Making McEliece and Regev meet

Gilles Zémor
based on common work with C. Aguilar, O. Blazy, J-C. Deneuville, P. Gaborit

Bordeaux Mathematics Institute

March 21, 2019, Oberwolfach

the McEliece paridigm

Choose a code C that comes with a decodable algorithm, and publish a *random* generator matrix $\mathbf{G}$.

trapdoor encryption primitive:

$$\begin{array}{rcl} \mathcal{M} = \{0,1\}^m & \to & \{0,1\}^n \\ \mathbf{m} & \mapsto & \mathbf{mG} + \mathbf{e} \end{array}$$

for $\mathbf{e}$ random vector of small weight t.

Public matrix $\mathbf{G}$ should "look like" generator matrix of random code.

Decrypt with hidden decoding algorithm.

Historical instantiation: use a random Goppa code for C.

MDPC codes

Modern variant Misoczki, Tillich, Sendrier, Barreto 2012. Use for C a Moderate Density Parity-Check code.

$$\mathbf{H} = \begin{bmatrix} 111100 & \cdots & 000 & \cdots & 000 \\ & & \vdots & & \\ & & & & \end{bmatrix}$$

Codewords $\mathbf{x} = [x_1, \ldots, x_n]$ satisfy (somewhat) low-weight parity-check equations $\sigma(\mathbf{x}) = \mathbf{H}\mathbf{x}^T = 0$

$$x_3 + x_7 + x_{23} = 0$$

If received vector $\mathbf{y}$ satisfies, say:

$$y_3 + y_7 + y_{23} = 1$$
$$y_3 + y_5 + y_{11} = 1$$

then flip the value of y_3.

Decoding MDPC codes

Bit flipping algorithm: if flipping the value of a bit decreases the syndrome weight, then flip its value. Repeat.

The higher the weight w of the parity-checks, the lower the weight t of decodable error vectors: $wt \leq n$

On the other hand, the lower the weight w of the parity-checks, the easier it is to recover them from an arbitrary parity-check matrix of the code. Method: guess $n/2$ coordinates that are 0. Cost: 2^w.

Same algorithm as Information Set Decoding for random codes. Decoding t errors similarly costs 2^t guesses.

Meet in the middle. Choose $w = t \approx \sqrt{n}$.

the Alekhnovich cryptosystem

Public: random matrix **H**, together with vector **y**

$$\mathbf{H} = \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$
$$\mathbf{y} = \begin{bmatrix} \quad \mathbf{sH} + \varepsilon \quad \end{bmatrix}$$

Encryption of $m \in \mathbb{F}_2$, output $\mathcal{C}(m)$ equal to:

- if $m = 0$: uniform random vector **u** of $\mathbb{F}_2^n$
- if $m = 1$: vector $\mathbf{c} + \mathbf{e}$ where **e** of weight t and **c** codeword of code define by parity-check matrix **H and y.**

Notice: $\langle \mathbf{c} + \mathbf{e}, \varepsilon \rangle = \langle \mathbf{e}, \varepsilon \rangle$, probably 0 if **e** and ε of small enough weight.

So **decryption**: compute $\langle \mathcal{C}(m), \varepsilon \rangle$. If 0 declare $m = 1$ otherwise declare $m = 0$. Correct $\sim 3/4$ of the time.

Security

$$\mathbf{H} = \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} \mathbf{sH} + \varepsilon \end{bmatrix}$$

Assumption: difficult to distinguish whether **y** is

- random at distance t from code generated by rows of **H**,
- uniformly random.

Reduces to difficulty of decoding random codes.

Security argument:

- Attacker must continue to decrypt when **y** is uniformly random,
- and when $\mathbf{c} + \mathbf{e}$ is replace by uniformly random vector.

But then decryption is exactly the decision problem: our asymption says exactly that it is not possible to solve.

