A McEliece-type cryptosystem based on convolutional codes

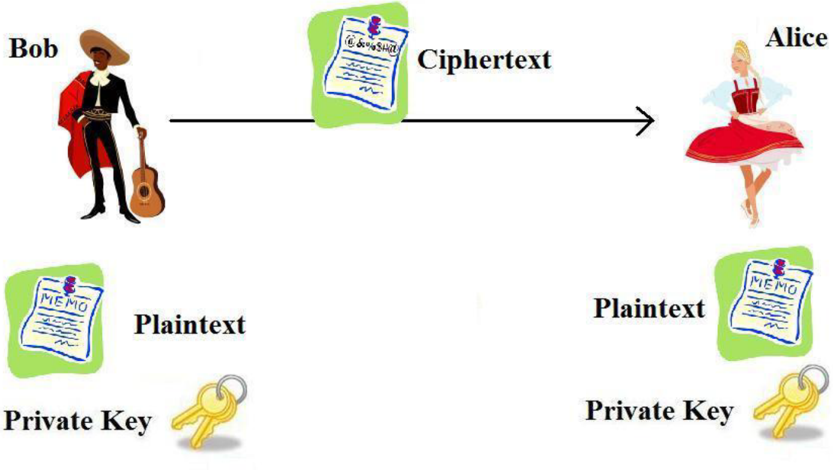
Paulo Almeida

Quantum Days

April 12, 2019

Overview

1. Motivation: Quantum Resistant Cryptography
2. Coding Theory
3. Finite Fields
4. Code-based cryptography: The original McEliece construction
5. New variant based on convolutional codes
6. Attacks
   - Comparisons

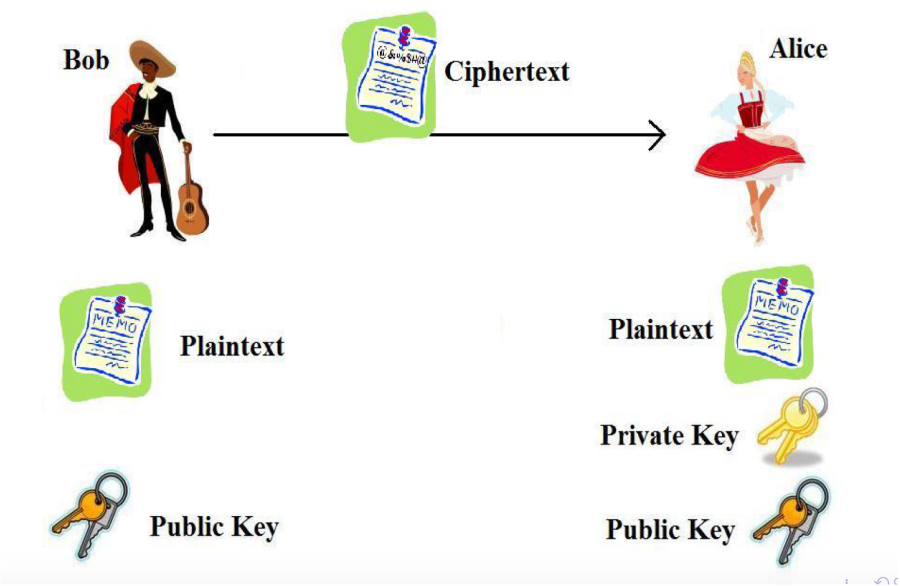

Bob
Ciphertext
Alice
Plaintext
Private Key
Plaintext
Private Key

- A critical need:

- A critical need: many machines needs to communicate securely among themselves. Prior secret key exchange is not feasible.

- A critical need: many machines needs to communicate securely among themselves. Prior secret key exchange is not feasible.
- A breakthrough: symmetric vs asymmetric - Diffie Hellmann key exchange

- A critical need: many machines needs to communicate securely among themselves. Prior secret key exchange is not feasible.
- A breakthrough: symmetric vs asymmetric - Diffie Hellmann key exchange
- Public key cryptography : the foundation of our modern communications systems, secure communication and signatures.


