The challenge of using sparse and structured codes in code-based cryptography

Marco Baldi

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

Korea Cryptography Forum Annual Symposium

Seoul, Korea
November 15, 2019

McEliece cryptosystem

- Proposed by Robert McEliece in 1978.
- Irreducible Goppa codes were used in the original proposal.
- Secret irreducible Goppa code:
  - irreducible polynomial of degree t over $GF(2^m)$,
  - length (maximum): $n = 2^m$,
  - dimension: $k \geq n - t \cdot m$,
  - correction capability: t errors.

Rationale

1. The number of irreducible polynomials of degree t over $GF(n)$ is $\approx n^t/t$.
2. The probability that a random polynomial is irreducible is $\approx 1/t$, and a fast algorithm exists for testing irreducibility.

R. McEliece, "Public-Key System Based on Algebraic Coding Theory," DSN Progress Report 44, pp. 114–116, 1978.

McEliece cryptosystem

- Proposed by Robert McEliece in 1978.
- Irreducible Goppa codes were used in the original proposal.
- Secret irreducible Goppa code:
  - irreducible polynomial of degree t over $GF(2^m)$,
  - length (maximum): $n = 2^m$,
  - dimension: $k \geq n - t \cdot m$,
  - correction capability: t errors.

Rationale

1. The number of irreducible polynomials of degree t over $GF(n)$ is $\approx n^t/t$.
2. The probability that a random polynomial is irreducible is $\approx 1/t$, and a fast algorithm exists for testing irreducibility.

R. McEliece, "Public-Key System Based on Algebraic Coding Theory," DSN Progress Report 44, pp. 114–116, 1978.

McEliece cryptosystem - key generation

Private key

- $k \times n$ generator matrix $\mathbf{G}$ of a secret Goppa code,
- random dense $k \times k$ non-singular "scrambling" matrix $\mathbf{S}$,
- random $n \times n$ permutation matrix $\mathbf{P}$.

Public key

$$\mathbf{G}' = \mathbf{S} \cdot \mathbf{G} \cdot \mathbf{P}$$

- The public code is permutation equivalent to the secret code.
- Security relies on the hardness of decoding a random-like code.

E. Berlekamp, R. McEliece and H. van Tilborg, "On the inherent intractability of certain coding problems," IEEE Trans. Inf. Theory, vol. 24, no. 3, pp. 384–386, May 1978.

McEliece cryptosystem - key generation

Private key

- $k \times n$ generator matrix $\mathbf{G}$ of a secret Goppa code,
- random dense $k \times k$ non-singular "scrambling" matrix $\mathbf{S}$,
- random $n \times n$ permutation matrix $\mathbf{P}$.

Public key

$$\mathbf{G}' = \mathbf{S} \cdot \mathbf{G} \cdot \mathbf{P}$$

- The public code is permutation equivalent to the secret code.
- Security relies on the hardness of decoding a random-like code.

E. Berlekamp, R. McEliece and H. van Tilborg, "On the inherent intractability of certain coding problems," IEEE Trans. Inf. Theory, vol. 24, no. 3, pp. 384–386, May 1978.

McEliece cryptosystem - key generation

Private key

- $k \times n$ generator matrix $\mathbf{G}$ of a secret Goppa code,
- random dense $k \times k$ non-singular "scrambling" matrix $\mathbf{S}$,
- random $n \times n$ permutation matrix $\mathbf{P}$.

Public key

$$\mathbf{G}' = \mathbf{S} \cdot \mathbf{G} \cdot \mathbf{P}$$

- The public code is permutation equivalent to the secret code.
- Security relies on the hardness of decoding a random-like code.

E. Berlekamp, R. McEliece and H. van Tilborg, "On the inherent intractability of certain coding problems," IEEE Trans. Inf. Theory, vol. 24, no. 3, pp. 384–386, May 1978.

McEliece cryptosystem - encryption

1. Alice gets Bob's public key $\mathbf{G}'$.
2. She generates a random error vector of length n and weight t.
3. She encrypts any k-bit block $\mathbf{u}$ as

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}' + \mathbf{e} = \mathbf{c} + \mathbf{e}$$

Alert

This only provides semantic security!

McEliece cryptosystem - encryption

1. Alice gets Bob's public key $\mathbf{G}'$.
2. She generates a random error vector of length n and weight t.
3. She encrypts any k-bit block $\mathbf{u}$ as

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}' + \mathbf{e} = \mathbf{c} + \mathbf{e}$$

Alert

This only provides semantic security!

McEliece cryptosystem - decryption

1. Bob computes

$$\begin{aligned}\mathbf{x}' &= \mathbf{x} \cdot \mathbf{P}^{-1} = \\ &= (\mathbf{u} \cdot \mathbf{S} \cdot \mathbf{G} \cdot \mathbf{P} + \mathbf{e}) \cdot \mathbf{P}^{-1} = \\ &= \mathbf{u} \cdot \mathbf{S} \cdot \mathbf{G} + \mathbf{e} \cdot \mathbf{P}^{-1}\end{aligned}$$

2. Bob decodes the secret code and obtains

$$\mathbf{u}' = \mathbf{u} \cdot \mathbf{S}$$

3. Bob computes $\mathbf{u} = \mathbf{u}' \cdot \mathbf{S}^{-1}$.

Niederreiter cryptosystem - key generation

Private key

- $r \times n$ parity-check matrix $\mathbf{H}$ of a secret code,
- random dense $r \times r$ non-singular "scrambling" matrix $\mathbf{S}$.

Public key

$$\mathbf{H}' = \mathbf{S} \cdot \mathbf{H}$$

Niederreiter cryptosystem - key generation

Private key

- $r \times n$ parity-check matrix $\mathbf{H}$ of a secret code,
- random dense $r \times r$ non-singular "scrambling" matrix $\mathbf{S}$.

Public key

$$\mathbf{H}' = \mathbf{S} \cdot \mathbf{H}$$

Niederreiter cryptosystem - encryption

1. Alice gets Bob's public key $\mathbf{H}'$.
2. She maps each block of the secret message into an error pattern $\mathbf{e}$ with length n and weight t.
3. She encrypts $\mathbf{e}$ as

$$\mathbf{x} = \mathbf{H}' \cdot \mathbf{e}^{\mathrm{T}} = \mathbf{S} \cdot \mathbf{H} \cdot \mathbf{e}^{\mathrm{T}}$$

Alert

We still only have semantic security!

Niederreiter cryptosystem - encryption

1. Alice gets Bob's public key $\mathbf{H}'$.
2. She maps each block of the secret message into an error pattern $\mathbf{e}$ with length n and weight t.
3. She encrypts $\mathbf{e}$ as

$$\mathbf{x} = \mathbf{H}' \cdot \mathbf{e}^{\mathrm{T}} = \mathbf{S} \cdot \mathbf{H} \cdot \mathbf{e}^{\mathrm{T}}$$

Alert

We still only have semantic security!

Niederreiter cryptosystem - decryption

1. Bob computes
$$\mathbf{x}' = \mathbf{S}^{-1} \cdot \mathbf{x} = \mathbf{H} \cdot \mathbf{e}^{\mathrm{T}}$$
2. Bob performs syndrome decoding of the secret code and obtains $\mathbf{e}$ from $\mathbf{x}'$.
3. He demaps $\mathbf{e}$ into the corresponding secret message block.

McEliece/Niederreiter cryptosystems

- GRS codes originally used in Niederreiter were attacked.
- But Goppa codes resisted cryptanalysis for about 40 years.
- These systems are faster than competing solutions...
- ...but they require large public keys (56 KiB or more for 80-bit security).
- Attacks based on distinguishers pose some threats on high rate Goppa codes.
- They also invalidate all existing McEliece cryptosystem security proofs for high rate Goppa codes.

- D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-Quantum Cryptography*, vol. 5299 of Springer LNCS, pp. 31–46, 2008.
- J.-C. Faugère, V. Gauthier, A. Otmani, L. Perret, and J.-P. Tillich, "A distinguisher for high rate McEliece cryptosystems," In Proc. Information Theory Workshop 2011, pp. 282–286, Paraty, Brasil, 2011.

McEliece/Niederreiter cryptosystems

- GRS codes originally used in Niederreiter were attacked.
- But Goppa codes resisted cryptanalysis for about 40 years.
- These systems are faster than competing solutions...
- ...but they require large public keys (56 KiB or more for 80-bit security).
- Attacks based on distinguishers pose some threats on high rate Goppa codes.
- They also invalidate all existing McEliece cryptosystem security proofs for high rate Goppa codes.

▶ D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-Quantum Cryptography*, vol. 5299 of Springer LNCS, pp. 31–46, 2008.

▶ J.-C. Faugère, V. Gauthier, A. Otmani, L. Perret, and J.-P. Tillich, "A distinguisher for high rate McEliece cryptosystems," In Proc. Information Theory Workshop 2011, pp. 282–286, Paraty, Brasil, 2011.

McEliece/Niederreiter cryptosystems

- GRS codes originally used in Niederreiter were attacked.
- But Goppa codes resisted cryptanalysis for about 40 years.
- These systems are faster than competing solutions...
- ...but they require large public keys (56 KiB or more for 80-bit security).
- Attacks based on distinguishers pose some threats on high rate Goppa codes.
- They also invalidate all existing McEliece cryptosystem security proofs for high rate Goppa codes.