Reducing to decoding random codes

$$\mathbf{H} = \left[\quad \right]$$
$$\mathbf{y} = \left[\quad \mathbf{sH} + \varepsilon \quad \right]$$

Ingredients:

Trick: if you can solve the decision (guessing) problem, you have a device that, given $\mathbf{y} = \mathbf{sH} + \varepsilon$, computes, for any choice of r, $\langle s, r \rangle$ better than $(1/2, 1/2)$-guessing.

Accessing **s** now becomes the decoding problem from a noisy codeword of a Reed-Muller code of order 1. Possible in sub-linear time. Goldreich-Levin theorem.

Regev version (binary)

Public: random matrix **H**, together with vector **y**

$$\begin{array}{ll} \mathbf{H} = & \left[\quad \quad \quad \right] \\ \mathbf{y} = & \left[\quad \mathbf{sH} + \varepsilon \quad \right] \end{array}$$

Encryption of $m \in \mathbb{F}_2$, output

$$\mathcal{C}(m) = (\sigma(\mathbf{e}) = \mathbf{He}^T, \mathbf{z} = m + \langle \mathbf{e}, \mathbf{y} \rangle)$$

for **e** random of small weight t.

Decryption:

$$\mathbf{z} + \langle \mathbf{s}, \sigma(\mathbf{e}) \rangle = m + \langle \mathbf{e}, \varepsilon \rangle.$$

Both **e** and ε of weight $< \sqrt{n}$.

Vector version

Public: random matrix **H** and $\ell \times n$ matrix **Y**. Auxilliary code $C \subset \mathbb{F}_2^k$.

$$\mathbf{H} = \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} \mathbf{SH} + \mathbf{E} \end{bmatrix}$$

Encryption of $\mathbf{m} \in C \subset \mathbb{F}_2^\ell$, output

$$\mathcal{C}(m) = (\sigma(\mathbf{e}) = \mathbf{He}^T, \mathbf{z} = \mathbf{m} + \mathbf{Ye}^T)$$

for $\mathbf{e} \in \mathbb{F}_2^n$ random of small weight $t < \sqrt{n}$.

Decryption:

$$\mathbf{z} + \mathbf{S}\sigma(\mathbf{e})^T = \mathbf{m} + \mathbf{Ee}^T.$$

Security argument: same.

Variation: Alekhnovich meets MDPC-McEliece

Public: random matrix $\begin{bmatrix} \mathbf{H} \\ \mathbf{Y} \end{bmatrix}$. No auxiliary code.

$$\mathbf{H} = \begin{bmatrix} \\ \\ \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} \mathbf{SH} + \mathbf{E} \end{bmatrix}$$

C code whose parity-check matrix is $\begin{bmatrix} \mathbf{H} \\ \mathbf{Y} \end{bmatrix}$. Generator matrix **G**.

Encryption primitive: $\mathbf{m} \mapsto \mathcal{C}(\mathbf{m}) = \mathbf{mG} + \mathbf{e}$
for **e** vector of low weight t.

Decryption: compute $\mathbf{E}\mathcal{C}(\mathbf{m})^T$, the **E**-syndrome of $\mathcal{C}(\mathbf{m})$. Equal to $\mathbf{Ee}^T$. Use bit-flip (MDPC) decoding !

Reduces to MDPC-McEliece when $\mathbf{H} = 0$.

Towards greater efficiency, double-circulant codes

Codes with parity-check (or generator) matrices of the form

$$\mathbf{H} = \left[\quad \mathbf{I}_n \quad | \quad \mathrm{rot}(\mathbf{h}) \right].$$

Equivalently, code invariant by simultaneous cyclic shifts of coordinates $1 \cdots n$ and $n+1 \cdots 2n$.

Long history. Hold many records for minimum distance. Above GV bound (by a non-exponential factor), [Gaborit Z. 2008].

No known decoding algorithm improves significantly over decoding random codes. As for wider class of *quasi-cyclic* codes.

Boosts MDPC-McEliece. Use double-circulant MDPC code. Defined by a vector $\mathbf{h}$, means needs n bits instead of n^2.

With a random double circulant code

Public key: **G** generator matrix of auxiliary code C of length n.