Bob
Ciphertext
Alice
MEMO
Plaintext
Plaintext
MEMO
Private Key
Public Key
Public Key

- Factoring integers is "*hard*" → RSA

- Factoring integers is "*hard*" → RSA

- Finding discrete logarithms is "*hard*" → ElGamal

- Factoring integers is "*hard*" → RSA

- Finding discrete logarithms is "*hard*" → ElGamal

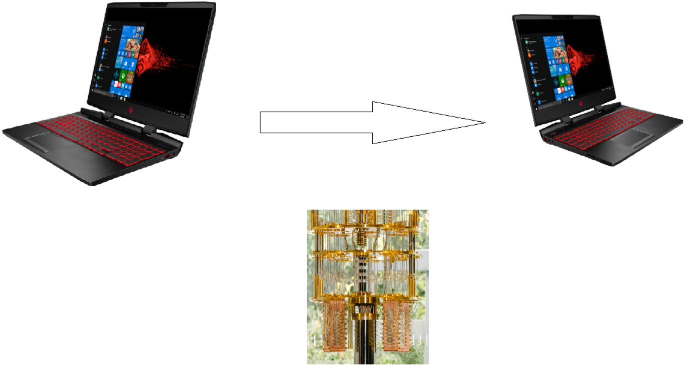
- Quantum computer + Shor's algorithm → End of RSA and systems based on DLP (ElGamal, Elliptic Curve Cryptography).

Motivation

"The era of fully fledged quantum computers threatens to destroy internet security as we know it. Researchers are in a race against time to prepare new cryptographic techniques before the arrival of quantum computers..."

Tanja Lange (cryptographer)

Post-Quantum Cryptography

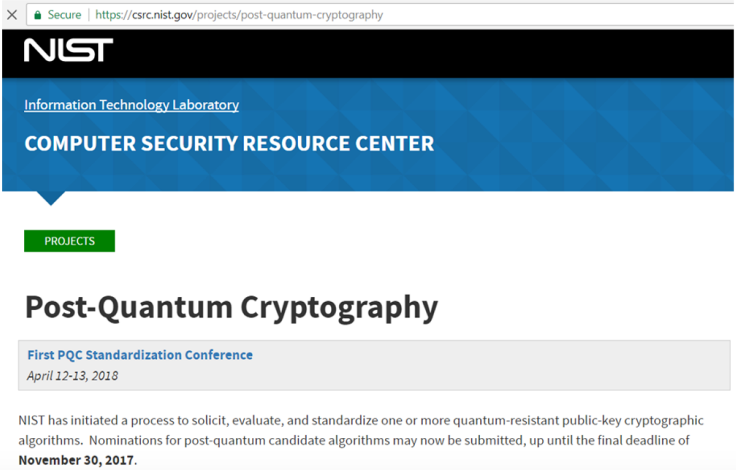


International Security Agencies are looking for Quantum Resistant Cryptography

- In August 2015 the National Security Agency (NSA) released a major policy statement on the need for Post-Quantum cryptography.

- ”Unfortunately, the growth of elliptic curve use has bumped up against the fact of continued progress in the research on quantum computing, necessitating a re-evaluation of our cryptographic strategy.”
- For those partners and vendors that have not yet made the transition to Suite B algorithms, we recommend not making a significant expenditure to do so at this point but instead to prepare for the upcoming quantum resistant algorithm transition.”

Quantum Computers - National Institute of Standards and Technology (NIST)



Secure | https://csrc.nist.gov/projects/post-quantum-cryptography
NIST
Information Technology Laboratory
COMPUTER SECURITY RESOURCE CENTER
PROJECTS

Post-Quantum Cryptography

First PQC Standardization Conference

April 12-13, 2018

NIST has initiated a process to solicit, evaluate, and standardize one or more quantum-resistant public-key cryptographic algorithms. Nominations for post-quantum candidate algorithms may now be submitted, up until the final deadline of **November 30, 2017**.

