24TH CONFERENCE ON ACA

Santiago de Compostela

Session on CACTC

Computer Algebra Tales on Goppa Codes and McEliece Cryptography

N. Sayols & S. Xambó

UPC

19/6/2018

Index

- Ingredients of a McEliece cryptosystem (McECS)
- The PyECC CAS
- Construction of McECS
- Security analysis and the post-quantum scenario
- Code samples
- Conclusions, discussion and future outlook

- $F = F_q$, a finite field of cardinal q (*base field*). The most important case will be $F = \mathbb{Z}_2$. N
- k a positive integer. The vectors of F^k are called *information vectors*, or *messages*.
- $n > k$ an integer. The vectors of F^n are called *transmission vectors*. If $\boldsymbol{x} \in F^n$, we let $|\boldsymbol{x}|$ denote the number of non-zero components of $\boldsymbol{x}$ and we say that it is the *weight* of $\boldsymbol{x}$.

Notations. $F(r, s)$ denotes the space of matrices of type $r \times s$ with entries in F and $F(r) = F(r, r)$.

A receiving user needs the following data:

- $G \in F(k, n)$ such that $\operatorname{rank}(G) = k$;
- $S \in F(k)$ invertible and chosen at random;
- $P \in F(n)$ a random permutation matrix;
- t, a positive integer; and
- $g : X \to F^k$, $X \subseteq F^n$, such that for any $\boldsymbol{u} \in F^k$ and all $\boldsymbol{e} \in F^n$ with $|\boldsymbol{e}| \leq t$,

$$\boldsymbol{x} = \boldsymbol{u}G + \boldsymbol{e} \in X \;\text{ and }\; g(\boldsymbol{x}) = \boldsymbol{u}. \tag{1}$$

The map g is called a t-error-correcting *G-decoder*, or simply *decoder*, and the vectors of X are said to be *g-decodable*.

- Private key: $\{G, S, P\}$
- Public key: $\{G', t\}$, where $G' = SGP$.

Encryption protocol

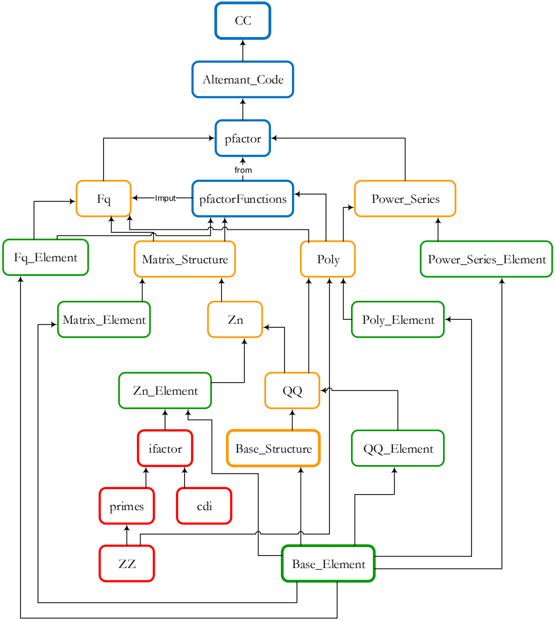
The protocol that a user has to follow to encript and send a message $\boldsymbol{u}$ to the user whose public key is $\{G', t\}$ consists of two steps:

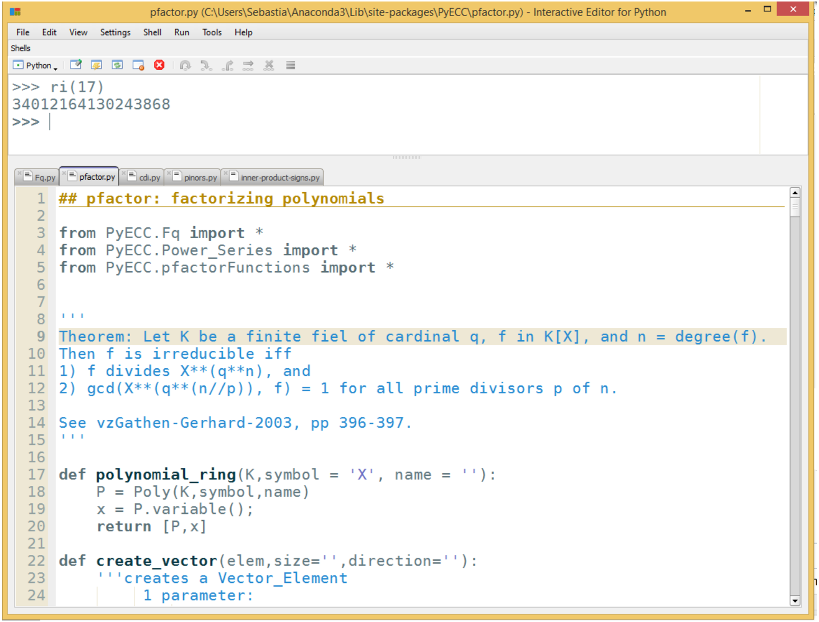
- Random generation of a transmission vector $\boldsymbol{e}$ of weight t;
- Sending the vector $\boldsymbol{x} = \boldsymbol{u}G' + \boldsymbol{e} = \boldsymbol{u}SGP + \boldsymbol{e}$ to that user.

Decryption protocol

Consists of four steps that only use private data of the receiver and the vector $\boldsymbol{x}$ sent by the emitter:

- Set $\boldsymbol{y} = \boldsymbol{x}P^{-1}$, so that $\boldsymbol{y} = (\boldsymbol{u}S)G + \boldsymbol{e}P^{-1}$.
- Set $\boldsymbol{x}' = g(\boldsymbol{y})$. Since P is a permutation matrix,
$$|\boldsymbol{e}P^{-1}| = |\boldsymbol{e}| = t,$$
and hence $\boldsymbol{x}'$ is well defined, as g corrects t errors. The result is $\boldsymbol{x}' = (\boldsymbol{u}S)G$, which says that $\boldsymbol{x}'$ is the linear combination of the rows of G with coefficients $\boldsymbol{u}' = \boldsymbol{u}S$.
- Since G has rank k, $\boldsymbol{u}'$ is uniquely determined by $\boldsymbol{x}'$ and can be obtained by solving the system of linear equations $\boldsymbol{x}' = \boldsymbol{u}'G$, where $\boldsymbol{u}'$ is the unknown vector.
- Let $\boldsymbol{u} = \boldsymbol{u}'S^{-1}$, which agrees with the message sent by the emitter.


CC
Alternant_Code
pfactor
from
Fq
Imput
pfactorFunctions
Power_Series
Fq_Element
Matrix_Structure
Poly
Power_Series_Element
Matrix_Element
Zn
Poly_Element
Zn_Element
QQ
ifactor
Base_Structure
QQ_Element
primes
cdi
ZZ
Base_Element


pfactor.py (C:\Users\Sebastia\Anaconda3\Lib\site-packages\PyECC\pfactor.py) - Interactive Editor for Python
File Edit View Settings Shell Run Tools Help
Shells
Python
>>> ri(17)
340121641302438 68
>>>
Fq.py pfactor.py cdi.py pinors.py inner-product-signs.py
pfactor: factorizing polynomials
from PyECC.Fq import *
from PyECC.Power_Series import *
from PyECC.pfactorFunctions import *
'''
Theorem: Let K be a finite fiel of cardinal q, f in K[X], and n = degree(f).
Then f is irreducible iff
1) f divides X**(q**n), and
2) gcd(X**(q**(n//p)), f) = 1 for all prime divisors p of n.
See vzGathen-Gerhard-2003, pp 396-397.
'''
def polynomial_ring(K,symbol = 'X', name = ''):
P = Poly(K,symbol,name)
x = P.variable();
return [P,x]
def create_vector(elem,size='',direction=''):
'''creates a Vector_Element
1 parameter:

jupyter CC-4-1 Last Checkpoint: 2 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Python [default]

Markdown | CellToolbar

Let $\alpha \in \mathbb{F}_8$ and assume that $\alpha^3 = \alpha + 1$. Consider the matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \end{pmatrix}$$

and let C be the alternant binary code associated to H. Let us see that $C \equiv [7, 3, 4]$, so that $d = 4 > 3 = r + 1$.

