Code-based digital signatures exploiting sparse vectors

Marco Baldi

Università Politecnica delle Marche
Ancona, Italy
m.baldi@univpm.it

CBC 2013
Rocquencourt, France
June 11, 2013

Sparsity has several advantages...

Large families of codes with:

- very good (and equivalent) error correction ability
- random-based design
- low-complexity, capacity-achieving iterative decoding
- small storage space due to sparse matrices (at least in principle)
- chance to obtain random-based structured (QC) codes
- possibly some predictable properties (minimum distance and multiplicity)

...but also some drawbacks

- The public matrix cannot be sparse, otherwise the secret matrix can be recovered through correlation-based techniques
- If the public matrix is dense, but the code is permutation equivalent to the private one, dual code attacks can still exploit sparsity
- In general, sparsity of the private code shall be disguised and permutation equivalence with the public code avoided

- C. Monico, J. Rosenthal, and A. Shokrollahi, *Using low density parity check codes in the McEliece cryptosystem,* in Proc. IEEE ISIT 2000, Sorrento, Italy, Jun. 2000, p. 215.
- M. Baldi, F. Chiaraluce, *Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC codes,* Proc. IEEE ISIT 2007, Nice, France, Jun. 2007, pp. 2591–2595.
- A. Otmani, J.P. Tillich, L. Dallot, *Cryptanalysis of two McEliece cryptosystems based on quasi-cyclic codes,* Proc. SCC 2008, Beijing, China, Apr. 2008.
- M. Baldi, M. Bodrato, F. Chiaraluce, *A New Analysis of the McEliece Cryptosystem based on QC-LDPC Codes,* Proc. SCN 2008, Amalfi, Italy, vol. 5229 of LNCS, pp. 246–262.

Disguising the code sparsity

A possible solution to disguise the code sparsity is:

- multiply its sparse parity-check matrix $\boldsymbol{H}$ by a denser matrix $\boldsymbol{Q}$: $\boldsymbol{H}' = \boldsymbol{H}\boldsymbol{Q}$
- obtain a dense generator matrix $\boldsymbol{G}'$ corresponding to $\boldsymbol{H}'$
- use $\boldsymbol{G}'$ as the public key

This way:

- the public matrix is dense
- the public code is no longer permutation equivalent to the private code

This prevents from attacking either the public matrix or its dual code

- M. Baldi, M. Bianchi, F. Chiaraluce, *Security and complexity of the McEliece cryptosystem based on QC-LDPC codes*, IET Information Security, in press, `http://arxiv.org/abs/1109.5827`.

Disguising the code sparsity (2)

However, this way the number of intentional errors is increased.
E.g. in the McEliece cryptosystem based on LDPC codes:

- Private key: $\{\boldsymbol{S}, \boldsymbol{H}, \boldsymbol{Q}\}$
- Public key: $\boldsymbol{G}' = \boldsymbol{S}^{-1}\boldsymbol{G}\boldsymbol{Q}^{-1}$
- Encryption: $\boldsymbol{x} = \boldsymbol{u}\boldsymbol{G}' + \boldsymbol{e}$
- First decryption step: $\boldsymbol{x}' = \boldsymbol{x}\boldsymbol{Q} = \boldsymbol{u}\boldsymbol{S}^{-1}\boldsymbol{G} + \boldsymbol{e}\boldsymbol{Q}$

Bob must hence correct the error vector $\boldsymbol{e}\boldsymbol{Q}$
If m is the row and column weight of $\boldsymbol{Q}$, the errors become $\leq tm$ (hence, their number increases up to m times)

Another way to disguise sparsity

$$\boldsymbol{Q} = \mathbf{1}_{n\times n} + \boldsymbol{P}$$

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

$$\boldsymbol{e}_1\boldsymbol{Q} = \boldsymbol{e}_4 \longrightarrow \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\boldsymbol{e}_2\boldsymbol{Q} = \boldsymbol{e}_2' \longrightarrow \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Another way to disguise sparsity (2)

- With $\boldsymbol{Q} = \mathbf{1}_{n \times n} + \boldsymbol{P}$, the error vectors such that $\boldsymbol{e} \cdot \mathbf{1}_{n \times n} = \mathbf{0}$ are only permuted by $\boldsymbol{Q}$
- $\mathbf{1}_{n \times n}$ can be seen as the (redundant) parity-check matrix of the $(n, n-1)$ single parity-check code
- Hence, selecting the error vectors with even parity produces a set of error vectors over which $\boldsymbol{Q}$ becomes a permutation
- This position can be generalized to any other $(n, n-1)$ binary code