- Cryptosystems based on error-correcting codes (Code-based Cryptography) are one of the most promising post-quantum alternatives for Public Key Cryptography: the McEliece cryptosystem.
- McEliece cryptosystem is one of the oldest public-key cryptosystems that is immune to existing quantum computer algorithm attacks.
- It was introduced in 1978 by Robert McEliece.
- It is based on the hardness of finding the nearest codeword for a general linear binary code, which is considered a NP-hard problem.
- The original McEliece cryptosystem is based on binary Goppa codes.

Detour - Computational Complexity

Addition

$$\begin{array}{r} {\scriptstyle 1111} \\ 1111000 \\ \underline{+0011110} \\ 10010110 \end{array}$$

Suppose both numbers have k bits. For each of the k columns we do one (one bit operation). So we have $O(k) = O(\log(n))$ bit operations.

Product

For example:

$$\begin{array}{r} 11101 \\ \underline{\times 1101} \\ 00011101 \\ 01110100 \\ \underline{11101000} \\ 101111001 \end{array}$$

In order to perform the product of a number n with k bits by a number m with l bits, we obtain at most l rows, where each row has a copy of the number n shift to the right by a certain distance.

Each row has at most $k + l - 1$ bits (add zeros after each copy of n). So, at most we perform $l - 1$ additions of two numbers each of which has at most $k + l - 1$ bits.

If $k \geq l$ the number of bit operations is $O(k) = O(\log^2(n))$.

Class P

A algorithm that performs a calculation (or answer to a question) involving integers $n_1 \geq \cdots \geq n_r$ with $k_1, \ldots, k_r$ bits, respectively, is a *polynomial time algorithm* if exists an integer d_1 such that

$$\text{number of bit operations} = O(k_1^{d_1}) = O(\log^{d1} n_1).$$

One operation or task has polynomial time if there is a polynomial time algorithm that executes that operation. In this case we say that the operation is in the *class P*.

Addition, product, square roots, cubic roots, Euclidean algorithm and primality testing are all in the class P.

Class NP

If the check to the solution of a certain question can be performed in polynomial time, we say that this question is in the *class NP*.
The factorization of integers is in the class NP, since given n, p and q positive integers, we can check if $n = pq$ by performing the product of p and q, which is a polynomial time algorithm.
Hence, factorization is in the class NP.
Clearly P$\subset$NP.
For classic computers there is still no polynomial time algorithm for the factorization of integers, so factorization may not be in class P.
The Shor's algorithm for quantum computers is in the class BQP (bounded error quantum polinomial).
Therefore, with quantum computers factorization is fast.

Linear Codes

When a message is transmitted over a channel, many errors can occur, such as errors provoked by dust and scratches in a CD, noise in high frequency transmission in mobile phones, or thermal noise in the receiver in deep space communications.

If, for every block of a message we add some well chosen bits obtained from that block, we can detect if an error occurred and even correct it.

Linear Codes

Let $u = x_1x_2x_3x_4$ represents a four bits message. Let $x_5 = x_2 + x_3 + x_4$, $x_6 = x_1 + x_3 + x_4$ and $x_7 = x_1 + x_2 + x_4$. The sequence $x_1x_2x_3x_4x_5x_6x_7$ is a *codeword*. This is called a $[7, 4]$-code, since for each 4 bits of message, it is transmitted 7 bits, where the last 3 are verification bits.

Examples of codewords are $1000011, 0100101, 0010110$ and 0001111. All the other codewords are linear combinations of these codewords.

Notice that there are only 16 possible codewords, but 128 possible 7-bit sequences. The minimum number of nonzero coordinates in the difference between any two codewords is 3 (i. e. the minimum distance of the $[7, 4]$-code is 3). So, we call this a $[7, 4, 3]$-code.

