2, \ldots, \alpha_n \in K$
- A degree-t, square-free polynomial $g \in K[x]$ such that $g(\alpha_i) \neq 0$ for $1 \leq i \leq n$

$\alpha_1, \alpha_2, \ldots, \alpha_n$ and g can be used to compute the canonical parity-check matrix $H \in K^{t \times n}$

$$H_{i,j} = \frac{\alpha_j^{i-1}}{g(\alpha_j)} \text{ for } 1 \leq i \leq t, 1 \leq j \leq n$$

H is then transformed into reduced row-echelon form (or systematic form per DJB) $H' = [I_t \mid T]$ for some $T \in K^{t \times (n-t)}$. T is returned as the public key. $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_n, g)$ is returned as the secret key.

Encryption. The message space is the subset of $\mathbb{F}_2^n$ whose Hamming weight is exactly t:

$$\mathcal{M} = \{\mathbf{e} \in \mathbb{F}_2^n \mid wt(\mathbf{e}) = t\}$$

To encrypt, compute:

$$\mathbf{y} \leftarrow [I_t \mid T]\mathbf{e}$$

$\mathbf{y} \in K^t$ is returned as the ciphertext.

Decryption. Given $\mathbf{y} \in K^t$ and $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_n, g)$, let $\mathbf{r} \leftarrow (y_1, y_2, \ldots, y_t, 0, 0, \ldots, 0) \in \mathbb{F}_2^n$. then feed $(\Gamma, \mathbf{r})$ into some Goppa decoder, which will directly recover $\mathbf{e} \in \mathbb{F}_2^n$ such that $wt(\mathbf{e}) = t$. Return $\mathbf{e}$ as the decryption.

2 Preliminaries

3 Understanding binary Goppa code decoding

3.1 Polynomial interpolation

Theorem 3.1 (Lagrange Interpolation)**.** *Let K be a finite field, $\alpha_1, \alpha_2, \ldots, \alpha_n \in K$ be n distinct field elements, and $r_1, r_2, \ldots, r_n \in K$ be n (possibly non-distinct) elements. There exists a unique degree-$(n-1)$ polynomial f such that $f(\alpha_i) = r_i$:*

$$f = \sum_{i=1}^{n} \left(r_i \prod_{j \neq i} \frac{x - \alpha_j}{\alpha_i - \alpha_j} \right)$$

It's easy to check that the interpolation is correct, and that the degree of f is less than n. If there exists another $g \in K[x]$ with $\deg(g) < n$ that interpolates $(\alpha_i, r_i)_{i=1}^n$, then $f - g$ is a polynomial with n distinct roots. However, $\deg(f - g)$ is less than n since $\deg(f), \deg(g) < n$. Therefore, it must be that $f - g = 0$, which means that $f = g$, thus proving uniqueness.

Shamir's secret sharing. In the secret sharing scheme proposed by Shamir in 1979 [Sha79], the secret is a degree-$t-1$ polynomial, and each share is $(\alpha, g(\alpha))$ for some $\alpha \in K$. If there are t distinct shares, then the secret polynomial can be recovered; if there are only $t-1$ shares, then there are as many unique polynomials as there are distinct possible points in the field, meaning that if the field K has cryptographically large order, then any adversary with only $t-1$ shares will have negligible probability of recovering the secret. This scheme is information-theoretically secure.