Another way to disguise sparsity (3)

$$\boldsymbol{Q} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix} \cdot \begin{bmatrix} b_1 & b_2 & b_3 & \ldots & b_n \end{bmatrix} + \boldsymbol{P} = \boldsymbol{a}^T \cdot \boldsymbol{b} + \boldsymbol{P}$$

- $\boldsymbol{a}$ is the parity-check matrix of a $(n, n-1)$ linear block code C_a
- If $\boldsymbol{e} \cdot \boldsymbol{a}^T = 0$, $\boldsymbol{e} \cdot \boldsymbol{Q} = \boldsymbol{e} \cdot \boldsymbol{P}$
- Hence, the error vectors $\in C_a$ are only permuted by $\boldsymbol{Q}$

Another way to disguise sparsity (4)

General setting

- $\boldsymbol{a}$ and $\boldsymbol{b}$ are two non-singular binary (or non-binary) $z \times n$ matrices
- $\boldsymbol{Q} = \boldsymbol{a}^T \cdot \boldsymbol{b} + \sum_{i=1}^{m} \boldsymbol{P}_i$
- $\boldsymbol{a}$ defines an $(n, n-z)$ linear block code C_a
- If $\boldsymbol{e} \in C_a$, $\boldsymbol{e} \cdot \boldsymbol{Q}$ has weight $\leq m$
- If $\boldsymbol{a}$ can be made public (only $\boldsymbol{b}$ kept secret), $\boldsymbol{e} \in C_a$ is found by encoding through $\boldsymbol{a}$
- Otherwise $\boldsymbol{e} \in C_a$ can be found at random (q^z attempts on average, for a q-ary code)
- Note: $\boldsymbol{Q}$ must be non-singular, but this is easy to obtain

Example of application (McEliece system)

- $\boldsymbol{Q} = \boldsymbol{a}^T \cdot \boldsymbol{b} + \sum_{i=1}^{m} \boldsymbol{P}_i$ can replace $\boldsymbol{P}$ in McEliece
- The public and private codes are no longer permutation equivalent
- This could allow to restore the use of GRS codes
- If $m = 1$, $\boldsymbol{Q}$ becomes a permutation $\forall \boldsymbol{e} \in C_a$
- $\boldsymbol{a}$ must be kept secret to avoid subcode attacks
- If $m = 1$ and $z = 1$, there are distinguishers able to tell the public matrices from random ones
- This can be avoided by setting $m > 1$ or $z > 1$

- M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, D. Schipani, *A variant of the McEliece cryptosystem with increased public key security,* Proc. WCC 2011, Paris, France, 11-15 Apr. 2011.
- A. Couvreur, P. Gaborit, V. Gauthier, A. Otmani, J.-P. Tillich, *Distinguisher-Based Attacks on Public-Key Cryptosystems Using Reed-Solomon Codes,* Proc. WCC 2013, Bergen, Norway, Apr. 2013.
- M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, D. Schipani, *Enhanced public key security for the McEliece cryptosystem,* submitted, 2011, http://arxiv.org/abs/1108.2462.

Application to code-based signatures

Good news

First proposal of sparse code-based signature scheme

Application to code-based signatures

Good news

First proposal of sparse code-based signature scheme

Bad news

The security assessment is still at the beginning

Application to code-based signatures

Good news

First proposal of sparse code-based signature scheme

Bad news

The security assessment is still at the beginning

But...

...finding better attack procedures does not necessarily mean to abandon a system (the LDPC/MDPC story begun this way)

Preliminaries on code-based signatures

- Quantum computers will also endanger many widespread signature schemes (like DSA and RSA signatures)
- Only a few replacements are available up to now (like hash-based signatures)
- Code-based digital signatures are post-quantum
- But finding efficient code-based solutions is still a challenge
- Two main proposals: Kabatianskii-Krouk-Smeets (KKS) and Courtois-Finiasz-Sendrier (CFS) schemes

- N. Courtois, M. Finiasz and N. Sendrier, *How to achieve a McEliece-based digital signature scheme,* Proc. ASIACRYPT 2001, Vol. 2248 of LNCS, pp. 157-174, Springer, Heidelberg, 2001.
- G. Kabatianskii, E. Krouk and B. Smeets, *A digital signature scheme based on random error correcting codes,* Proc. 6th IMA Int. Conf. on Cryptography and Coding, pp. 161-167, London, UK, 1997.

Preliminaries on code-based signatures (2)

- In KKS, two different size codes are used to create the trapdoor, one selecting the subset support of the other
- An important weakness of this system was recently pointed out
- There are still some parameter choices for which it is secure
- But KKS is only suitable for generating few signatures
- The CFS scheme instead implements a hash-and-sign scheme exploits only one secret code
- The most dangerous attacks are generalized birthday attacks

▶ A. Otmani and J. P. Tillich, *An efficient attack on all concrete KKS proposals*, Proc. PQCrypto 2011, Nov 29–Dec 2, Taipei, Taiwan.