Linear Codes

The word 0101 is transmitted as 0101010, so, if we receive 1101010 we know that an error occurred and we may assume that the original message was "0101", since this corresponds to the closest codeword.

The aim of coding theory is to find codes which

- transmit quickly;
- contain many valid codewords;
- can correct or at least detect many errors.

The $[7,4,3]$-code was invented by Richard W. Hamming in 1950, while working at Bell Telephone Laboratories and was the first ever code to be used. Hamming used it for the punched card reader of Bell Model V.

Linear Codes

A Reed-Muller binary (32,6,16)-code was used to transmit black and white images during the Mariner 9 mission in 1972.

The Voyager 1 and 2 spacecraft needed to transmit hundreds of colour pictures of Jupiter and Saturn in their 1979, 1980, and 1981 fly-by's within a constrained telecommunications bandwidth.

Color image transmission required three times the amount of data as black and white images, so the RM (32,6,16)-code that was used to transmit the black and white images was switched to the Golay (24,12,8)-code.

This Golay code is only triple-error correcting, but it could be transmitted at a much higher data rate.

Linear Codes- Generator Matrix

One way of defining a code is through its *generator matrix*. For example the generator matrix of the $[7,4,3]$-code is

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

So if the message is $u = x_1x_2x_3x_4$ then uG is the corresponding codeword. Given a code $\mathcal{C}$ and G its generator matrix, we have $\mathcal{C} = \mathrm{Im}\mathcal{G}$.

Linear Codes- Parity Check Matrix

Another way of defining a code is through its *parity check matrix*. For example the parity check matrix of the $[7,4,3]$-code is

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}.$$

The codewords are the vectors v such that $Hv = 0$. Notice that $GH^T = 0$. So a code is the subspace $\ker(H)$.

Finite Fields

A *finite field* (also called *Galois field*) $\mathbb{F}_q$ has $q = p^m$ elements, where p is prime and $m \geq 1$ and is obtained from a *primitive* polynomial over $\mathbb{F}_p$ of degree m.
A primitive polynomial is a irreducible polynomial that has a root of order $p^m - 1$.

Example

Consider $p = 2$ and $f(x) = x^3 + x + 1$. Then $\mathbb{F}_2 = \{0, 1\}$ and clearly $f(x)$ is irreducible over $\mathbb{F}_2$ because 0 and 1 are not roots of $f(x)$. Now take α such that $f(\alpha) = 0$. Then $\alpha^3 = \alpha + 1$. So we construct $\mathbb{F}_8$ using this polynomial as follows: Its elements are

$$0, 1, \alpha, \alpha + 1, \alpha^2, \alpha^2 + 1, \alpha^2 + \alpha, \alpha^2 + \alpha + 1.$$

Finite Fields

We can write all nonzero elements as a power of α, and as vectors of $\mathbb{F}_2^3$, namely

α^0	$=$	1	$=$	100	α^1	$=$	α	$=$	010
α^2	$=$	α^2	$=$	001	α^3	$=$	$1+\alpha$	$=$	110
α^4	$=$	$\alpha^2+\alpha$	$=$	011	α^5	$=$	$\alpha^2+\alpha+1$	$=$	111
α^6	$=$	α^2+1	$=$	101			0	$=$	000

Notice that $\alpha^7 = 1$.

The original construction

Key generation

Let be $\mathcal{C}$ a $[n, k, 2t+1]$ linear code that has an efficient decoding algorithm that can correct up to t errors.

- Compute a $k \times n$ generator matrix G for $\mathcal{C}$;
- Generate a random $k \times k$ invertible matrix S;
- Generate a random $n \times n$ permutation matrix P;
- Compute the $k \times n$ matrix $G' = SGP$.

The *public key* is (G', t) and the *private key* is (S, G, P).

Encryption

To encrypt a message m of length k:

- Compute mG' and add a random error vector e of weight t and length n;
- Send $m' = mG' + e$.