- $\mathbf{H} = \begin{bmatrix} \mathbf{I}_n & | & \mathrm{rot}(\mathbf{h}) \end{bmatrix}$.
- Syndrome $\boldsymbol{\sigma}$ of a vector $[\mathbf{x}, \mathbf{y}]$ of low weight (t, t).

$$\begin{aligned} \sigma(\mathbf{x}, \mathbf{y}) = \mathbf{H} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} &= \mathbf{x}^T + \mathrm{rot}(\mathbf{h})\mathbf{y}^T \\ &= (\mathbf{x} + \mathbf{h} \cdot \mathbf{y})^T \\ \boldsymbol{\sigma} &= \mathbf{x} + \mathbf{hy} \end{aligned}$$

hy: polynomial multiplication in $\mathbb{F}_2[X]/(X^n + 1)$.

Encryption: $\mathbf{r}_1, \mathbf{r}_2, \varepsilon$ of low weight.

$$(\boldsymbol{\lambda} = \sigma(\mathbf{r}_1, \mathbf{r}_2) = \mathbf{r}_1 + \mathbf{hr}_2, \boldsymbol{\rho} = \mathbf{mG} + \boldsymbol{\sigma}\mathbf{r}_2 + \varepsilon)$$

Decryption:

$$\boldsymbol{\rho} + \boldsymbol{\lambda}\mathbf{y} = \mathbf{mG} + \mathbf{yr}_1 + \mathbf{xr}_2 + \varepsilon.$$

Codeword of C plus (somewhat) small noise.

Security

Public key: regular error-correcting code C,

- $\mathbf{H} = \left[\quad \mathbf{I}_n \quad | \quad \mathrm{rot}(\mathbf{h}) \right]$.
- $\sigma(\mathbf{x}, \mathbf{y}) = \mathbf{H} \left[\begin{smallmatrix} \mathbf{x} \\ \mathbf{y} \end{smallmatrix} \right]$. Attacker must continue to decrypt when $\mathbf{x}, \mathbf{y}$ uniformly random (instead of low-weight).

Encryption:

$$(\boldsymbol{\lambda} = \sigma(\mathbf{r}_1, \mathbf{r}_2) = \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2, \boldsymbol{\rho} = \mathbf{m}\mathbf{G} + \boldsymbol{\sigma}\mathbf{r}_2 + \varepsilon)$$

Rewrite as:

$$\begin{bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\rho} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{m}\mathbf{G} \end{bmatrix} + \begin{bmatrix} \mathbf{I}_n & 0 & \mathrm{rot}(\mathbf{h}) \\ 0 & \mathbf{I}_n & \mathrm{rot}(\boldsymbol{\sigma}) \end{bmatrix} \begin{bmatrix} \mathbf{r}_1 \\ \varepsilon \\ \mathbf{r}_2 \end{bmatrix}.$$

So attack must continue to work when $\mathbf{r}_1, \mathbf{r}_2, \varepsilon$ are also replaced by uniform. Otherwise we can distinguish between uniform and uniform of small distance from triple circulant quasi-cyclic code.

Note that presence of noise vector ε is *essential*.

New idea

Vector ε important for security argument, but otherwise underused. Why not use it to carry information ?

Decoder knows $\mathbf{x}, \mathbf{y}$, so low-weight $\mathbf{r}_1, \mathbf{r}_2$ can be recovered from

$$\mathbf{x}\mathbf{r}_2 + \mathbf{y}\mathbf{r}_1 = \begin{bmatrix} \mathrm{rot}(\mathbf{x}) & \mathrm{rot}(\mathbf{y}) \end{bmatrix} \begin{bmatrix} \mathbf{r}_2 \\ \mathbf{r}_1 \end{bmatrix}$$

and from

$$\mathbf{x}\mathbf{r}_2 + \mathbf{y}\mathbf{r}_1 + \varepsilon = \begin{bmatrix} \mathrm{rot}(\mathbf{x}) & \mathrm{rot}(\mathbf{y}) & \mathbf{I}_n \end{bmatrix} \begin{bmatrix} \mathbf{r}_2 \\ \mathbf{r}_1 \\ \varepsilon \end{bmatrix}$$