CFS scheme

- Private key: $\{\boldsymbol{S}, \boldsymbol{H}\}$, with
  - $\boldsymbol{H}$: parity-check matrix of a secret t-error correcting Goppa code $C(n, k)$
  - $\boldsymbol{S}$: $n \times n$ non-singular random matrix
- $\mathcal{H}$: public hash algorithm with r-bit digest
- $\mathcal{F}$: function able to transform (in a reasonable time) any hash value computed through $\mathcal{H}$ into a correctable syndrome through C

CFS scheme (2)

Given the file to be signed D:

1. The signer computes $\boldsymbol{h} = \mathcal{H}(D)$
2. The signer computes $\boldsymbol{s} = \mathcal{F}(\boldsymbol{h})$ such that $\boldsymbol{s}' = \boldsymbol{S}^{-1} \cdot \boldsymbol{s}$ is a correctable syndrome (the parameters to be used in $\mathcal{F}$ are made public)
3. Through syndrome decoding, the signer finds $\boldsymbol{e}$ with weight $\leq t$ such that $\boldsymbol{s}' = \boldsymbol{H} \cdot \boldsymbol{e}$
4. The signature of D is $\boldsymbol{e}$
5. The verifier receives the signed $\widehat{D}$ and computes $\boldsymbol{H}' \cdot \boldsymbol{e} = \boldsymbol{S} \cdot \boldsymbol{H} \cdot \boldsymbol{e} = \boldsymbol{S} \cdot \boldsymbol{s}' = \boldsymbol{s}$
6. He also computes $\widehat{\boldsymbol{h}} = \mathcal{H}(\widehat{D})$ and $\widehat{\boldsymbol{s}} = \mathcal{F}(\widehat{\boldsymbol{h}})$
7. If $\widehat{\boldsymbol{s}} = \boldsymbol{s}$, $\widehat{D}$ is accepted, otherwise discarded

CFS scheme (3)

Main limits of the CFS scheme:

- It is very hard to find a function $\mathcal{F}$ that quickly transforms an arbitrary hash vector into a correctable syndrome
- Two possible solutions:
  1. appending a counter to the message
  2. performing complete decoding
- Both of them require a very special choice of the code parameters
- Codes with very high rate and very small error correction capability are needed
- This has exposed the cryptosystem to attacks based on the generalized birthday algorithm
- In addition, the key size and decoding complexity can be very large

Sparse code-based signatures

Variant of the CFS scheme in which:

1. Only a subset of sparse syndromes is considered
2. Goppa codes are replaced with low-density generator-matrix (LDGM) codes,

Main advantages:

1. Significant reductions in the public key size are achieved
2. Classical attacks against the CFS scheme are inapplicable
3. Decoding is replaced by a straightforward vector manipulation

- M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, D. Schipani, *Using LDGM Codes and Sparse Syndromes to Achieve Digital Signatures*, Proc. PQCrypto 2013, Limoges, France, June 2013.

A simple observation

- Given an $r \times 1$ syndrome vector $\boldsymbol{s}$
- Given a code with parity-check matrix $\boldsymbol{H} = [\boldsymbol{X}|\boldsymbol{I}_r]$, with $\boldsymbol{I}_r$ the $r \times r$ identity matrix
- The error vector $\boldsymbol{e} = \left[\mathbf{0}_{1\times k}|\boldsymbol{s}^T\right]$ verifies $\boldsymbol{H}\boldsymbol{e}^T = \boldsymbol{s}$
- If $\boldsymbol{s}$ has weight $\leq t$, $\boldsymbol{e}$ is unique
- Otherwise, $\boldsymbol{e}$ is not unique, but the map $\boldsymbol{s} \leftrightarrow \boldsymbol{e}$ remains unique

Can this be exploited to facilitate code-based signing?

There are several issues:

1. The map $\boldsymbol{s} \leftrightarrow \boldsymbol{e}$ is trivial
2. The public syndrome should undergo (at least) a secret permutation before obtaining $\boldsymbol{e}$
3. Also $\boldsymbol{e}$ should be somehow disguised before being made public
4. Sparsity could be exploited to distinguish $\boldsymbol{e}$ from other vectors in the same coset...
5. ...but it shall not endanger the public key
6. The map $\boldsymbol{s} \leftrightarrow \boldsymbol{e}$ is also linear

System description

Key generation:

- Private key: $\{\boldsymbol{Q}, \boldsymbol{H}, \boldsymbol{S}\}$, with
  - $\boldsymbol{H}$: $r \times n$ parity-check matrix of a secret code $C(n,k)$
  - $\boldsymbol{Q} = \boldsymbol{R} + \boldsymbol{T}$
  - $\boldsymbol{R} = \boldsymbol{a}^T \cdot \boldsymbol{b}$, having rank $z \ll n$
  - $\boldsymbol{T}$: sparse random matrix with row and column weight m_T, such that $\boldsymbol{Q}$ is full rank
  - $\boldsymbol{S}$: sparse non-singular $n \times n$ matrix with row and column weight $m_S \ll n$
- Public key: $\boldsymbol{H}' = \boldsymbol{Q}^{-1} \cdot \boldsymbol{H} \cdot \boldsymbol{S}^{-1}$

System description (2)

Public functions:

1. $\mathcal{H}$: hash function
2. $\mathcal{F}_\Theta$: function that converts the output of $\mathcal{H}$ into a sparse r-bit vector $\boldsymbol{s}$ of weight $w \ll r$

- The output of $\mathcal{F}_\Theta$ depends on the parameter Θ
- Θ is associated to the message and made public as well

For example:

- Given the x-bit message digest $\boldsymbol{h} = \mathcal{H}(M)$
- The function $\mathcal{F}$ appends it with the y-bit value $\boldsymbol{l}$ of a counter
- $[\boldsymbol{h}|\boldsymbol{l}]$ is then mapped uniquely into one of the $\binom{r}{w}$ r-bit vectors of weight w

Signature generation

- Given the document M
- The signer computes $\boldsymbol{h} = \mathcal{H}(M)$
- The signer finds Θ_M such that $\boldsymbol{s} = \mathcal{F}_{\Theta_M}(\boldsymbol{h})$ verifies $\boldsymbol{b} \cdot \boldsymbol{s} = \mathbf{0}_{z \times 1}$
  - For the counter-based implementation, $\Theta_M = \boldsymbol{l}$
  - This step requires 2^z attempts, on average
- The signer computes the private syndrome $\boldsymbol{s}' = \boldsymbol{Q} \cdot \boldsymbol{s}$ of weight $\leq m_T w$
- The signer computes the private error vector $\boldsymbol{e} = [\mathbf{0}_{1 \times k} | \boldsymbol{s}'^T]$
- The signer selects a random codeword $\boldsymbol{c} \in C$ with small Hamming weight (w_c)
- The signer computes the public signature of M as $\boldsymbol{e}' = (\boldsymbol{e} + \boldsymbol{c}) \cdot \boldsymbol{S}^T$

Signature generation issues

1. If the choice of the codeword $\boldsymbol{c}$ is completely random and independent of the document to be signed, the signature changes each time a document is signed, and this exposes the system to attacks exploiting many signatures of the same document
   - Hence, $\boldsymbol{c}$ must be chosen as a deterministic function of M
   - For example, $\boldsymbol{s}$ or $[\boldsymbol{h}|\boldsymbol{l}]$ can be used as a seed to select $\boldsymbol{c}$

2. If $\boldsymbol{c} = \mathbf{0}_{1\times n}, \forall M$, the signing map becomes linear
   - This justifies the presence of the random codeword $\boldsymbol{c}$
   - $\boldsymbol{c}$ must have weight $0 < w_c \ll n$

Linear map risk

- If $\boldsymbol{c} = \mathbf{0}_{1 \times n}, \forall M$, $\boldsymbol{e}' = W(\boldsymbol{s})$, where W is a linear bijective map
- In fact, $W(\boldsymbol{s}_1 + \boldsymbol{s}_2) = W(\boldsymbol{s}_1) + W(\boldsymbol{s}_2)$
- After intercepting some documents and signatures, and attacked could forge a valid signature for the public syndrome $\boldsymbol{s}_x$
- He expresses $\boldsymbol{s}_x$ as a linear combination of intercepted public syndromes $\boldsymbol{s}_x = \boldsymbol{s}_{i_1} + \boldsymbol{s}_{i_2} + \ldots \boldsymbol{s}_{i_N}$
- And forges the signature as $\boldsymbol{e}'_x = \boldsymbol{e}'_{i_1} + \boldsymbol{e}'_{i_2} + \ldots \boldsymbol{e}'_{i_N}$
- With the random codeword $\boldsymbol{c}$, W becomes an affine map depending on $\boldsymbol{c}$: $W_c(\boldsymbol{s})$
- Given $\boldsymbol{e}'_1 = W_{\boldsymbol{c}_1}(\boldsymbol{s}_1)$ and $\boldsymbol{e}'_2 = W_{\boldsymbol{c}_2}(\boldsymbol{s}_2)$
- $\boldsymbol{e}'_f = \boldsymbol{e}'_1 + \boldsymbol{e}'_2 = W_{\boldsymbol{c}_1}(\boldsymbol{s}_1) + W_{\boldsymbol{c}_2}(\boldsymbol{s}_2) = W_{\boldsymbol{c}_1+\boldsymbol{c}_2}(\boldsymbol{s}_1 + \boldsymbol{s}_2)$
- $\boldsymbol{c}_1 + \boldsymbol{c}_2$ is a codeword $\in C$, but of weight $> w_c$