Decryption

To decrypt m':

- Compute $m'P^{-1} = (mS)G + eP^{-1}$;
- Since $(mS)G$ is a codeword in $\mathcal{C}$ and the permuted error vector eP^{-1} has weight t, the decoding algorithm $\mathcal{D}$ can be applied to $m'P^{-1}$ to obtain mS;
- Calculate m using S^{-1}.

Advantages

- There exists a fast algorithm to encrypt and decrypt the message (compared, for instance, with the RSA).
- No effective quantum algorithm is known to break McEliece cryptosystem - is one of the most trusted candidates for post-quantum cryptography.

Disadvantages

- Large keys and low transmission rate.

- Many variants of McEliece cryptosystem have been proposed to replace Goppa codes, in order to reduce the size of public key.

Table: Some variants of McEliece using other types of codes

Type of code	Proposed by	Current status
Binary Goppa Codes	R.McEliece, 1978	Unbroken as 2018
Generalized Reed-Solomon (GRS) codes	H.Niederreiter, 1986	Broken in 1992
Maximum rank distance (MRD) codes	E.Gabidulin et al., 1991 and 1993	Broken in 1994 and 1996
Reed-Muller codes	V.M. Sidelnikov, 1994	Broken in 2007
Quasi-cyclic low density parity-check codes	M.Baldi et al, 2007	Broken in 2008

- But most proposed systems were broken.

Proposals based on convolutional codes

- In 2012 the first McEliece system based on convolutional codes was proposed by Londahl and Johansson.
- It was been broken by Landais and Tillich in 2013.
- In 2017, Hamza Moufek and Kenza Guenda proposed a new variant McEliece cryptosystem based on the Smith form of convolutional codes.

New Proposals - Submissions to NIST

- Tillich and others proposed varius code based cryptosystems namely
  BIG QUAKE based on quasi-cyclic binary Goppa codes;
  BIKE based on quasi-cyclic Moderate Density Parity-Check codes;
- Bernstein and others proposed a variation of the original McEliece cryptosystem.
- Barreto and others proposed DAGS based on quasi-Dyadic generalized Srivastava codes;
- HQC uses quasi-cyclic Hamming codes;
- LEDAkem and LEDApkc uses quasi-cyclic low density parity check codes;
- RLCE-KEM is based on random linear codes;
- RQC is based on rank quasi-cyclic codes;

Just to cite a few...

New variant considering convolutional codes

Our Idea (Jorge Brandão, Cláudia Sebastião, Diego Napp & Paulo Almeida

Create a system that uses in the public key, convolutional codes.

Notation

- $\mathbb{F} = \mathbb{F}_q$ be a finite field of size q
- $\mathbb{F}((D))$ be the field of formal Laurent series
- $\mathbb{F}(D, D^{-1})$ be the ring of Laurent polynomials
- $\mathbb{F}(D)$ is the field of rational polynomials
- $\mathbb{F}[D]$ be the ring of polynomials

Notice that

$$\mathbb{F}[D] \subseteq \mathbb{F}(D, D^{-1}) \subseteq \mathbb{F}(D) \subseteq \mathbb{F}((D)).$$

A *convolutional code* $\mathcal{C}$ of rate k/n is an $\mathbb{F}((D))$-subspace of $\mathbb{F}((D))^n$ of dimension k given by a rational *encoder matrix* $G(D) \in \mathbb{F}^{k \times n}(D)$,

The ingredients of our variant

1. $G \in \mathbb{F}^{k\times n}$ be an encoder of an a $[n, k, 2t+1]$-block code that has an efficient decoding algorithm that can correct up to t errors.
2. $$T(D, D^{-1}) = \sum_{i=-\mu}^{\mu} T_i D^i \in \mathbb{F}^{n\times n}(D, D^{-1}) = P^{-1}(D, D^{-1})$$
   a) its determinant is in $\mathbb{F}$;
   b) the positions of the nonzero columns of T_i form a partition of n;
   c) each row of T_i has at most one nonzero element, for $i = -\mu, \dots, 0, \dots, \mu$;
3. $$S(D) = \sum_{i=\mu}^{\nu} S_i D^i \in \mathbb{F}^{k\times k}[D] \tag{1}$$

with $S_\mu \in \mathbb{F}^{k\times k}$ an invertible constant matrix.