First the minimum distance d of C is ≥ 4, as any three columns of H are linearly independent over $\mathbb{F}_2$. On the other hand, the first three columns and the column of α^5 are linearly dependent, for $\alpha^5 = \alpha^2 + \alpha + 1$, and so $d = 4$. Finally the dimension of C is 3, because it has a control matrix of rank 4 over $\mathbb{F}_2$, as the the is shown by next CC script.

In [5]:

```
## Computing the dimension using the blow and prune functions
from PyECC.CC import *

n = 7; r = 2
K = Zn(2);

[F,a] = extension(K,[1,0,1,1],'a','F')

H = create_matrix(F,[[1,1,1,1,1,1,1],[1,a,a**2,a**3,a**4,a**5,a**6]])
show(blow(H,K))
show(prune(blow(H,K)))
```

```
[[0      0      0      0      0      0      0]
 [0      0      0      0      0      0      0]
 [1      1      1      1      1      1      1]
 [0      0      1      0      1      1      1]
 [0      1      0      1      1      1      0]
 [1      0      0      1      0      1      1]] :: Matrix[Z2]
```

```
A = Zn(17)
a = 3>>A
--> 3 :: Z17

order(a)
--> 16

# Powers of a, seen as integers
[lift(a**j) for j in range(17)]
--> [1, 3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1]

# Order of 3 mod 17
order(3,17)
--> 16

# Powers of 3 mod 17
[3**j % 17 for j in range(17)]
--> [1, 3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1]
```

To create the polynomial ring $P = A[X]$:

```
[P,X] = polynomial_ring(A)
```

To get an irreducible monic polynomial $f \in P$ of degree t:

```
f = get_irreducible_polynomial(P,t)
```

Gauss formula for the number $I_q(t)$ of monic irreducible polynomials of degree t over F_q:

$$I_q(t) = \frac{1}{t}\sum_{d|t} \mu(t/d)q^d = \frac{q^t}{t} + \cdots$$

It follows that the probability of getting an irreducible polynomial out of all monic polynomial of degree t is:

$$I_q(t)/q^t = \frac{1}{t} + \cdots .$$

```
# Creating Z2
K = Zn(2)
# Creating F = F8 as K[X]/(f=X^3+X+1), a = X mod f
[F,a] = extension(K,[1,0,1,1],'a','F')

# In general (A ring)
[B,x] = extension(A,[1,a1,...,ar],'x','B')
# creates the ring
```

$$B = A[X]/(f = X^r + a_1X^{r-1} + \cdots + a_{r-1}X + a_r),$$

If $f = X^r + a_1X^{r-1} + \cdots + a_{r-1}X + a_r \in A[X] = P$, we also can use the following syntax:

```
[P,X] = polynomial_ring(A,'X')
f = X**r + a_1*X**(r-1) + ··· + a_{r-1}*X + a_r
[B,x] = extension(A,f,'x','B')
```

```
# Creation of a length n vector
# with coefficients in the ring A
x = vector(A,n)

# creation of a matrix in A(k,n), A a ring
M = matrix(A,k,n)

# Assignment of values
x[j] = a
M[i,j] = a
```

The function `scramble_matrix(A,k)` creates a $S \in A(k)$ with $\det(A) = 1$ and which is random under these conditions.

Note that the function `rd(A)` returns an element of A selected at random.

```
def scramble_matrix(A,k):
    U = matrix(A,k,k)
    L = matrix(A,k,k)
    for i in range(k):
        U[i,i] = L[i,i] = 1
        for j in range(i+1,k):
            U[i,j] = rd(A)
            L[j,i] = rd(A)
    return L*U
```

The function `permutation_matrix(n)` creates a random permutation matrix of order n.

```
def permutation_matrix(n):
    N = list(range(n))
    p = rd_choice(N,n)
    P = matrix(ZZ(),n,n)
    for j in range(n):
        P[j,p[j]] = 1
    return P
```