Signature verification

- The verifier receives the message M, its signature $\boldsymbol{e}'$ and the associated parameter Θ_M
- He first checks that the weight of $\boldsymbol{e}'$ is $\leq (m_T w + w_c) m_S$, otherwise the signature is discarded
- He then computes $\widehat{\boldsymbol{s}} = \mathcal{F}_{\Theta_M}(\mathcal{H}(M))$ and checks that $\widehat{\boldsymbol{s}}$ has weight w, otherwise the signature is discarded
- He then computes $\boldsymbol{H}' \cdot \boldsymbol{e}'^T = \boldsymbol{Q}^{-1} \cdot \boldsymbol{H} \cdot \boldsymbol{S}^{-1} \cdot \boldsymbol{S} \cdot (\boldsymbol{e}^T + \boldsymbol{c}^T) = \boldsymbol{Q}^{-1} \cdot \boldsymbol{H} \cdot (\boldsymbol{e}^T + \boldsymbol{c}^T) = \boldsymbol{Q}^{-1} \cdot \boldsymbol{H} \cdot \boldsymbol{e}^T = \boldsymbol{Q}^{-1} \cdot \boldsymbol{s}' = \boldsymbol{s}$
- If $\boldsymbol{s} = \widehat{\boldsymbol{s}}$, the signature is accepted, otherwise it is discarded

Which codes to use?

In this scheme, we need secret codes characterized by:

- Chance to design large random-based families of codes
- Easiness of finding low weight codewords
- Possibility to design structured codes (e.g. QC)

$\vdots$

Which codes to use?

In this scheme, we need secret codes characterized by:

- Chance to design large random-based families of codes
- Easiness of finding low weight codewords
- Possibility to design structured codes (e.g. QC)

⋮

Low Density Generator Matrix (LDGM) codes!

LDGM codes

- Codes having sparse generator matrices
- Achieve very good performance in concatenated schemes
- May have low-density parity-check (LDPC) matrices (and also free of short cycles)
- Can be designed in QC form (QC-LDGM)

- J. F. Cheng and R. J. McEliece, *Some high-rate near capacity codecs for the Gaussian channel,* in Proc. 34th Allerton Conference on Communications, Control and Computing, Allerton, IL, Oct. 1996.
- J. Garcia-Frias and W. Zhong, *Approaching Shannon performance by iterative decoding of linear codes with low-density generator matrix,* IEEE Commun. Lett., Vol. 7, No. 6, pp. 266–268, Jun. 2003.
- M. González-López, F. J. Vázquez-Araújo, L. Castedo, and J. Garcia-Frias, *Serially-concatenated low-density generator matrix (SCLDGM) codes for transmission over AWGN and Rayleigh fading channels,* IEEE Trans. Wireless Commun., Vol. 6, No. 8, pp. 2753–2758, Aug. 2007.
- M. Baldi, F. Bambozzi, F. Chiaraluce, *On a Family of Circulant Matrices for Quasi-Cyclic Low-Density Generator Matrix Codes,* IEEE Trans. on Information Theory, Vol. 57, No. 9, pp. 6052–6067, 2011.

Random-based design of LDGM codes

First approach (systematic $\boldsymbol{G}$ and LDPC code too):

- Randomly select a sparse $k \times r$ matrix $\boldsymbol{D}$ with row-weight $w_g - 1 \ll n$
- Set $\boldsymbol{G} = [\boldsymbol{I}_k|\boldsymbol{D}]$, with $\boldsymbol{I}_k$ the $k \times k$ identity matrix

Second approach (non-systematic $\boldsymbol{G}$ and non-LDPC code):

- Randomly select k linearly independent vectors with length n and Hamming weight $w_g \ll n$
- Use them to form the rows of $\boldsymbol{G}$

The latter requires to check the linear independence of the rows of $\boldsymbol{G}$, but it increases the degrees of freedom for random-based designs