3.2 Low-degree approximant

Is low-degree approximant related to approximant in linear algebra? Probably yes, but we need a clear connection

Theorem 3.2 (Best approximant theorem)**.** *Let K be some finite field and n, t be non-negative integers such that $2t < n$. Given polynomials $A, B \in K[x]$ such that $\deg(B) < \deg(A)$, then there exists unique pair of polynomials $a, b \in K[x]$ such that $\deg(a) \leq t$, $\deg(b) < \deg(a)$, $\gcd(a, b) = 1$, and $\deg(aB - bA) < n - t$. If $c, d \in K[x]$ is such that $\deg(c) \leq t$ and $\deg(cB - dA) < n - t$, then $(c, d) = \lambda(a, b)$ for some $\lambda \in K[x]$.*

Proof. Let's first bound the degree of the polynomial $aB - bA$: $\deg(aB) = \deg(a) + \deg(B) < n + t$, and similarly we have $\deg(bA) = \deg(b) + \deg(A) < n + t$. Therefore, $\deg(aB - bA) = \max(\deg(bA), \deg(aB)) < n + t$, which means that the maximal degree of $aB - bA$ is $n - t - 1$. We can write $aB - bA$ as follows:

$$aB - bA = \sum_{i=0}^{n+t-1} k_i \cdot x^i$$

Now consider a function σ that maps (a, b) to the higher-order $2t$ coefficients of $aB - bA$. Since the map $(a, b) \mapsto aB - bA$ is trivially linear, and the map from $aB - bA$ to its coefficients is trivially linear, it is easy to see that $\sigma : (a, b) \mapsto (k_{n+t-1}, k_{n+t-2}, \ldots, k_{n-t})$ is also a linear map.

Because $\deg(a) \leq t$ and $\deg(b) < t$, a can have up to $t + 1$ non-zero coefficients, and b can have up to t non-zero coefficients, so (a, b) is an $2t + 1$-dimensional linear space, while $\text{Img}(\sigma)$ has $2t$ dimensions. By the rank-nullity theorem, we know that $\sigma(a, b) = \mathbf{0}$ has non-zero solution. With this non-zero solution, the first $2t$ coefficients of $aB - bA$ are all zeros, so the degree of $aB - bA$ must be less than $n - t$.

At this point we have arrived at some (a, b) such that $\deg(b) < \deg(a) \leq t$ and $\deg(aB - bA) < n - t$. If $\gcd(a, b) \neq 1$, then we can divide a, b by the non-trivial GCD, which will make them co-prime while $\deg(aB - bA)$ must still be less than $n - t$. This proves the existence of co-prime degree-t approximants, now we will prove the uniqueness of such a solution.

Suppose that $c, d \in K[x]$ are polynomials such that $\deg(d) < \deg(c) \leq t$ and $\deg(cB - dA) < n - t$ (but they are not necessarily co-prime), then consider the following equation:

$$\det\left(\begin{bmatrix} a & c \\ aB - bA & cB - dA \end{bmatrix}\right) = a(cB - dA) - c(aB - bA) = (cb - ad)A$$

The degree of the middle term $\deg(a(cB-dA)-c(aB-bA))$ is less than n because $\deg(cB-dA), \deg(aB-bA)$ are both less than $n - t$ and $\deg(a), \deg(c)$ are both at most t. On the other hand, since we assumed $\deg(A) = n$, $a(cB - dA) - c(aB - bA) = (cb - ad)A$ must imply that $cb - ad = 0$. In addition, since we assumed $\gcd(a, b) = 1$, it must be that $a \mid c$, from here we naturally get $d = c/a \cdot b$. In other words, $(c, d) = c/a \cdot (a, b)$ is a non-zero multiple of (a, b). Any other degree-t approximants cannot have lower degree than (a, b), which makes (a, b) the *best approximants*. □

Note that the proof above also gave an algorithm for finding (a, b): given A, B, the coefficients of $aB - bA$ can be explicitly written as a linear system of equation with respect to the coefficients ot (a, b), so finding non-zero (a, b) is equivalent to solving a linear system $M \in K^{(2t+1)\times 2t}$. For the remainder of the paper we assume the existence of a well-defined function `approximate` : $(A, B, t) \mapsto (a, b)$

3.3 Interpolation with error

Let n, t be non-negative integers such that $2t < n$. Using low-degree polynomial approximants, we can recover a degree-$(n - 2t - 1)$ polynomial using evaluations at n distinct points with up to t errors. Specifically:

1. $f \in K[x]$ is a polynomial with degree $\deg(f) < n - 2t$
2. $\alpha_1, \alpha_2, \ldots, \alpha_n \in K$ are distinct field elements
3. $r_1, r_2, \ldots, r_n \in K$ are such that $r_i \neq f(\alpha_i)$ at no more than t positions

The algorithm is as follows

1. Let $A = \prod_{i=1}^{n}(x - \alpha_i)$
2. Interpolate $B \in K[x]$ such that $B(\alpha_i) = r_i$ for $1 \leq i \leq n$
3. Compute degree-t approximant (a, b) of (A, B) using theorem 3.2. The error can be recovered if an only if $a \mid A$
4. Compute $\hat{f} = B - bA/a$, which will be the output

Proof. let $S = \{i \mid r_i = f(\alpha_i)\}$ denote the set of positions where r_i is correct. A can be factored into two polynomials:

$$A = \left(\prod_{i \in S}(x - \alpha_i)\right)\left(\prod_{i \notin S}(x - \alpha_i)\right)$$

We denote the first factor by $E = \prod_{i \in S}(x - \alpha_i)$ and the second factor by $c = \prod_{i \notin S}(x - \alpha_i)$, then $A = Ec$.

Now observe the polynomial $(B - f)$: since for all $i \in S$ we have $B(\alpha_i) = r_i = f(\alpha_i)$, we know that $x - \alpha_i$ divides $B - f$, and since α_i's are all distinct, $\prod_{i \in S}(x - \alpha_i) = E$ must divide $B - f$. In other words:

$$Ed = B - f$$

for some non-zero polynomial $d \in K[x]$. Observe that:

$$Ad = (Ec)d = (Ed)c = (B - f)c = Bc - fc$$

which re-arranges to:

$$fc = cB - dA$$

Let's take a look at the degrees of each term:

- $\deg(f) < n - 2t$ by assumption
- $\deg(c) \leq t$ since we assumed there to be no more than t errors
- $\deg(fc) = \deg(f) + \deg(c) < (n - 2t) + t = n - t$
- $\deg(A) = n$, $\deg(B) = n - 1 < \deg(A)$

In other words, (c, d) are degree-t approximants of (A, B). By Theorem 3.2, we know that $(c, d) = \lambda(a, b)$ for some non-zero $\lambda \in K[x]$ and (a, b) are the best degree-t approximants of (A, B). This means that $a \mid c$, and since $c \mid A$, it naturally follows that $a \mid A$.

Substituting (c, d) with $\lambda(a, b)$ gives us the desired results:

$$f = B - bA/a$$

□

Theorem 3.3. *$r_i \neq f(\alpha_i)$ if and only if $a(\alpha_i) = 0$*

Proof. TODO: prove this theorem □

This is in fact, a Reed-Solomon decoder. In Reed-Solomon code, the set of symbols is the set of degree-$n - 2t$ polynomials, encoding involves evaluating the polynomial at n distinct points. If there are no more than t errors, then applying algorithm 3.3 will allow the decoder to recover the polynomial.

3.4 Goppa decoding

In this section we restrict K to be finite field of characteristic 2 ($K = \mathbb{F}_{2^m}$).

Each binary Goppa code is parameterized by n distinct field elements $\alpha_1, \alpha_2, \ldots, \alpha_n \in K$ (called **field ordering**) and a degree-t irreducible polynomial $g \in K[c]$ such that $g(\alpha_i) \neq 0$ for all $1 \leq i \leq n$. The set of code words is defined as follows:

$$\mathcal{C} = \{\mathbf{c} \in \mathbb{F}_2^n \mid \sum_{i=1}^{n} \frac{c_i A}{x - \alpha_i} \equiv 0 \mod g\}$$

where $A = \prod_{i=1}^{n}(x - \alpha_i)$.

3.5 Parity check matrix

4 Implementation details

References

[McE78] Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244:114–116, 1978.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.