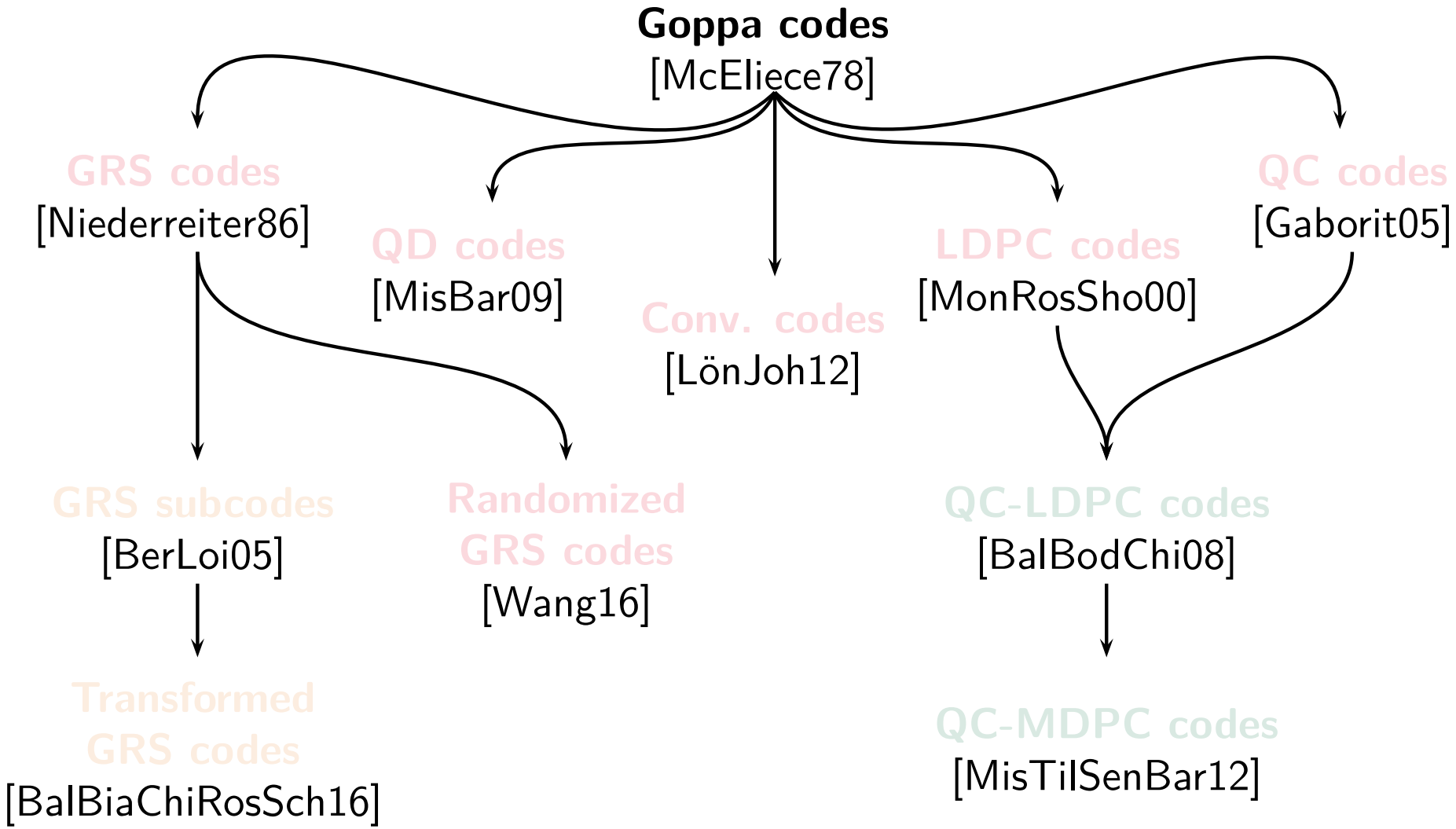
- D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-Quantum Cryptography*, vol. 5299 of Springer LNCS, pp. 31–46, 2008.
- J.-C. Faugère, V. Gauthier, A. Otmani, L. Perret, and J.-P. Tillich, "A distinguisher for high rate McEliece cryptosystems," In Proc. Information Theory Workshop 2011, pp. 282–286, Paraty, Brasil, 2011.

McEliece/Niederreiter cryptosystems

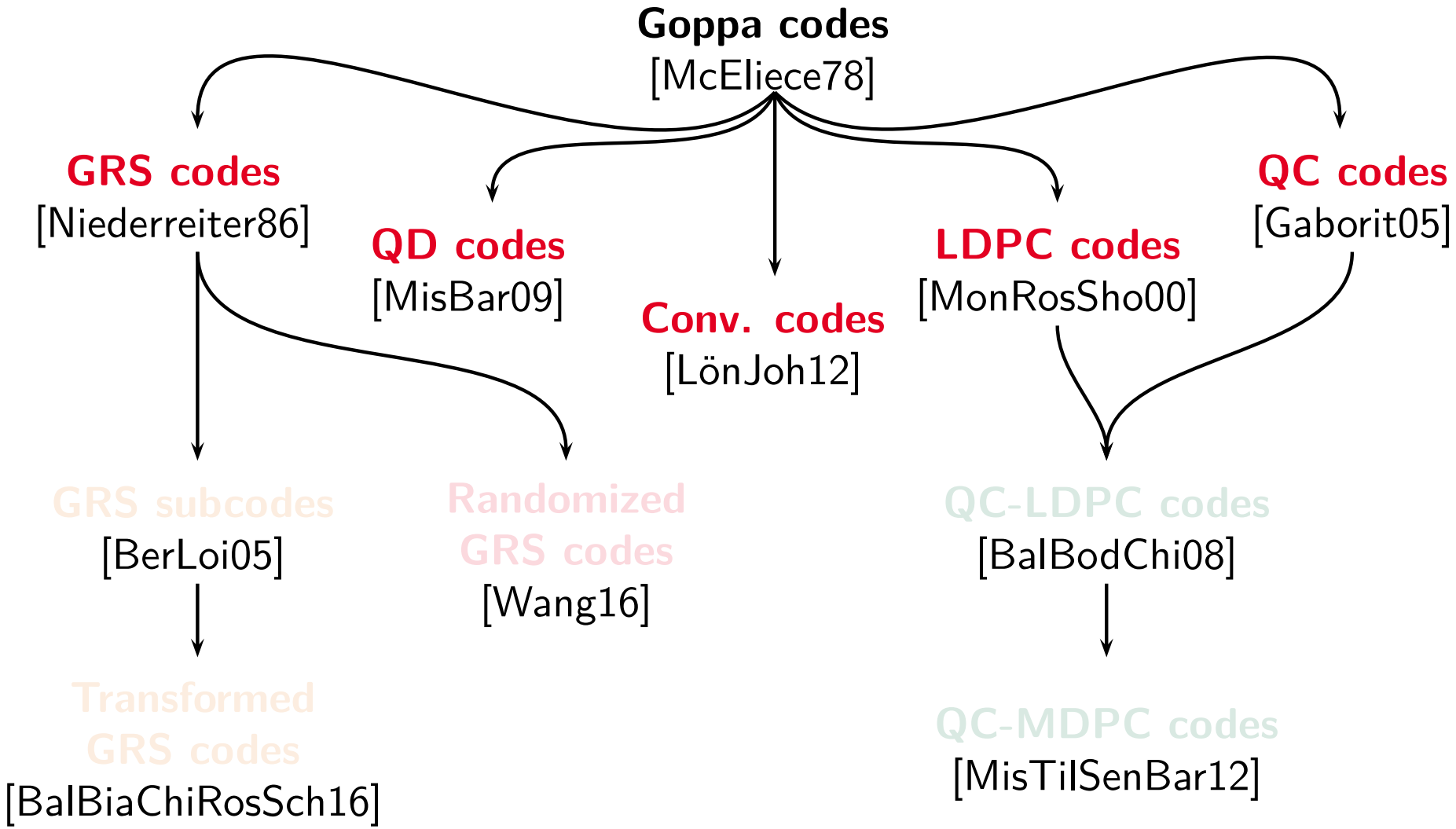
- GRS codes originally used in Niederreiter were attacked.
- But Goppa codes resisted cryptanalysis for about 40 years.
- These systems are faster than competing solutions...
- ...but they require large public keys (56 KiB or more for 80-bit security).
- Attacks based on distinguishers pose some threats on high rate Goppa codes.
- They also invalidate all existing McEliece cryptosystem security proofs for high rate Goppa codes.

- D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-Quantum Cryptography*, vol. 5299 of Springer LNCS, pp. 31–46, 2008.
- J.-C. Faugère, V. Gauthier, A. Otmani, L. Perret, and J.-P. Tillich, "A distinguisher for high rate McEliece cryptosystems," In Proc. Information Theory Workshop 2011, pp. 282–286, Paraty, Brasil, 2011.

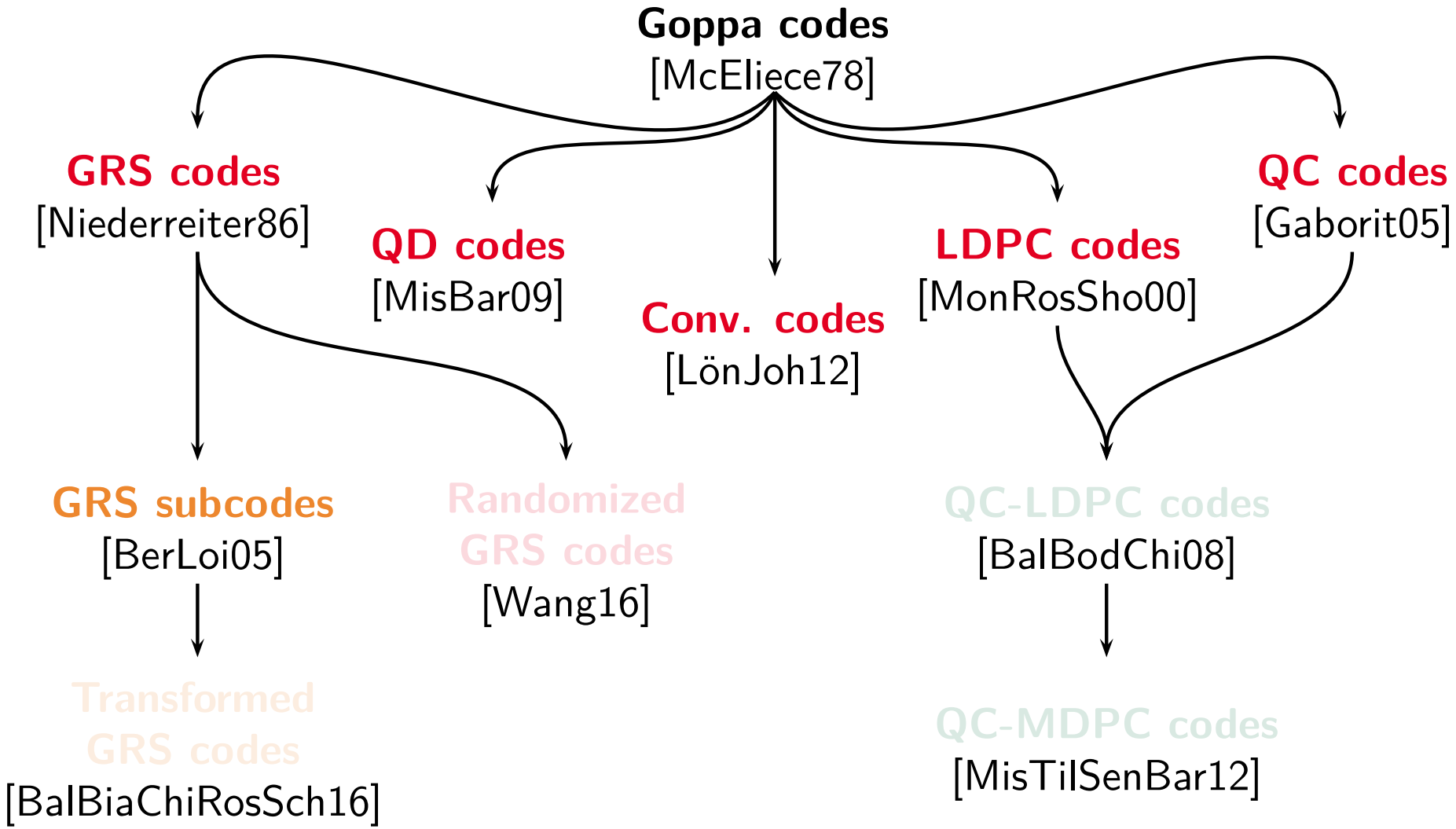
Alternatives to Goppa codes (Hamming metric)