Low weight codewords in LDGM codes

- Due to sparsity, by summing two or more rows of $\boldsymbol{G}$ we get a codeword with weight $\geq w_g$
- Hence (except for trivial cases) the LDGM code has minimum distance w_g
- We need random codewords with weight $\approx w_c \ll n$
- We suppose (w.l.o.g.) that $w_g | w_c$
- By summing $\frac{w_c}{w_g}$ rows of $\boldsymbol{G}$, chosen at random, we get a codeword with Hamming weight about w_c
- Some row of $\boldsymbol{G}$ can be added or replaced to adjust the resulting weight
- The number of codewords with weight $\approx w_c$ is about

$$A_{w_c} \approx \binom{k}{\frac{w_c}{w_g}}$$

QC-LDGM codes

- Using QC-LDGM codes allows to reduce the key size
- General form for a QC-LDGM:

$$\boldsymbol{G}_{QC} = \begin{bmatrix} \boldsymbol{C}_{0,0} & \boldsymbol{C}_{0,1} & \boldsymbol{C}_{0,2} & \cdots & \boldsymbol{C}_{0,n_0-1} \\ \boldsymbol{C}_{1,0} & \boldsymbol{C}_{1,1} & \boldsymbol{C}_{1,2} & \cdots & \boldsymbol{C}_{1,n_0-1} \\ \boldsymbol{C}_{2,0} & \boldsymbol{C}_{2,1} & \boldsymbol{C}_{2,2} & \cdots & \boldsymbol{C}_{2,n_0-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{C}_{k_0-1,0} & \boldsymbol{C}_{k_0-1,1} & \boldsymbol{C}_{k_0-1,2} & \cdots & \boldsymbol{C}_{k_0-1,n_0-1} \end{bmatrix}$$

- $\boldsymbol{C}_{i,j}$ is a $p \times p$ sparse circulant matrix or null matrix
- The code has $n = n_0 p$, $k = (n_0 - r_0)p = k_0 p$ and $r = r_0 p$
- Storing the parity-check matrix $\boldsymbol{H}_{QC}$ requires $r_0 n_0 p = rn/p$ bits

Using QC-LDGM codes in the signature scheme

- We must preserve the QC structure of $\boldsymbol{H}_{QC}$ also in the public key $\boldsymbol{H}'_{QC}$
- Hence, both $\boldsymbol{Q}$ and $\boldsymbol{S}$ must be in QC form as well
- $\boldsymbol{S}_{QC}$: random block of $n_0 \times n_0$ sparse or null circulant matrices with overall row and column weight m_S
- $\boldsymbol{R}_{QC} = \left(\boldsymbol{a}_{r_0}^T \cdot \boldsymbol{b}_{r_0}\right) \otimes \mathbf{1}_{p \times p}$, with $\boldsymbol{a}_{r_0}$ and $\boldsymbol{b}_{r_0}$ two $z \times r_0$ binary matrices, $\mathbf{1}_{p \times p}$ the all-one $p \times p$ matrix and $\otimes$ the Kronecker product
- $\boldsymbol{T}_{QC}$: random block of $n_0 \times n_0$ sparse or null circulant matrices with overall row and column weight m_T
- $\boldsymbol{Q}_{QC} = \boldsymbol{R}_{QC} + \boldsymbol{T}_{QC}$
- The condition becomes $(\boldsymbol{b}_{r_0} \otimes \mathbf{1}_{1 \times p}) \cdot s = 0_{z \times 1}$

Foreword on attacks

- The security assessment of this scheme is still at the beginning
- Some possible vulnerabilities have already been devised
- We estimated the security level of the system only based on such vulnerabilities
- But work is in progress!

Vulnerabilities and density of the signature

- The signature $\boldsymbol{e}'$ is an error vector corresponding to the public syndrome $\boldsymbol{s}$ through the public code parity-check matrix $\boldsymbol{H}'$
- If $\boldsymbol{e}'$ has a low weight, it is difficult to find, otherwise signatures can be forged
- If $\boldsymbol{e}'$ has a too low weight the supports of $\boldsymbol{e}$ and $\boldsymbol{c}$ could be almost disjoint, and the link between the support of $\boldsymbol{e}$ (i.e., $\boldsymbol{s}$) and that of $\boldsymbol{e}'$ could be discovered

Hence, the density of $\boldsymbol{e}'$ must be:

1. sufficiently low to avoid forgeries
2. sufficiently high to avoid support decompositions

Forgery attacks

- To forge signatures, an attacker could search for an $n \times r$ right-inverse matrix $\boldsymbol{H}'_r$ of $\boldsymbol{H}'$ (i.e., $\boldsymbol{H}' \cdot \boldsymbol{H}'_r = \boldsymbol{I}_r$)
- Then, $\boldsymbol{e}_f = (\boldsymbol{H}'_r \cdot \boldsymbol{s})^T$ is a forged signature
- It is easy to find a right-inverse matrix able to forge dense signatures: if $\boldsymbol{H}' \cdot \boldsymbol{H}'^T$ is invertible, $\boldsymbol{H}'_r = \boldsymbol{H}'^T \cdot \left(\boldsymbol{H}' \cdot \boldsymbol{H}'^T\right)^{-1}$ is a right-inverse matrix of $\boldsymbol{H}'$
- But $\left(\boldsymbol{H}' \cdot \boldsymbol{H}'^T\right)^{-1}$ is dense, hence $\boldsymbol{H}'_r$ is dense as well
- So $\boldsymbol{H}'_r$ only allows to forge dense signatures
- Since it uses signatures with weight $\leq (m_T w + w_c) m_S$, the system is robust against this kind of forged signatures