The public key

$$G'(D) = S(D)GP(D, D^{-1}).$$

Secret key

$$\{S(D), G, P(D, D^{-1})\}$$

$$S(D) = \begin{bmatrix} D & D^5+D^4+D^3 & D^4 & D^5 \\ 0 & 0 & D^2 & D \\ 0 & D & 0 & 0 \\ 0 & 0 & D & 0 \end{bmatrix}, \qquad G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$T(D, D^{-1}) = \begin{bmatrix} D & 0 & D^{-1} & 0 & 0 & 0 & 0 \\ 0 & D & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ D & 0 & 0 & D^{-1} & 0 & 0 & 0 \\ 0 & D & 0 & D^{-1} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & D & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \qquad P(D, D^{-1}) = \begin{bmatrix} 0 & 0 & 0 & D^{-1} & -D^{-1} & D^{-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{D} \\ D & 0 & 0 & -D & D & -D & 0 \\ 0 & 0 & 0 & -D & D & -D & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

$$G'(D) = S(D)GP(D, D^{-1}) =$$
$$\begin{bmatrix} D^5 & D^3 & D^4+D^3+D & D^5+1 & D^6+D^5+1 & D^6+D^4+D+1 & D^5+D^2+D \\ D^3 & D^2+D & D & D^3 & D^3+D^2 & D^3+D & 0 \\ 0 & D & D & 0 & 0 & 0 & D+1 \\ D^2 & D & 0 & D^2 & D^2 & D^2+D & 0 \end{bmatrix}.$$

$$\begin{bmatrix} D & D^5+D^4+D^3 & D^4 & D^5 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Encryption

Alice selects an error vector $\mathbf{e}(D)$ satisfying

$$\mathbf{wt}((\mathbf{e}_i, \mathbf{e}_{i+1}, \ldots, \mathbf{e}_{i+2\mu})) \leq t$$

and encrypts the message $\mathbf{u}(D) = \mathbf{u}_0 + \mathbf{u}_1 D + \mathbf{u}_2 D^2 + \cdots + \mathbf{u}_\ell D^\ell \in \mathbb{F}[D]^k$ as

$$\mathbf{y}(D) = \mathbf{u}(D) G'(D, D^{-1}) + \mathbf{e}(D).$$

then sends the ciphertext $\mathbf{y}(D) = \sum_{i \geq 0} \mathbf{y}_i D^i$.

Decryption

Bob multiplies $\mathbf{y}(D)$ by the matrix $T(D, D^{-1})$ and obtains

$$\begin{aligned} \mathbf{y}(D)T(D, D^{-1}) &= \mathbf{u}(D)S(D)G + \mathbf{e}(D)T(D, D^{-1}) \\ &= \widehat{\mathbf{u}}(D)G + \widehat{\mathbf{e}}(D) \end{aligned}$$

decode using G and finally recover the message $\mathbf{u}(D)$ from $\mathbf{u}(D)S(D)$.

Remarks

Each row of T_i has at most one nonzero element, then $\mathbf{wt}(\mathbf{e}_j) \geq \mathbf{wt}(\mathbf{e}_j T_i)$ for all $j \geq 0$ and $-\mu \leq i \leq \mu$. Take $\mathbf{e}_j = 0$ for $j < 0$, then $\widehat{\mathbf{e}}_s \leq t$ since

$\widehat{\mathbf{e}}_s = \sum_{i=-\mu}^{\mu} \mathbf{e}_{s-i} T_i$, for $s \geq -\mu$