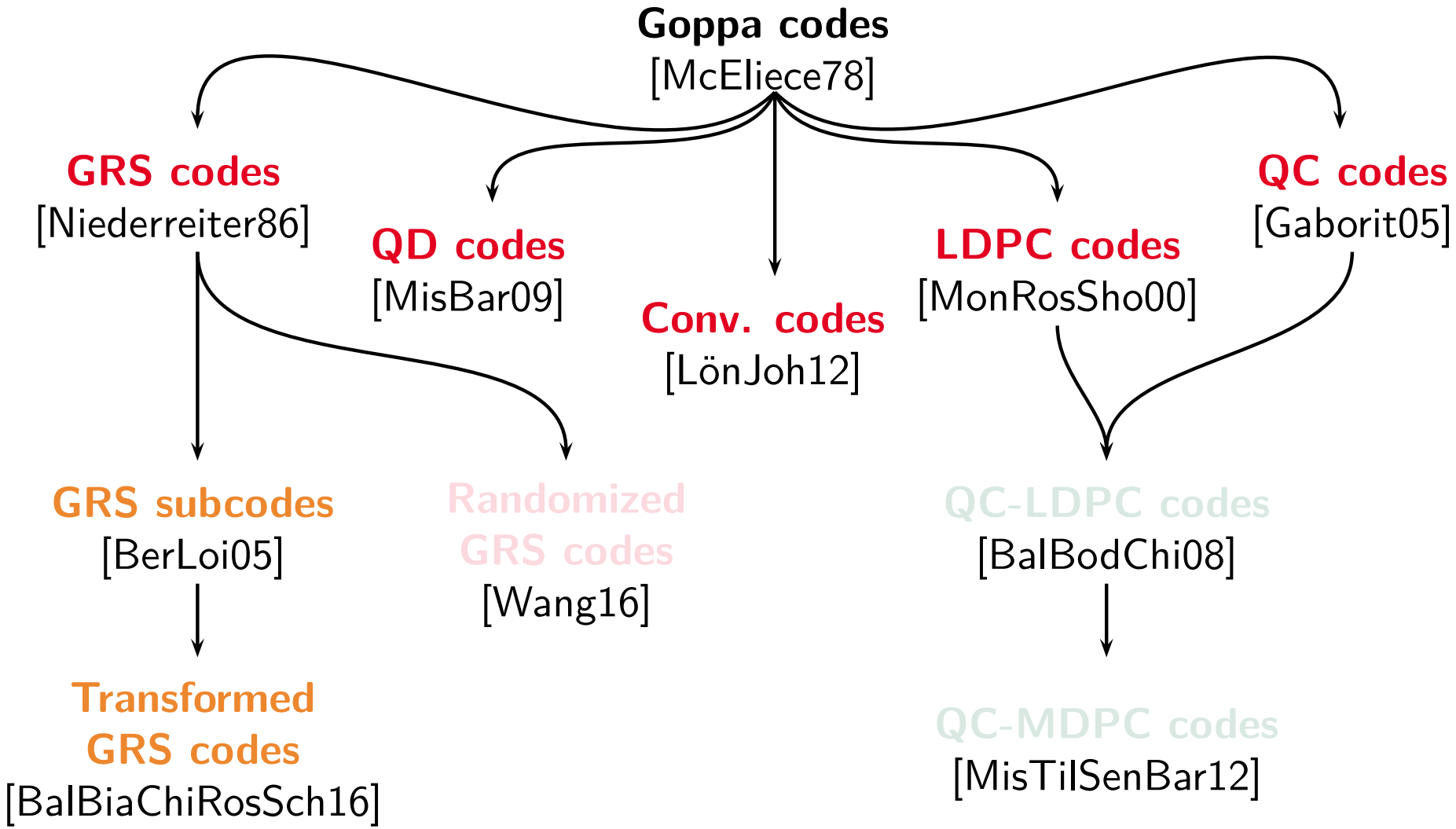
Alternatives to Goppa codes (Hamming metric)



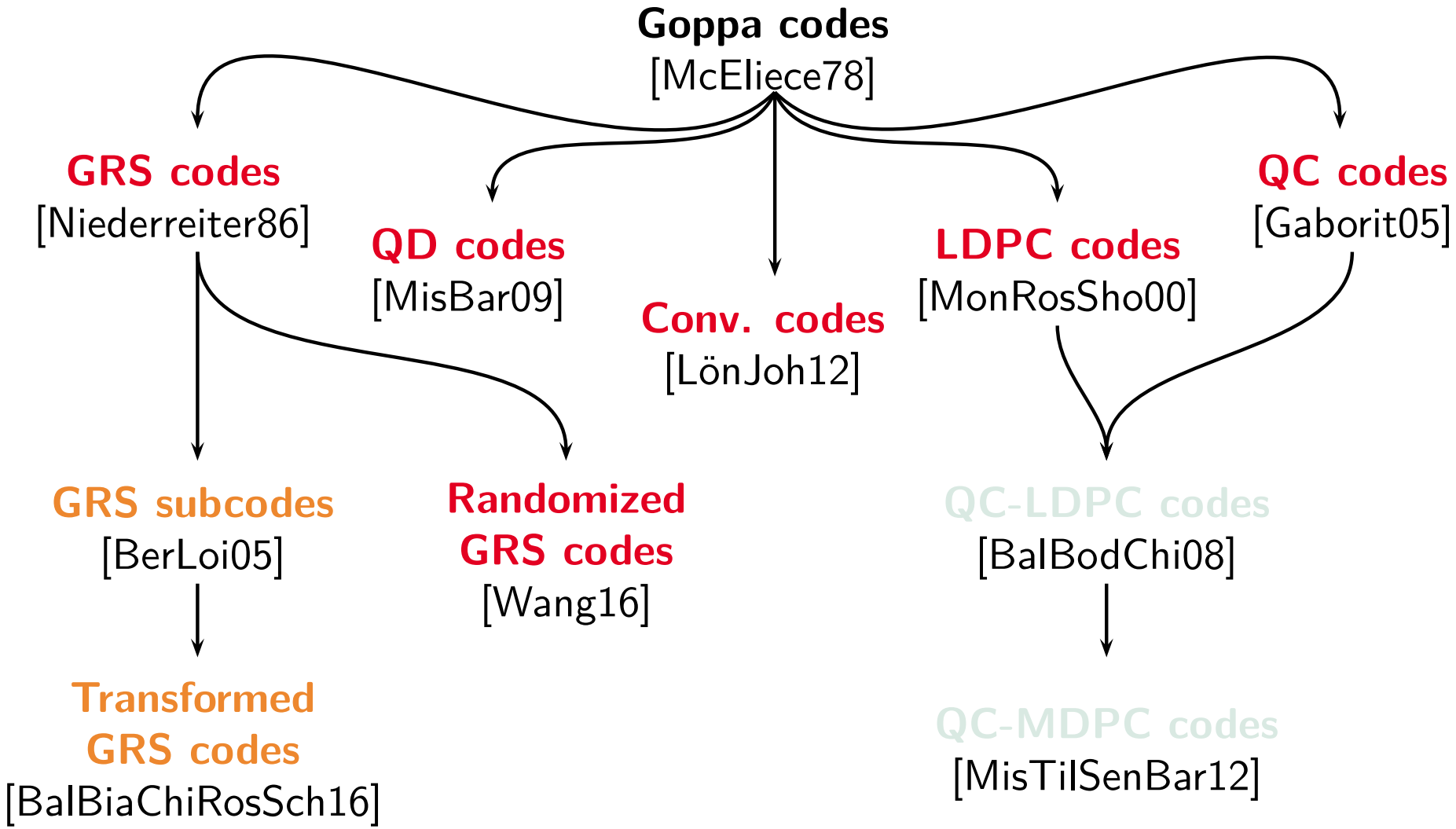
Alternatives to Goppa codes (Hamming metric)