Forgery attacks (3)

- The right-inverse matrix is not unique, hence an attacker could look for a sparse one
- Given an $n \times n$ matrix $\boldsymbol{Z}$ such that $\boldsymbol{H}' \cdot \boldsymbol{Z} \cdot \boldsymbol{H}'^T$ is invertible, $\boldsymbol{H}''_r = \boldsymbol{Z} \cdot \boldsymbol{H}'^T \cdot \left(\boldsymbol{H}' \cdot \boldsymbol{Z} \cdot \boldsymbol{H}'^T\right)^{-1}$ is another valid right-inverse matrix of $\boldsymbol{H}'$
- If $\boldsymbol{H}'$ contains an invertible $r \times r$ square block, a right-inverse is also obtained as a null matrix with the inverse of such block in the same position
- However, a sparse right-inverse is too difficult to find (and may not even exist)
- We can assume that an attacker may succeed to forge signatures with weight about $r/2 < n/2$
- But we consider valid public signatures with weight $r/3$ or less

Forgery attacks (3)

- Alternatively, an attacker could try syndrome decoding of $\boldsymbol{s}$ through $\boldsymbol{H}'$, hoping to find a sparse vector $\boldsymbol{e}_f$
- He may have the advantage of searching for one out of many possible vectors
- Several algorithms can be used for this purpose
- Their complexity decreases when an attacker aims to solve only one out of many decoding instances

- C. Peters, "Information-set decoding for linear codes over F_q," in Post-Quantum Cryptography, Vol. 6061 of LNCS, Springer, 2010, pp. 81–94.
- D. J. Bernstein, T. Lange, and C. Peters, "Smaller decoding exponents: ball-collision decoding," in *CRYPTO 2011*, Vol. 6841 of LNCS, Springer, 2011, pp. 743–760.
- A. Becker, A. Joux, A. May, and A. Meurer, "Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding," in *EUROCRYPT 2012*, Vol. 6841 of LNCS, Springer, 2012.
- N. Sendrier, "Decoding one out of many". In B.-Y. Yang, editor, Post-Quantum Cryptography, Vol. 7071 of LNCS, Springer, 2011, pp. 51–67.

Support decomposition attacks

- An attacker could try to discover the relationships between the supports of $\boldsymbol{s}$ and $\boldsymbol{e}'$, i.e., to remove the effect of the random codeword
- He collects a sufficiently large number L of pairs $(\boldsymbol{s}, \boldsymbol{e}')$
- He intersects the supports (i.e., compute the bit-wise AND) of all the $\boldsymbol{s}$ vectors
- He obtains a vector $\boldsymbol{s}_L$ that may have a small weight $w_L \geq 1$
- If this succeeds, the attacker analyzes the vectors $\boldsymbol{e}'$, and selects the mw_L set bit positions that appear more frequently
- If these bit positions actually correspond to the w_L bits set in s_L, then the attacker has discovered the relationship between them

Support decomposition attacks (2)

- Alternatively, an attacker could exploit information set decoding to remove the effect of the random codeword
- Since $\boldsymbol{e}' = (\boldsymbol{e} + \boldsymbol{c}) \cdot \boldsymbol{S}^T = \boldsymbol{e}'' + \boldsymbol{c}''$, with $\boldsymbol{c}''$ such that $\boldsymbol{H}'\boldsymbol{c}''^T = \boldsymbol{0}$, $\boldsymbol{e}''$ can be considered as an error vector with weight $\leq m_T m_S w$ affecting the codeword $\boldsymbol{c}''$ of the public code
- The attacker considers a random subset of k coordinates of the public signature $\boldsymbol{e}'$ and assume that no errors occurred on these coordinates
- In this case, he can easily recover $\boldsymbol{e}''$ and, hence, remove the effect of the random codeword $\boldsymbol{c}$
- The probability that there are no errors in the chosen k coordinates is $\binom{n-m_T m_S w}{k} / \binom{n}{k}$
- Its inverse is a rough estimate of the work factor of this attack