Further, one can retrieve $\mathbf{u}(D)$ from

$$\mathbf{u}(D)S(D) = \sum_{i=\mu}^{\ell+\nu} \widehat{\mathbf{u}}_i D^i$$

$$\begin{bmatrix} \widehat{\mathbf{u}}_{\mu} & \widehat{\mathbf{u}}_{\mu+1} & \cdots & \widehat{\mathbf{u}}_{\ell+\nu} \end{bmatrix} =$$

$$= \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \cdots & \mathbf{u}_\ell \end{bmatrix} \underbrace{\begin{bmatrix} S_\mu & S_{\mu+1} & \cdots & S_\nu & & & \\ & S_\mu & S_{\mu+1} & \cdots & S_\nu & & \\ & & \ddots & \ddots & \ddots & \ddots & \\ & & & S_\mu & \cdots & & S_\nu \\ & & & & \ddots & & \vdots \\ & & & & & \ddots & \vdots \\ & & & & & & S_\mu \end{bmatrix}}_{= \quad S_{truc}(\ell)},$$

and the matrix $S_{truc}(\ell)$ is invertible as S_μ is invertible.

The ciphertext is generated as $\mathbf{y}(D) = \mathbf{u}(D)G'(D) + \mathbf{e}(D)$ or equivalently,

$$\left[\begin{array}{cccccc} \mathbf{y}_0 & \mathbf{y}_1 & \cdots & \mathbf{y}_\ell & \cdots & \mathbf{y}_{\ell+\nu+\mu} \end{array}\right] \tag{2}$$

is equal to the multiplication of

$$\left[\begin{array}{cccc} \mathbf{u}_0 & \mathbf{u}_1 & \cdots & \mathbf{u}_\ell \end{array}\right] \tag{3}$$

with

$$\begin{bmatrix} G'_0 & G'_1 & \cdots & G'_{\nu+\mu} & & & & & \\ & G'_0 & G'_1 & \cdots & G'_{\nu+\mu} & & & & \\ & & \ddots & \ddots & & \ddots & & & \\ & & & G'_0 & G'_1 & \cdots & G'_{\nu+\mu} & & \\ & & & & \ddots & \ddots & & \ddots & \\ & & & & & G'_0 & G'_1 & \cdots & G'_{\nu+\mu} \end{bmatrix} \tag{4}$$

and adding the error vector

Advantages

- Fast encryption and decryption (decoding complexity is $O(n^2\mu)$)
- We can process the information *sequentially*
- The first components of G' are not full row rank and so our scheme seems to be invulnerable to the attacks against previous McEliece cryptosystems based on convolutional codes.

Attacks: General ideas

We are particularly interested in the security of the following interval of data $\begin{bmatrix} \mathbf{y}_0 & \mathbf{y}_1 & \cdots & \mathbf{y}_s \end{bmatrix}$ which is equal to

$$\begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \cdots & \mathbf{u}_s \end{bmatrix} \begin{bmatrix} G_0' & G_1' & \cdots & G_s' \\ & G_0' & \cdots & G_{s-1}' \\ & & \ddots & \vdots \\ & & & G_0' \end{bmatrix} + \begin{bmatrix} \mathbf{e}_0 & \mathbf{e}_1 & \cdots & \mathbf{e}_s \end{bmatrix}$$

Structural Attacks

Structural attacks seem difficult as large part of the public key are generated randomly.

Information Set Decoding Attacks

The truncated sliding matrix is not an encoder, i.e., is not full rank ⇒ The work factor grows exponentially with the rank deficiency.

Work Factor

	Binary Goppa	Our proposal
k (dimension)	2288	45
n (length)	2960	63
q (field)	2048	64
m (memory)	0	99
Key size	1537536	340200
WF (Work Factor)	2^{128}	2^{301}

Acknowledgements:

This work was supported in part by the Portuguese Foundation for Science and Technology (FCT-Fundação para a Ciência e a Tecnologia), through CIDMA - Center for Research and Development in Mathematics and Applications, within project UID/MAT/04106/2019

We thank the organizers for such a nice meeting