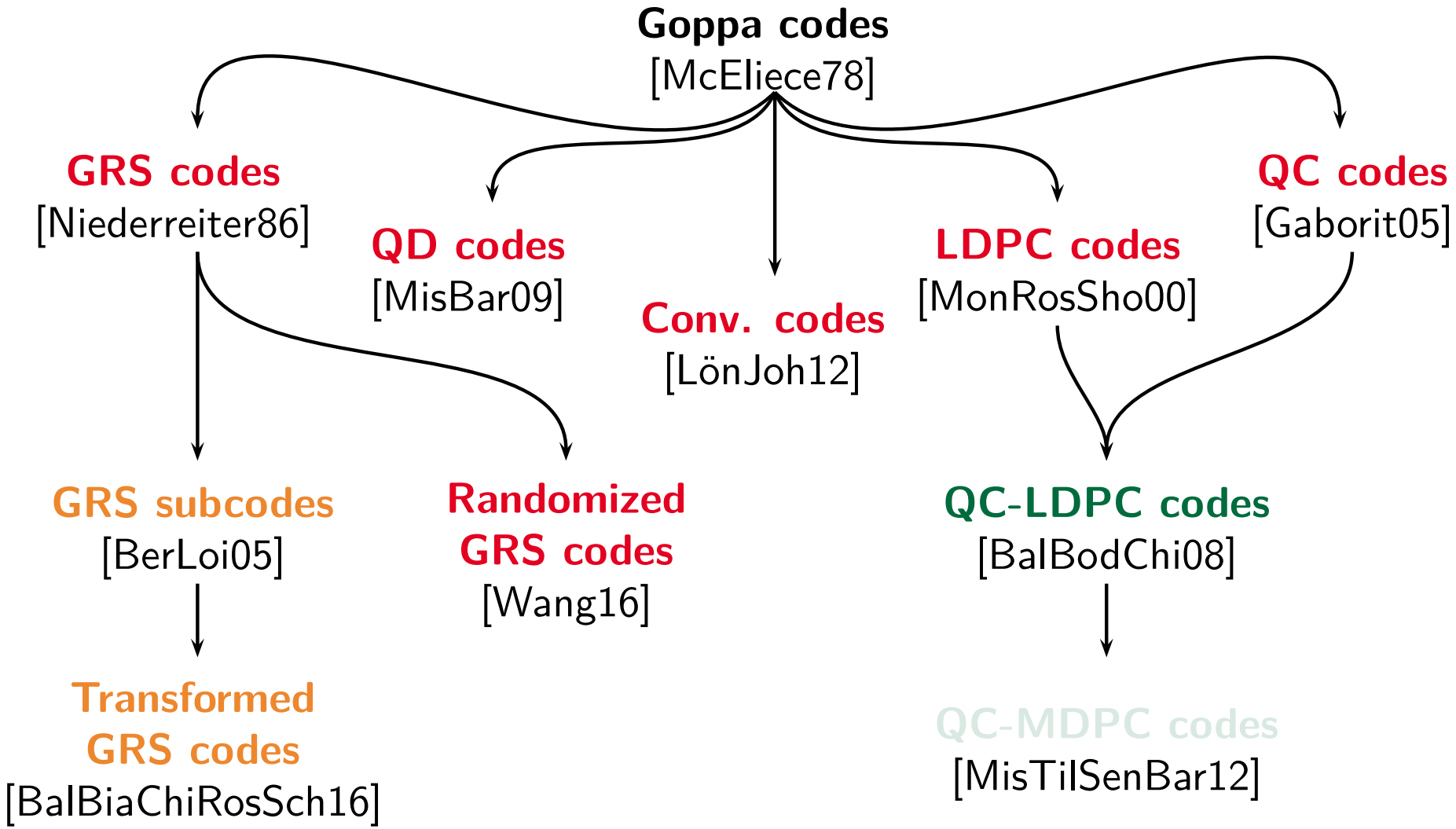
Alternatives to Goppa codes (Hamming metric)



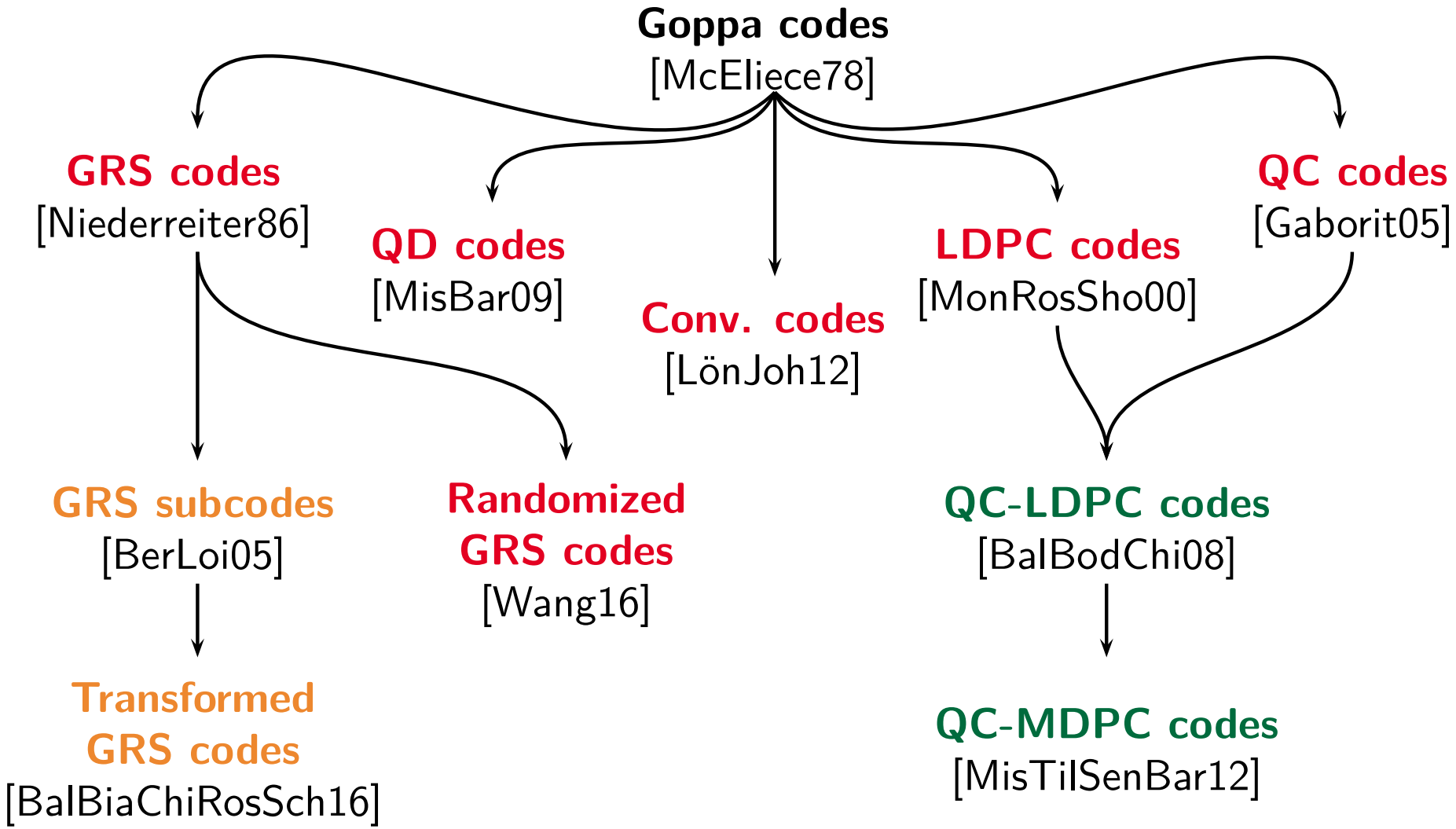
Alternatives to Goppa codes (Hamming metric)



Alternatives to Goppa codes (Hamming metric)



Alternatives to Goppa codes (Hamming metric)



Variants in other metrics (rank)

- Gabidulin codes are optimal codes in the rank metric (like GRS codes in the Hamming metric).
- The Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem exploits them to build a code-based trapdoor in the rank metric domain.
- Using codes in the rank metric prevents the use of information set decoding approaches.
- But other dangerous attacks have been found.
- Secure instances still exist, with reasonably small keys.
- Complexity of decoding in the rank domain may be an issue.

- R. Overbeck, "Structural attacks for public key cryptosystems based on Gabidulin codes," Journal of Cryptology, vol. 21, no. 2, pp. 280–301, 2008.
- H. Rashwan, E.M. Gabidulin, B. Honary, "Security of the GPT cryptosystem and its applications to cryptography," Security and Communication Networks. vol. 4, no. 8, pp. 937–946, 2011.

Variants in other metrics (rank)

- Gabidulin codes are optimal codes in the rank metric (like GRS codes in the Hamming metric).
- The Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem exploits them to build a code-based trapdoor in the rank metric domain.
- Using codes in the rank metric prevents the use of information set decoding approaches.
- But other dangerous attacks have been found.
- Secure instances still exist, with reasonably small keys.
- Complexity of decoding in the rank domain may be an issue.

- R. Overbeck, "Structural attacks for public key cryptosystems based on Gabidulin codes," Journal of Cryptology, vol. 21, no. 2, pp. 280–301, 2008.
- H. Rashwan, E.M. Gabidulin, B. Honary, "Security of the GPT cryptosystem and its applications to cryptography," Security and Communication Networks. vol. 4, no. 8, pp. 937–946, 2011.

Variants in other metrics (rank)

- Gabidulin codes are optimal codes in the rank metric (like GRS codes in the Hamming metric).
- The Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem exploits them to build a code-based trapdoor in the rank metric domain.
- Using codes in the rank metric prevents the use of information set decoding approaches.
- But other dangerous attacks have been found.
- Secure instances still exist, with reasonably small keys.
- Complexity of decoding in the rank domain may be an issue.

- R. Overbeck, "Structural attacks for public key cryptosystems based on Gabidulin codes," Journal of Cryptology, vol. 21, no. 2, pp. 280–301, 2008.
- H. Rashwan, E.M. Gabidulin, B. Honary, "Security of the GPT cryptosystem and its applications to cryptography," Security and Communication Networks. vol. 4, no. 8, pp. 937–946, 2011.

Variants in other metrics (rank)

- Gabidulin codes are optimal codes in the rank metric (like GRS codes in the Hamming metric).
- The Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem exploits them to build a code-based trapdoor in the rank metric domain.
- Using codes in the rank metric prevents the use of information set decoding approaches.
- But other dangerous attacks have been found.
- Secure instances still exist, with reasonably small keys.
- Complexity of decoding in the rank domain may be an issue.

R. Overbeck, "Structural attacks for public key cryptosystems based on Gabidulin codes," Journal of Cryptology, vol. 21, no. 2, pp. 280–301, 2008.

H. Rashwan, E.M. Gabidulin, B. Honary, "Security of the GPT cryptosystem and its applications to cryptography," Security and Communication Networks. vol. 4, no. 8, pp. 937–946, 2011.

Variants in other metrics (rank)

- Gabidulin codes are optimal codes in the rank metric (like GRS codes in the Hamming metric).
- The Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem exploits them to build a code-based trapdoor in the rank metric domain.
- Using codes in the rank metric prevents the use of information set decoding approaches.
- But other dangerous attacks have been found.
- Secure instances still exist, with reasonably small keys.
- Complexity of decoding in the rank domain may be an issue.

- R. Overbeck, "Structural attacks for public key cryptosystems based on Gabidulin codes," Journal of Cryptology, vol. 21, no. 2, pp. 280–301, 2008.
- H. Rashwan, E.M. Gabidulin, B. Honary, "Security of the GPT cryptosystem and its applications to cryptography," Security and Communication Networks. vol. 4, no. 8, pp. 937–946, 2011.

Variants in other metrics (rank)

- Gabidulin codes are optimal codes in the rank metric (like GRS codes in the Hamming metric).
- The Gabidulin-Paramonov-Tretjakov (GPT) cryptosystem exploits them to build a code-based trapdoor in the rank metric domain.
- Using codes in the rank metric prevents the use of information set decoding approaches.
- But other dangerous attacks have been found.
- Secure instances still exist, with reasonably small keys.
- Complexity of decoding in the rank domain may be an issue.