Key recovery attacks

- The public code admits a generator matrix in the form $\boldsymbol{G}'_I = \boldsymbol{G} \cdot \boldsymbol{S}^T$, which is rather sparse
- So, the public code contains low weight codewords
- They coincide with the rows of $\boldsymbol{G}'_I$ and have weight $\approx w_g \cdot m_S$
- We can consider their multiplicity equal to k
- They can be searched by using again low-weight codeword searching algorithms
- After recovering $\boldsymbol{G}'_I$, it can be separated into $\boldsymbol{G}$ and $\boldsymbol{S}^T$ by exploiting their sparsity

Key recovery attacks (2)

- The matrix $\boldsymbol{b}$ is public
- If it was not public, an attacker could obtain the vector space it generates by observing $O(r)$ public syndromes $\boldsymbol{s}$ ($\boldsymbol{b} \cdot \boldsymbol{s} = \mathbf{0}_{z \times 1}$)
- Hence an attacker may know an $r \times r$ matrix $\boldsymbol{V}$ such that $\boldsymbol{R} \cdot \boldsymbol{V} = 0 \Rightarrow \boldsymbol{Q} \cdot \boldsymbol{V} = \boldsymbol{T} \cdot \boldsymbol{V}$
- He also knows that the public code admits any non-singular generator matrix in the form $\boldsymbol{G}'_X = \boldsymbol{X} \cdot \boldsymbol{G} \cdot \boldsymbol{S}^T$
- $\boldsymbol{G}'_I$ is the sparsest among them, and it can be attacked by searching for low weight codewords in the public code
- Knowing $\boldsymbol{V}$ is useless to reduce the complexity of attacking either $\boldsymbol{H}'$ or one of the possible $\boldsymbol{G}'_X$

Other attacks

- If the system admits up to N_s different signatures, it is sufficient to collect $\approx \sqrt{N_s}$ different signatures to mount a collision birthday attack
- Hence, the security level cannot exceed $\sqrt{N_s}$
- N_s can be increased by increasing w
- In fact, we do not actually need a private code of minimum distance greater than $2w$
- This is due to the special mapping between sparse syndromes and error vectors

Attacks against CFS

- The CFS scheme was successfully attacked by exploiting syndrome decoding based on the generalized birthday algorithm
- Since we can use different code parameters (and, in particular, lower code rates), we obtain huge work factors for the proposed system
- Taking into account the structured nature of the matrices can reduce the attack work factor on the order of 2^{10}
- It is very unlikely that this strategy can endanger the proposed signature scheme

- M. Finiasz and N. Sendrier, "Security bounds for the design of code-based cryptosystems," Proc. ASIACRYPT '09, Tokyo, Japan, Dec 6–10, 2009, pp. 88–105.
- L. Minder and A. Sinclair, "The Extended k-tree Algorithm," Journal of Cryptology, Vol. 25, No. 2, pp. 349–382, 2012.
- R. Niebuhr, P.-L. Cayrel and J. Buchmann, "Improving the efficiency of Generalized Birthday Attacks against certain structured cryptosystems," Proc. WCC 2011, Paris, France, Apr. 11-15, 2011.

System examples ($d = 2$ and $w_L = 2$)

SL (bits)	n	k	p	w	w_g	w_c	z	m_T	m_S	A_{w_c}	N_s	S_k (KiB)
80	9800	4900	50	18	20	160	2	1	9	$2^{82.76}$	$2^{166.10}$	117
120	24960	10000	80	23	25	325	2	1	14	$2^{140.19}$	$2^{242.51}$	570
160	46000	16000	100	29	31	465	2	1	20	$2^{169.23}$	$2^{326.49}$	1685

- M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, D. Schipani, *Using LDGM Codes and Sparse Syndromes to Achieve Digital Signatures*, Proc. PQCrypto 2013, Limoges, France, June 4-7, 2013.

Comments

- For 80-bit security, the original CFS system needs a Goppa code with $n = 2^{21}$ and $r = 210$, which gives a key size of 52.5 MiB
- By using the parallel CFS, the same security level is obtained with key sizes between 1.25 MiB and 20 MiB
- The proposed system requires a public key of only 117 KiB to achieve 80-bit security
- In addition, it exploits a straightforward decoding procedure for the secret code
- On the other hand, 2^z attempts are needed, on average, to find an $\boldsymbol{s}$ vector such that $\boldsymbol{b} \cdot \boldsymbol{s} = \mathbf{0}_{z \times 1}$
- But this check is very simple to perform, especially for very small values of z

► M. Finiasz, "Parallel-CFS strengthening the CFS McEliece-based signature scheme," Proc. PQCrypto 2010, Darmstadt, Germany, May 25-28, 2010, pp. 61–72.

ESCAPADE Research Project

Research project financed by the Italian Ministry of Education, University and Research (MIUR)

http://escapade.dii.univpm.it/