- $F = F_q$, $q = 2$ (many constructions work also for $q > 2$).
- $\bar{F} = F_{q^m}$, m a positive integer. If $\beta \in \bar{F}$, $[\beta]$ will denote the colum vector of its components with respect to a basis of $\bar{F}/F$.
- $\boldsymbol{\alpha} = \alpha_1, \ldots, \alpha_n \in \bar{F}$ distinct elements, so that $n \leq q^m$.
- $p \in \bar{F}[X]$ a polynomial of degree $r > 0$ such that $p(\alpha_j) \neq 0$ $(j = 1, \ldots, n)$.
- Set $h_j = 1/p(\alpha_j)$ $(j = 1, \ldots, n)$ and

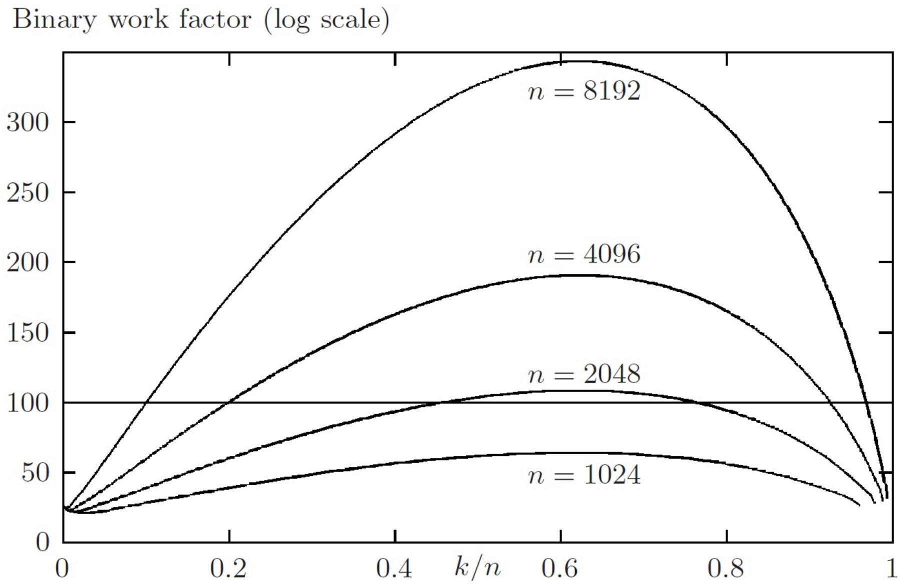
$$\bar{H} = \begin{pmatrix} h_1 & \cdots & h_n \\ h_1\alpha_1 & \cdots & h_n\alpha_n \\ \vdots & & \vdots \\ h_1\alpha_1^{r-1} & \cdots & h_n\alpha_n^{r-1} \end{pmatrix} \in \bar{F}(r, n).$$

- Let $H \in F(r', n)$ be the result of replacing each entry β of $\bar{H}$ by $[\beta]$ (this yields a matrix $[H] \in F(mr, n)$), followed by deleting from $[H]$ any row that is in the span of the previous ones. Note that $r' \leq mr$. It also holds that $r \leq r'$, as the $\langle H \rangle_{\bar{F}} = \langle \bar{H} \rangle_{\bar{F}}$.
- Let $\Gamma = \Gamma(p, \alpha) = \{\boldsymbol{x} \in F^n : \boldsymbol{x} H^T = 0\}$. It is a code of type $[n, k = n - r']$. This code is called the *classical Goppa code* associated to p and $\boldsymbol{\alpha}$.
- We have $n - mr \leq k \leq n - r$.
- Fact: If $G \in F(k, n)$ is a generating matrix of Γ, there is G-decoder that corrects $r/2$ errors in general, and r errors in the binary case provided p has no multiple roots in $\bar{F}$. See, for example, [2, P.4.7]

- Let $\boldsymbol{\alpha}$ be the set of elements of $\bar{F}$. Hence $n = 2^m$.
- Let $p \in \bar{F}[X]$ be a monic irreducible of degree $t > 1$. Then p has no roots in $\bar{F}$ and so a generating matrix G of $\Gamma(p, \boldsymbol{\alpha})$ has a decoder g that corrects t errors.