- R. Overbeck, "Structural attacks for public key cryptosystems based on Gabidulin codes," Journal of Cryptology, vol. 21, no. 2, pp. 280–301, 2008.
- H. Rashwan, E.M. Gabidulin, B. Honary, "Security of the GPT cryptosystem and its applications to cryptography," Security and Communication Networks. vol. 4, no. 8, pp. 937–946, 2011.

Variants of interest

We focus on variants using:

- GRS codes
- LDPC/MDPC codes

Using generalized Reed-Solomon codes

Advantages

- GRS codes are maximum distance separable (MDS) codes, while Goppa codes are subfield subcodes of them.
- Using GRS codes would allow to reduce the public key size thanks to their optimum error correction capability.
- GRS codes are widespread and enjoy optimized implementations of encoding and decoding algorithms.

Drawbacks

- GRS codes have more intrinsic structure than Goppa codes.

▶ V. M. Sidelnikov, S. O. Shestakov, "On insecurity of cryptosystems based on generalized Reed-Solomon codes," Discrete Mathematics and Applications, vol. 2, no. 4, pp. 439–444, 1992.

Using generalized Reed-Solomon codes

Advantages

- GRS codes are maximum distance separable (MDS) codes, while Goppa codes are subfield subcodes of them.
- Using GRS codes would allow to reduce the public key size thanks to their optimum error correction capability.
- GRS codes are widespread and enjoy optimized implementations of encoding and decoding algorithms.

Drawbacks

- GRS codes have more intrinsic structure than Goppa codes.

▶ V. M. Sidelnikov, S. O. Shestakov, "On insecurity of cryptosystems based on generalized Reed-Solomon codes," Discrete Mathematics and Applications, vol. 2, no. 4, pp. 439–444, 1992.

Berger-Loidreau cryptosystem

- It is difficult to hide the structure of a secret GRS code $\mathcal{C}(n, k, d)$.
- Berger and Loidreau proposed to resort to a subcode of $\mathcal{C}$, namely $\mathcal{C}'(n, k - l, d')$, with $d' \geq d$.
- The subcode is constructed by adding l rows to the parity-check matrix of $\mathcal{C}$.
- The decoding algorithm of $\mathcal{C}$ can still be used to correct up to $t = \lfloor (d-1)/2 \rfloor$ errors.

Weakness

An attack devised by Wieschebrink in 2010 severely limits the choices of parameters that prevent recovering the secret subcode.

- T. P. Berger, P. Loidreau, "How to Mask the Structure of Codes for a Cryptographic Use," Designs, Codes and Cryptography, vol. 35, no. 1, pp. 63–79, 2005.
- C. Wieschebrink, "Cryptanalysis of the Niederreiter public key scheme based on GRS subcodes," Post-Quantum Cryptography (PQCrypto 2010), vol. 6061 of Springer LNCS, pp. 61–72, 2010.

Berger-Loidreau cryptosystem

- It is difficult to hide the structure of a secret GRS code $\mathcal{C}(n, k, d)$.
- Berger and Loidreau proposed to resort to a subcode of $\mathcal{C}$, namely $\mathcal{C}'(n, k - l, d')$, with $d' \geq d$.
- The subcode is constructed by adding l rows to the parity-check matrix of $\mathcal{C}$.
- The decoding algorithm of $\mathcal{C}$ can still be used to correct up to $t = \lfloor (d-1)/2 \rfloor$ errors.

Weakness

An attack devised by Wieschebrink in 2010 severely limits the choices of parameters that prevent recovering the secret subcode.

- T. P. Berger, P. Loidreau, "How to Mask the Structure of Codes for a Cryptographic Use," Designs, Codes and Cryptography, vol. 35, no. 1, pp. 63–79, 2005.
- C. Wieschebrink, "Cryptanalysis of the Niederreiter public key scheme based on GRS subcodes," Post-Quantum Cryptography (PQCrypto 2010), vol. 6061 of Springer LNCS, pp. 61–72, 2010.

BBCRS cryptosystem

- An alternative approach to disguise the secret GRS code is to avoid permutation equivalence between the private and public codes.
- The main idea is to replace the classical permutation matrix $\mathbf{P}$ with a more general transformation matrix $\mathbf{Q}$.
- The price to pay is an increase in the number of errors to be corrected by Bob.
- But $\mathbf{Q}$ can be designed in such a way as to keep this number limited.

▶ M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, D. Schipani, "Enhanced Public Key Security for the McEliece Cryptosystem," Journal of Cryptology, Vol. 29, No. 1, pp 1–27, 2016.

BBCRS cryptosystem

- An alternative approach to disguise the secret GRS code is to avoid permutation equivalence between the private and public codes.
- The main idea is to replace the classical permutation matrix $\mathbf{P}$ with a more general transformation matrix $\mathbf{Q}$.
- The price to pay is an increase in the number of errors to be corrected by Bob.
- But $\mathbf{Q}$ can be designed in such a way as to keep this number limited.

M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, D. Schipani, "Enhanced Public Key Security for the McEliece Cryptosystem," Journal of Cryptology, Vol. 29, No. 1, pp 1–27, 2016.

BBCRS cryptosystem

- An alternative approach to disguise the secret GRS code is to avoid permutation equivalence between the private and public codes.
- The main idea is to replace the classical permutation matrix $\mathbf{P}$ with a more general transformation matrix $\mathbf{Q}$.
- The price to pay is an increase in the number of errors to be corrected by Bob.
- But $\mathbf{Q}$ can be designed in such a way as to keep this number limited.

▶ M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, D. Schipani, "Enhanced Public Key Security for the McEliece Cryptosystem," Journal of Cryptology, Vol. 29, No. 1, pp 1–27, 2016.

BBCRS cryptosystem - key generation

Private key

- $r \times n$ parity-check matrix $\mathbf{H}$ of a GRS code with length n and dimension $k = n - r$, defined over $\mathbb{F}_q$, able to correct t errors.
- random non-singular $r \times r$ scrambling matrix $\mathbf{S}$.
- secret non-singular $n \times n$ transformation matrix $\mathbf{Q}$.

Public key

$$\mathbf{H}' = \mathbf{S}^{-1} \cdot \mathbf{H} \cdot \mathbf{Q}^T.$$

- The public code is no longer permutation equivalent to the secret code.

BBCRS cryptosystem - key generation

Private key

- $r \times n$ parity-check matrix $\mathbf{H}$ of a GRS code with length n and dimension $k = n - r$, defined over $\mathbb{F}_q$, able to correct t errors.
- random non-singular $r \times r$ scrambling matrix $\mathbf{S}$.
- secret non-singular $n \times n$ transformation matrix $\mathbf{Q}$.

Public key

$$\mathbf{H}' = \mathbf{S}^{-1} \cdot \mathbf{H} \cdot \mathbf{Q}^T.$$

- The public code is no longer permutation equivalent to the secret code.

BBCRS cryptosystem - key generation

Private key

- $r \times n$ parity-check matrix $\mathbf{H}$ of a GRS code with length n and dimension $k = n - r$, defined over $\mathbb{F}_q$, able to correct t errors.
- random non-singular $r \times r$ scrambling matrix $\mathbf{S}$.
- secret non-singular $n \times n$ transformation matrix $\mathbf{Q}$.

Public key

$$\mathbf{H}' = \mathbf{S}^{-1} \cdot \mathbf{H} \cdot \mathbf{Q}^T.$$

- The public code is no longer permutation equivalent to the secret code.

BBCRS cryptosystem - design of Q

- Alice randomly chooses two matrices ($\mathbf{a}$ and $\mathbf{b}$) defined over $\mathbb{F}_q$, having size $z \times n$ ($z \ll n$) and rank z.
- Alice computes $\mathbf{R} = \mathbf{a}^T \cdot \mathbf{b}$, having size $n \times n$ and rank $z \ll n$.
- Alice randomly chooses a sparse $n \times n$ matrix $\mathbf{T}$ with average row and column weight $m \ll n$.

Matrix Q

$$\mathbf{Q} = \mathbf{R} + \mathbf{T}$$

Rationale

$\mathbf{Q}$ is made of two components:

- $\mathbf{R}$ is dense, but has low rank and is canceled by Bob.
- $\mathbf{T}$ is sparse, and keeps errors to be corrected by Bob limited.

BBCRS cryptosystem - design of Q

- Alice randomly chooses two matrices ($\mathbf{a}$ and $\mathbf{b}$) defined over $\mathbb{F}_q$, having size $z \times n$ ($z \ll n$) and rank z.
- Alice computes $\mathbf{R} = \mathbf{a}^T \cdot \mathbf{b}$, having size $n \times n$ and rank $z \ll n$.
- Alice randomly chooses a sparse $n \times n$ matrix $\mathbf{T}$ with average row and column weight $m \ll n$.

Matrix Q

$$\mathbf{Q} = \mathbf{R} + \mathbf{T}$$

Rationale

$\mathbf{Q}$ is made of two components:

- $\mathbf{R}$ is dense, but has low rank and is canceled by Bob.
- $\mathbf{T}$ is sparse, and keeps errors to be corrected by Bob limited.

BBCRS cryptosystem - design of Q

- Alice randomly chooses two matrices ($\mathbf{a}$ and $\mathbf{b}$) defined over $\mathbb{F}_q$, having size $z \times n$ ($z \ll n$) and rank z.
- Alice computes $\mathbf{R} = \mathbf{a}^T \cdot \mathbf{b}$, having size $n \times n$ and rank $z \ll n$.
- Alice randomly chooses a sparse $n \times n$ matrix $\mathbf{T}$ with average row and column weight $m \ll n$.

Matrix Q

$$\mathbf{Q} = \mathbf{R} + \mathbf{T}$$

Rationale

$\mathbf{Q}$ is made of two components:

- $\mathbf{R}$ is dense, but has low rank and is canceled by Bob.
- $\mathbf{T}$ is sparse, and keeps errors to be corrected by Bob limited.

BBCRS cryptosystem - encryption

1. Alice maps the cleartext vector into an error vector $\mathbf{e}$ having weight $t_p = \left\lfloor \frac{t}{m} \right\rfloor \leq t$.
2. Alice computes the ciphertext as the syndrome

$$\mathbf{x} = \mathbf{H}' \cdot \mathbf{e}^T$$

Number of intentional errors

Differently from the original McEliece/Niederreiter systems, where $t_p = t$, this system uses $t_p = \left\lfloor \frac{t}{m} \right\rfloor$.