New key-exchange protocol: Ourobouros

- Alice sends $\mathbf{h}$ and $\sigma(\mathbf{x}, \mathbf{y}) = \mathbf{x} + \mathbf{hy}$ for secret $\mathbf{x}, \mathbf{y}$ of low weight.
- Bob sends
  - $\sigma(\mathbf{r}) = \mathbf{r}_1 + \mathbf{hr}_2$ for secret $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2)$ of low weight.
  - $\beta = (\mathbf{x} + \mathbf{hy})\mathbf{r}_2 + \varepsilon + f(\mathrm{hash}(\mathbf{r}))$

  where ε is secret to be exchanged, and f transforms input into (pseudo)-random noise of low weight.
- Alice computes

$$\mathbf{y}(\mathbf{r}_1 + \mathbf{hr}_2) + \beta$$

  which equals

$$\mathbf{xr}_2 + \mathbf{yr}_1 + \varepsilon + \mathbf{e}$$

  which Alice *decodes* to recover $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2)$ from which she accesses exchanged key ε.

Security

Identical argument to previous protocol, namely, once $\mathbf{x}, \mathbf{y}$ are changed to uniform random, then

$$\mathbf{x}\mathbf{r}_2 + \mathbf{y}\mathbf{r}_1 + \mathbf{e}$$

cannot be distinguished from uniform random.

Low weight vector $\mathbf{e} = f(\mathrm{hash}(\mathbf{r}))$ plays exactly the same role that was played before by ε.

The three variants based on quasi-cyclic codes make up the BIKE suite proposal to NIST.

Extension to Rank metric

The rank metric is defined in finite extensions.

Code C is simply $[n, k]$ linear code over $\mathbb{F}_Q = \mathbb{F}_{q^m}$, extension of $\mathbb{F}_q$.

Elements of $\mathbb{F}_Q$ can be seen as m-tuples of elements of $\mathbb{F}_q$.

Norm of an $\mathbb{F}_Q$-vector is simply its rank viewed as an $m \times n$-matrix.

Distance between $\mathbf{x}$ and $\mathbf{y}$ is simply the rank of $\mathbf{x} - \mathbf{y}$.

Decoding problem is NP-hard (under probabilistic reductions, Gaborit Z. 2016).

the Support connection

The support of a word $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ of rank r is a space E of dim r such that $\forall x_i, x_i \in E$.

- how does one recover a word associated to a given syndrome ?
1) find the support (at worst, guess !)
2) solve a system from the syndrome equations to recover the $x_i \in E$.
This is information set decoding.

remark: for Hamming metric, Newton binomial, for rank distance, Gaussian binomial: $\rightarrow$ complexity grows faster.
$\Rightarrow$ rank metric induces smaller parameters for a given complexity.

Low Rank Parity Check Codes

LDPC: parity-check matrix with low weights (ie: small support)
$\rightarrow$ equivalent for rank metric : dual with small rank support

Definition

A Low Rank Parity Check (LRPC) code of rank d, length n and dimension k over $\mathbb{F}_{q^m}$ is a code with $(n-k) \times n$ parity check matrix $\mathbf{H} = (h_{ij})$ such that the sub-vector space of $\mathbb{F}_{q^m}$ generated by its coefficients h_{ij} has dimension at most d. We call this dimension the weight of $\mathbf{H}$.

In other terms: all coefficients h_{ij} of $\mathbf{H}$ belong to the same 'low' vector space $F = \langle F_1, F_2, \cdots, F_d \rangle$ of $\mathbb{F}_{q^m}$ of dimension d.

Concluding comments

- Quasi-cyclic codes need $X^n - 1$ to avoid small factors. $1 + X + \cdots + X^{n-1}$ irreducible.
- In rank metric, $X^n + a$, $a \in \mathbb{F}_q$.
- Lack of Decision to Search reduction.