This ends the theoretical construction of a McECS with the following parameters:

- $n = 2^m$, where m is any posivive integer, and p is monic irreducible of degree t.
- $\bar{H} \in \bar{F}(t, n)$ and $G \in F(k, n)$, where $k = n - \mathrm{rank}(H)$ $(n - tm \leq k \leq n - t)$.
- Original example: $m = 10$, $n = 1024$, $t = 50$, $k = 524$ (in this case $k = n - tm$, the minimum possible given m and t.



Horizontal axis $R = k/n$, WF curves for $n = 2^j$, $j = 10, \ldots, 13$. N

```
## Computing the dimension using the blow and prune functions
from CC import *

n = 7; r = 2
K = Zn(2);

[F,a] = extension(K,[1,0,1,1],'a','F')

H = create_matrix(F,[[1,1,1,1,1,1,1],[1,a,a**2,a**3,a**4,a**5,a**6]])
show(blow(H,K))
show(prune(blow(H,K)))
```

```
[[0      0      0      0      0      0      0]
 [0      0      0      0      0      0      0]
 [1      1      1      1      1      1      1]
 [0      0      1      0      1      1      1]
 [0      1      0      1      1      1      0]
 [1      0      0      1      0      1      1]] :: Matrix[Z2]

[[1      1      1      1      1      1      1]
 [0      0      1      0      1      1      1]
 [0      1      0      1      1      1      0]
 [1      0      0      1      0      1      1]] :: Matrix[Z2]
```

```
F5 = Zn(5)
# Creation of F25, with generator x
[F25,x] = extension(F5,[1,0,-2],'x','F25')
# Creation of the polynomial ring F25[T]
[A,T] = polynomial_ring(F25,'T')
g = T**6 + T**3 + T + 1
a = Set(F25)[1:] # The non-zero elements of F25
a = [t for t in a if evaluate(g,t)!=0]
C = Goppa(g,a)
# generate a random error pattern of weight 3
e = rd_error_vector(Z5,n,3)
>> e = [0,1,0,0,0,3,0,4,0,0,0,0,0,0,0,0,0,0,0]
# Use the PGZ decoder for C
PGZ(e,C)
>>PGZ: Error positions [1,5,7], error values [1,3,4]
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] :: Vector[Z5]
```

- Code low level routines in cyton.
- Extend the package to include convolution codes.
- Second edition of Block Error-Correcting codes, based on PyECC
- Include PyECCin Sage?

More details: [5]

References (1)

[1] R. J. McEliece, "A public-key cryptosystem based on algebraic coding theory," 1978.

Jet Propulsion Laboratory, DSN Progress Report 42-44,
http://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF.

[2] S. Xambó-Descamps, *Block error-correcting codes: a computational primer*.

Univesitext, Springer, 2003.

[3] D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-Quantum Cryptography* (J. Buchanan and J. Ding, eds.), Lecture Notes Computer Science, pp. 31–46, 2008.

Proceedings of the Second international workshop PQCrypto 2008, Cincinnati, OH, USA, October 17-19.
https://cr.yp.to/codes/mceliece-20080807.pdf.

References (2)

[4] Post-Quantum Cryptography 2018.

First PQC Standardization Conference organized by the NIST Computer Security Resource Center.

https://csrc.nist.gov/Projects/Post-Quantum-Cryptography.

[5] N. Sayols and S. Xambó-Descamps, "Computer Algebra tales on Goppa Codes and McEliece Cryptography," 2018.

Thanks, ...

Why are we doing that ...

P

Note that we cannot avoid higher cardinals because (high degree) extensions of the base field will play a crucial role.

P

- Purely Python.
- Hirarchy of classes driven by several function definition files.
- The function files are grouped in two components: Low level Modular arithmetic utilities (in red) and high level interface utilities (in blue).
- The classes are grouped in two subsets: Element classes (in green) and algebraic structures (in orange).
- The arrows correspond to interdependencies.
- For details, see PyECC

P

P