BBCRS cryptosystem - encryption

1. Alice maps the cleartext vector into an error vector $\mathbf{e}$ having weight $t_p = \left\lfloor \frac{t}{m} \right\rfloor \leq t$.
2. Alice computes the ciphertext as the syndrome

$$\mathbf{x} = \mathbf{H}' \cdot \mathbf{e}^T$$

Number of intentional errors

Differently from the original McEliece/Niederreiter systems, where $t_p = t$, this system uses $t_p = \left\lfloor \frac{t}{m} \right\rfloor$.

BBCRS cryptosystem - decryption

1. Bob computes

$$\begin{aligned}\mathbf{x}' &= \mathbf{S} \cdot \mathbf{x} \\ &= \mathbf{H} \cdot \mathbf{Q}^T \cdot \mathbf{e}^T \\ &= \mathbf{H} \cdot (\mathbf{e} \cdot \mathbf{Q})^T \\ &= \mathbf{H} \cdot [\mathbf{e} \cdot (\mathbf{R} + \mathbf{T})]^T \\ &= \mathbf{H} \cdot \mathbf{b}^T \cdot \gamma + \mathbf{H} \cdot \mathbf{T}^T \cdot \mathbf{e}^T\end{aligned}$$

   where $\gamma = \mathbf{a} \cdot \mathbf{e}^T$.
2. Bob guesses the value of γ.
3. Bob computes

$$\begin{aligned}\mathbf{x}'' &= \mathbf{x}' - \mathbf{H} \cdot \mathbf{b}^T \cdot \gamma \\ &= \mathbf{H} \cdot \mathbf{T}^T \cdot \mathbf{e}^T \\ &= \mathbf{H} \cdot \mathbf{e}_T^T.\end{aligned}$$

BBCRS cryptosystem - decryption

1. Bob computes

$$\begin{aligned}\mathbf{x}' &= \mathbf{S} \cdot \mathbf{x} \\ &= \mathbf{H} \cdot \mathbf{Q}^T \cdot \mathbf{e}^T \\ &= \mathbf{H} \cdot (\mathbf{e} \cdot \mathbf{Q})^T \\ &= \mathbf{H} \cdot [\mathbf{e} \cdot (\mathbf{R} + \mathbf{T})]^T \\ &= \mathbf{H} \cdot \mathbf{b}^T \cdot \gamma + \mathbf{H} \cdot \mathbf{T}^T \cdot \mathbf{e}^T\end{aligned}$$

   where $\gamma = \mathbf{a} \cdot \mathbf{e}^T$.
2. Bob guesses the value of γ.
3. Bob computes

$$\begin{aligned}\mathbf{x}'' &= \mathbf{x}' - \mathbf{H} \cdot \mathbf{b}^T \cdot \gamma \\ &= \mathbf{H} \cdot \mathbf{T}^T \cdot \mathbf{e}^T \\ &= \mathbf{H} \cdot \mathbf{e}_T^T.\end{aligned}$$

BBCRS cryptosystem - decryption

1. Bob computes

$$\begin{aligned}\mathbf{x}' &= \mathbf{S} \cdot \mathbf{x} \\ &= \mathbf{H} \cdot \mathbf{Q}^T \cdot \mathbf{e}^T \\ &= \mathbf{H} \cdot (\mathbf{e} \cdot \mathbf{Q})^T \\ &= \mathbf{H} \cdot [\mathbf{e} \cdot (\mathbf{R} + \mathbf{T})]^T \\ &= \mathbf{H} \cdot \mathbf{b}^T \cdot \gamma + \mathbf{H} \cdot \mathbf{T}^T \cdot \mathbf{e}^T\end{aligned}$$

   where $\gamma = \mathbf{a} \cdot \mathbf{e}^T$.
2. Bob guesses the value of γ.
3. Bob computes

$$\begin{aligned}\mathbf{x}'' &= \mathbf{x}' - \mathbf{H} \cdot \mathbf{b}^T \cdot \gamma \\ &= \mathbf{H} \cdot \mathbf{T}^T \cdot \mathbf{e}^T \\ &= \mathbf{H} \cdot \mathbf{e}_T^T.\end{aligned}$$

BBCRS cryptosystem - decryption (ctd.)

4. Bob recovers $\mathbf{e}_T = \mathbf{e} \cdot \mathbf{T}$ (with weight $\leq t$) by syndrome decoding through the private code.
5. Bob computes $\mathbf{e} = \mathbf{e}_T \cdot \mathbf{T}^{-1}$.
6. Bob demaps $\mathbf{e}$ into its associated cleartext vector to get the secret message.

Note 1

Since $\mathbf{T}^T \cdot \mathbf{e}^T$ has weight $\leq m \cdot t_p \leq t$, $\mathbf{x}''$ is a correctable syndrome.

Note 2

Bob needs to perform, on average, $q^z/2$ attempts to guess γ (i.e., z values over $\mathbb{F}_q$)

BBCRS cryptosystem - decryption (ctd.)

4. Bob recovers $\mathbf{e}_T = \mathbf{e} \cdot \mathbf{T}$ (with weight $\leq t$) by syndrome decoding through the private code.
5. Bob computes $\mathbf{e} = \mathbf{e}_T \cdot \mathbf{T}^{-1}$.
6. Bob demaps $\mathbf{e}$ into its associated cleartext vector to get the secret message.

Note 1

Since $\mathbf{T}^T \cdot \mathbf{e}^T$ has weight $\leq m \cdot t_p \leq t$, $\mathbf{x}''$ is a correctable syndrome.

Note 2

Bob needs to perform, on average, $q^z/2$ attempts to guess γ (i.e., z values over $\mathbb{F}_q$)

BBCRS cryptosystem - decryption (ctd.)

4. Bob recovers $\mathbf{e}_T = \mathbf{e} \cdot \mathbf{T}$ (with weight $\leq t$) by syndrome decoding through the private code.
5. Bob computes $\mathbf{e} = \mathbf{e}_T \cdot \mathbf{T}^{-1}$.
6. Bob demaps $\mathbf{e}$ into its associated cleartext vector to get the secret message.

Note 1

Since $\mathbf{T}^T \cdot \mathbf{e}^T$ has weight $\leq m \cdot t_p \leq t$, $\mathbf{x}''$ is a correctable syndrome.

Note 2

Bob needs to perform, on average, $q^z/2$ attempts to guess γ (i.e., z values over $\mathbb{F}_q$)

Attacks against the BBCRS cryptosystem

- Some polynomial-time attacks based on distinguishers have been devised against the original BBCRS scheme instances.
- These attacks can be applied if:

$$\begin{cases} 1 \leq m \leq 1 + R - \frac{1}{n} - \sqrt{\frac{8}{n}R + \frac{1}{n^2}} < 2, \\ z = 1. \end{cases}$$

- Hence we need to:
  - Increase z (thus increasing the decryption complexity), or
  - increase m (thus increasing the number of errors to be corrected).
- Choosing $z = 1$ and $m > 1 + R$ avoids these attacks.

- A. Couvreur, P. Gaborit, V. Gauthier-Umaña, A. Otmani, J.-P. Tillich, "Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes," Designs, Codes and Cryptography, vol. 73, pp. 641–666, 2014.
- A. Couvreur, A. Otmani, J.-P. Tillich, V. Gauthier-Umaña, "A Polynomial-Time Attack on the BBCRS Scheme," Public-Key Cryptography (PKC 2015), vol. 9020 of Springer LNCS, pp. 175–193, 2015.

Attacks against the BBCRS cryptosystem

- Some polynomial-time attacks based on distinguishers have been devised against the original BBCRS scheme instances.
- These attacks can be applied if:

$$\begin{cases} 1 \leq m \leq 1 + R - \frac{1}{n} - \sqrt{\frac{8}{n}R + \frac{1}{n^2}} < 2, \\ z = 1. \end{cases}$$

- Hence we need to:
  - Increase z (thus increasing the decryption complexity), or
  - increase m (thus increasing the number of errors to be corrected).
- Choosing $z = 1$ and $m > 1 + R$ avoids these attacks.

- A. Couvreur, P. Gaborit, V. Gauthier-Umaña, A. Otmani, J.-P. Tillich, "Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes," Designs, Codes and Cryptography, vol. 73, pp. 641–666, 2014.
- A. Couvreur, A. Otmani, J.-P. Tillich, V. Gauthier-Umaña, "A Polynomial-Time Attack on the BBCRS Scheme," Public-Key Cryptography (PKC 2015), vol. 9020 of Springer LNCS, pp. 175–193, 2015.

Attacks against the BBCRS cryptosystem

- Some polynomial-time attacks based on distinguishers have been devised against the original BBCRS scheme instances.
- These attacks can be applied if:

$$\begin{cases} 1 \leq m \leq 1 + R - \frac{1}{n} - \sqrt{\frac{8}{n}R + \frac{1}{n^2}} < 2, \\ z = 1. \end{cases}$$

- Hence we need to:
  - Increase z (thus increasing the decryption complexity), or
  - increase m (thus increasing the number of errors to be corrected).
- Choosing $z = 1$ and $m > 1 + R$ avoids these attacks.

- A. Couvreur, P. Gaborit, V. Gauthier-Umaña, A. Otmani, J.-P. Tillich, "Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes," Designs, Codes and Cryptography, vol. 73, pp. 641–666, 2014.
- A. Couvreur, A. Otmani, J.-P. Tillich, V. Gauthier-Umaña, "A Polynomial-Time Attack on the BBCRS Scheme," Public-Key Cryptography (PKC 2015), vol. 9020 of Springer LNCS, pp. 175–193, 2015.

Secure BBCRS cryptosystem instances

Table: Goppa code-based McEliece/Niederreiter cryptosystem parameters for $SL = 180$ and $SL = 260$ bits.

SL	n	k	$t = t_p$	WF	KS	R_e(Nied.)	R_e(McEl.)
180	4096	3004	91	180.06	400.44	0.5724	0.7334
260	8192	6957	95	260.57	1048.82	0.6012	0.8492

Table: BBCRS cryptosystem parameters for $SL = 180$ and $SL = 260$ bits.

SL	n	k	m	t	t_p	WF	KS	R_e(Nied.)	R_e(McEl.)
180	946	504	1.686	221	131	180.24	268.87	0.4209	0.5328
260	1422	786	1.708	318	186	260.26	639.19	0.4111	0.5527

Evolution of the BBCRS cryptosystem

- Increasing m and z has a detrimental effect on the public key size and complexity of the original BBCRS system.
- In a new BCRSS variant $\mathbf{a}$ is made public, thus:
  - Alice can compute $\gamma = \mathbf{a} \cdot \mathbf{e}^T$ and send it to Bob.
  - Bob no longer needs to guess γ.
- z can be increased without increasing the decryption complexity.
- Some information on $\mathbf{R}$ is leaked, but secure parameter sets can be designed.
- Significant advantages over Goppa code-based systems and other GRS code-based systems are achieved.

M. Baldi, F. Chiaraluce, J. Rosenthal, P. Santini and D. Schipani, "Security of generalised Reed–Solomon code-based cryptosystems," IET Information Security, vol. 13, no. 4, pp. 404–410, 2019.

Evolution of the BBCRS cryptosystem

- Increasing m and z has a detrimental effect on the public key size and complexity of the original BBCRS system.
- In a new BCRSS variant $\mathbf{a}$ is made public, thus:
  - Alice can compute $\gamma = \mathbf{a} \cdot \mathbf{e}^T$ and send it to Bob.
  - Bob no longer needs to guess γ.
- z can be increased without increasing the decryption complexity.
- Some information on $\mathbf{R}$ is leaked, but secure parameter sets can be designed.
- Significant advantages over Goppa code-based systems and other GRS code-based systems are achieved.

M. Baldi, F. Chiaraluce, J. Rosenthal, P. Santini and D. Schipani, "Security of generalised Reed–Solomon code-based cryptosystems," IET Information Security, vol. 13, no. 4, pp. 404–410, 2019.

Evolution of the BBCRS cryptosystem

- Increasing m and z has a detrimental effect on the public key size and complexity of the original BBCRS system.
- In a new BCRSS variant $\mathbf{a}$ is made public, thus:
  - Alice can compute $\gamma = \mathbf{a} \cdot \mathbf{e}^T$ and send it to Bob.
  - Bob no longer needs to guess γ.
- z can be increased without increasing the decryption complexity.
- Some information on $\mathbf{R}$ is leaked, but secure parameter sets can be designed.
- Significant advantages over Goppa code-based systems and other GRS code-based systems are achieved.

M. Baldi, F. Chiaraluce, J. Rosenthal, P. Santini and D. Schipani, "Security of generalised Reed–Solomon code-based cryptosystems," IET Information Security, vol. 13, no. 4, pp. 404–410, 2019.

Evolution of the BBCRS cryptosystem

- Increasing m and z has a detrimental effect on the public key size and complexity of the original BBCRS system.
- In a new BCRSS variant $\mathbf{a}$ is made public, thus:
  - Alice can compute $\gamma = \mathbf{a} \cdot \mathbf{e}^T$ and send it to Bob.
  - Bob no longer needs to guess γ.
- z can be increased without increasing the decryption complexity.
- Some information on $\mathbf{R}$ is leaked, but secure parameter sets can be designed.
- Significant advantages over Goppa code-based systems and other GRS code-based systems are achieved.

M. Baldi, F. Chiaraluce, J. Rosenthal, P. Santini and D. Schipani, "Security of generalised Reed–Solomon code-based cryptosystems," IET Information Security, vol. 13, no. 4, pp. 404–410, 2019.

Evolution of the BBCRS cryptosystem

- Increasing m and z has a detrimental effect on the public key size and complexity of the original BBCRS system.
- In a new BCRSS variant $\mathbf{a}$ is made public, thus:
  - Alice can compute $\gamma = \mathbf{a} \cdot \mathbf{e}^T$ and send it to Bob.
  - Bob no longer needs to guess γ.
- z can be increased without increasing the decryption complexity.
- Some information on $\mathbf{R}$ is leaked, but secure parameter sets can be designed.
- Significant advantages over Goppa code-based systems and other GRS code-based systems are achieved.

M. Baldi, F. Chiaraluce, J. Rosenthal, P. Santini and D. Schipani, "Security of generalised Reed–Solomon code-based cryptosystems," IET Information Security, vol. 13, no. 4, pp. 404–410, 2019.

BCRSS cryptosystem - subcode attacks

- When $\mathbf{a}$ is public, Eve can consider

$$\mathbf{H}_S = \begin{bmatrix} \mathbf{H}' \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{S}^{-1} \cdot \mathbf{H} \cdot \mathbf{b}^T \cdot \mathbf{a} + \mathbf{S}^{-1} \cdot \mathbf{H} \cdot \mathbf{T}^T \\ \mathbf{a} \end{bmatrix}.$$

- For $m = 1$:
  - $\mathbf{H}_S$ defines a subcode of the public code that is permutation-equivalent to a subcode of the private code.
  - Security of the BCRSS scheme is equivalent to that of the BL scheme with same subcode parameters.
  - The dimension of the subcode is $n - \text{rank}\{\mathbf{H}_S\}$ and we can choose parameters that avoid known attacks on the subcode.
  - For $k \geq n/2$, subcode attacks are prevented if $k - z - 1 < 2k - n + 1$.

BCRSS cryptosystem - subcode attacks

- When $\mathbf{a}$ is public, Eve can consider

$$\mathbf{H}_S = \begin{bmatrix} \mathbf{H}' \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{S}^{-1} \cdot \mathbf{H} \cdot \mathbf{b}^T \cdot \mathbf{a} + \mathbf{S}^{-1} \cdot \mathbf{H} \cdot \mathbf{T}^T \\ \mathbf{a} \end{bmatrix}.$$

- For $m = 1$:
  - $\mathbf{H}_S$ defines a subcode of the public code that is permutation-equivalent to a subcode of the private code.
  - Security of the BCRSS scheme is equivalent to that of the BL scheme with same subcode parameters.
  - The dimension of the subcode is $n - \text{rank}\{\mathbf{H}_S\}$ and we can choose parameters that avoid known attacks on the subcode.
  - For $k \geq n/2$, subcode attacks are prevented if $k - z - 1 < 2k - n + 1$.

BCRSS cryptosystem - subcode attacks (2)

- For $m > 1$:
  - The subcode defined by $\mathbf{H}_S$ is no longer permutation-equivalent to a subcode of the private code.
  - Security is higher than security of BL.
  - Distinguisher attacks still apply.
  - But they can be countered by suitably choosing m and z.

BCRSS cryptosystem - decoding attacks

- Knowing $\mathbf{a}$ and γ facilitates decoding attacks.
- Since

$$\begin{bmatrix} \mathbf{x} \\ \gamma \end{bmatrix} = \mathbf{H}_S \cdot \mathbf{e}^T$$

  an attacker could perform syndrome decoding through $\mathbf{H}_S$.
- The code rate is $\frac{k-z}{n} < \frac{k}{n}$, and ISD is facilitated.
- The attack complexity decreases as long as z increases (when $z \geq k$ the attack becomes very simple).
- Parameter sets with large k and relatively small z can be chosen to thwart these attacks.

BCRSS cryptosystem - decoding attacks

- Knowing $\mathbf{a}$ and γ facilitates decoding attacks.
- Since

$$\begin{bmatrix} \mathbf{x} \\ \gamma \end{bmatrix} = \mathbf{H}_S \cdot \mathbf{e}^T$$

  an attacker could perform syndrome decoding through $\mathbf{H}_S$.
- The code rate is $\frac{k-z}{n} < \frac{k}{n}$, and ISD is facilitated.
- The attack complexity decreases as long as z increases (when $z \geq k$ the attack becomes very simple).
- Parameter sets with large k and relatively small z can be chosen to thwart these attacks.

BCRSS cryptosystem - decoding attacks

- Knowing $\mathbf{a}$ and γ facilitates decoding attacks.
- Since

$$\begin{bmatrix} \mathbf{x} \\ \gamma \end{bmatrix} = \mathbf{H}_S \cdot \mathbf{e}^T$$

  an attacker could perform syndrome decoding through $\mathbf{H}_S$.
- The code rate is $\frac{k-z}{n} < \frac{k}{n}$, and ISD is facilitated.
- The attack complexity decreases as long as z increases (when $z \geq k$ the attack becomes very simple).
- Parameter sets with large k and relatively small z can be chosen to thwart these attacks.

BCRSS cryptosystem - performance

Table: BCRSS system parameters for $m = 1.2$, $SL = 2^{180}$ and $SL = 2^{260}$.

SL	z	n	k	t	t_p	WF	KS(KiB)	R_e(Nied.)	R_e(McEl.)
180	10	796	634	81	67	181.03	130.09	0.5870	0.7866
180	50	918	740	89	74	180.19	210.43	0.4879	0.7644
260	10	1162	928	117	97	260.64	284.27	0.5894	0.7918
260	50	1276	1018	129	107	260.03	408.04	0.5128	0.7677

Table: Other algebraic code-based schemes, $SL = 2^{180}$ and $SL = 2^{260}$.

SL	Scheme	n	k	$t = t_p$	WF	KS(KiB)	R_e(Nied.)	R_e(McEl.)
180	Goppa	4096	3004	91	180.06	400.44	0.5724	0.7334
180	BL	1212	1062	75	180.05	342.15	0.3806	0.7525
260	Goppa	8192	6957	95	260.57	1048.82	0.6012	0.8492
260	BL	1788	1560	114	260.11	801.13	0.3732	0.7450

LDPC codes in the McEliece cryptosystem

- Low-density parity-check (LDPC) codes are capacity-achieving under iterative decoding.
- They allow a random-based design, resulting in large families of codes with similar characteristics and performance.
- The sparsity of their matrices is attractive for achieving compact representations (keys).
- All this makes them interesting for the use in McEliece/Niederreiter cryptosystem variants.

Alert

Public codes cannot be LDPC codes as well, otherwise secret codes are likely to be exposed.

- C. Monico, J. Rosenthal, and A. Shokrollahi, "Using low density parity check codes in the McEliece cryptosystem," 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,"

LDPC codes in the McEliece cryptosystem

- LDPC codes are capacity-achieving under iterative decoding.
- They allow a random-based design, resulting in large families of codes with similar characteristics and performance.
- The sparsity of their matrices is attractive for achieving compact representations (keys).
- All this makes them interesting for the use in McEliece/Niederreiter cryptosystem variants.

Alert

Public codes cannot be LDPC codes as well, otherwise secret codes are likely to be exposed.

- C. Monico, J. Rosenthal, and A. Shokrollahi, "Using low density parity check codes in the McEliece cryptosystem," 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.

QC-LDPC codes

- A linear block code is a quasi-cyclic (QC) code if:
  - its dimension and length are multiple of an integer p ($k = k_0 p, n = n_0 p$),
  - every cyclic shift of a codeword by n_0 positions yields another codeword.
- The generator and parity-check matrices of a QC code can assume two forms:
  - Circulant of blocks.
  - Block of circulants.

Advantage

The QC structure allows to represent the generator and parity-check matrices in a compact way (each circulant is completely described by its first row).

QC-LDPC codes

- A linear block code is a QC code if:
  - its dimension and length are multiple of an integer p ($k = k_0 p, n = n_0 p$),
  - every cyclic shift of a codeword by n_0 positions yields another codeword.
- The generator and parity-check matrices of a QC code can assume two forms:
  - Circulant of blocks.
  - Block of circulants.

Advantage

The QC structure allows to represent the generator and parity-check matrices in a compact way (each circulant is completely described by its first row).

Example of QC-(almost)LDPC code

$$\mathbf{H} = \left[\begin{array}{cccccccccc|cccccccccc}
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1
\end{array}\right]$$

- Number of ciculant blocks: $n_0 = 2$.
- Code rate: $R = \frac{n_0 - 1}{n_0} = 1/2$.
- Parity-check matrix column weight: $d_v = 3$.
- Parity-check matrix row weight: $d_c = n_0 d_v = 6$.

QC-LDPC and QC-MDPC codes

$$\mathbf{H} = \left[\begin{array}{cccccccccc|cccccccccc} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ & & & & & \vdots & & & & & & & & & & \vdots & & & & & & \end{array}\right]$$

- The parity-check matrix is described by its first row.
- The storage size increases linearly in the code length.
- The code length is usually very large ($10'000 \lessapprox n \lessapprox 100'000$)
- QC-LDPC codes usually have $d_c \approx \log n$.
- QC-LDPC codes with density larger than usual ($d_c \approx \sqrt{n}$) are also called QC-MDPC codes.

QC-LDPC and QC-MDPC codes

$$\mathbf{H} = \left[\begin{array}{cccccccccc|cccccccccc} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ & & & & & \vdots & & & & & & & & & & \vdots & & & & & & \end{array}\right]$$

- The parity-check matrix is described by its first row.
- The storage size increases linearly in the code length.
- The code length is usually very large ($10'000 \lessapprox n \lessapprox 100'000$)
- QC-LDPC codes usually have $d_c \approx \log n$.
- QC-LDPC codes with density larger than usual ($d_c \approx \sqrt{n}$) are also called QC-MDPC codes.

QC-LDPC and QC-MDPC codes

$$\mathbf{H} = \left[\begin{array}{cccccccccc|cccccccccc} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ & & & & & \vdots & & & & & & & & & & \vdots & & & & & & \end{array}\right]$$

- The parity-check matrix is described by its first row.
- The storage size increases linearly in the code length.
- The code length is usually very large ($10'000 \lessapprox n \lessapprox 100'000$)
- QC-LDPC codes usually have $d_c \approx \log n$.
- QC-LDPC codes with density larger than usual ($d_c \approx \sqrt{n}$) are also called QC-MDPC codes.

QC-LDPC and QC-MDPC codes

$$\mathbf{H} = \left[\begin{array}{cccccccccc|cccccccccc} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ & & & & & \vdots & & & & & & & & & & \vdots & & & & & & \end{array}\right]$$

- The parity-check matrix is described by its first row.
- The storage size increases linearly in the code length.
- The code length is usually very large ($10'000 \lessapprox n \lessapprox 100'000$)
- QC-LDPC codes usually have $d_c \approx \log n$.
- QC-LDPC codes with density larger than usual ($d_c \approx \sqrt{n}$) are also called QC-MDPC codes.

Bit flipping decoding

- Hard-decision iterative decoding algorithms for LDPC codes are known as bit flipping (BF) algorithms.
- They are well suited when soft information from the channel is not available (as in the McEliece cryptosystem).
- From [Gallager1962]:

> The decoder computes all the parity checks and then changes any digit that is contained in more than some fixed number of unsatisfied parity-check equations. Using these new values, the parity checks are recomputed, and the process is repeated until the parity checks are all satisfied.

▶ R. G. Gallager, "Low-density parity-check codes," IRE Trans. Inform. Theory, vol. 8, pp. 21–28, 1962.

Bit flipping decoding performance

- The decoding radius of LDPC codes under BF decoding cannot be determined analytically through closed form expressions.
- However, the average BF decoder performance can be estimated through a probabilistic model.
- It allows computing the threshold value of the number of errors for which BF converges to the right codeword in asymptotic conditions.

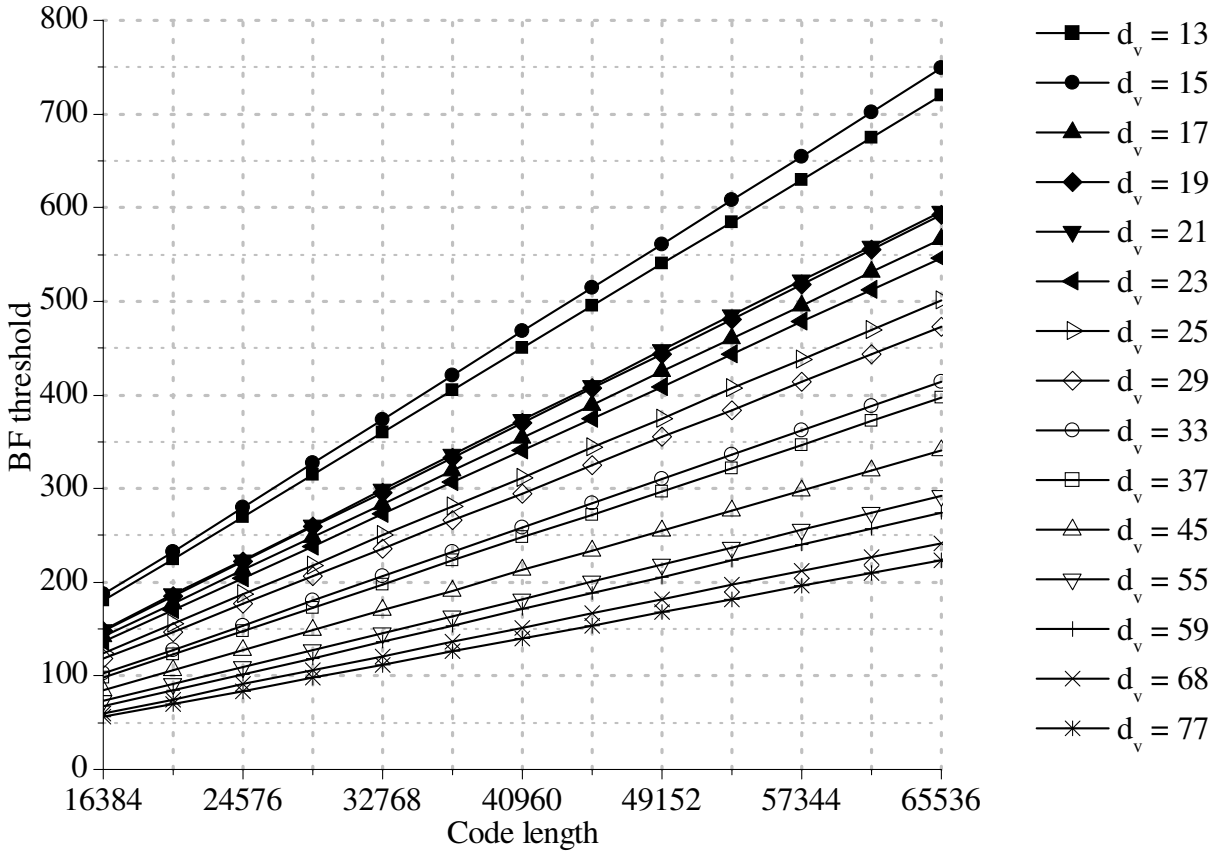
Bit flipping decoding performance

- The decoding radius of LDPC codes under BF decoding cannot be determined analytically through closed form expressions.
- However, the average BF decoder performance can be estimated through a probabilistic model.
- It allows computing the threshold value of the number of errors for which BF converges to the right codeword in asymptotic conditions.

Bit flipping decoding performance

- The decoding radius of LDPC codes under BF decoding cannot be determined analytically through closed form expressions.
- However, the average BF decoder performance can be estimated through a probabilistic model.
- It allows computing the threshold value of the number of errors for which BF converges to the right codeword in asymptotic conditions.

Bit flipping decoding performance - examples



BF decoding thresholds versus code length (n) for QC-LDPC codes with $n_0 = 4$ and several parity-check matrix column weights (d_v).

QC-LDPC code-based cryptosystems

- QC-LDPC codes bring important advantages in the framework of McEliece/Niederreiter cryptosystems:
  - The sparsity of their matrices enables very efficient decoding.
  - Quasi-cyclicity enables very compact keys.
- quasi-cyclic low-density parity-check (QC-LDPC) code-based systems cryptanalyzed for more than 10 years.
- quasi-cyclic moderate-density parity-check (QC-MDPC) code-based variants introduced more recently.

- M. Baldi, M. Bodrato, F. Chiaraluce, "A new analysis of the McEliece cryptosystem based on QC-LDPC codes", Proc. SCN 2008, vol. 5229 of LNCS, pp. 246–262, 2008.
- R. Misoczki, J.-P. Tillich, N. Sendrier, P.S.L.M. Barreto, "MDPC-McEliece: new McEliece variants from moderate density parity-check codes", Proc. IEEE ISIT 2013, pp. 2069–2073, July 2013.

QC-LDPC code-based cryptosystems

- QC-LDPC codes bring important advantages in the framework of McEliece/Niederreiter cryptosystems:
  - The sparsity of their matrices enables very efficient decoding.
  - Quasi-cyclicity enables very compact keys.
- QC-LDPC code-based systems cryptanalyzed for more than 10 years.
- QC-MDPC code-based variants introduced more recently.

- M. Baldi, M. Bodrato, F. Chiaraluce, "A new analysis of the McEliece cryptosystem based on QC-LDPC codes", Proc. SCN 2008, vol. 5229 of LNCS, pp. 246–262, 2008.
- R. Misoczki, J.-P. Tillich, N. Sendrier, P.S.L.M. Barreto, "MDPC-McEliece: new McEliece variants from moderate density parity-check codes", Proc. IEEE ISIT 2013, pp. 2069–2073, July 2013.

QC-LDPC code-based cryptosystems

- QC-LDPC codes bring important advantages in the framework of McEliece/Niederreiter cryptosystems:
  - The sparsity of their matrices enables very efficient decoding.
  - Quasi-cyclicity enables very compact keys.
- QC-LDPC code-based systems cryptanalyzed for more than 10 years.
- QC-MDPC code-based variants introduced more recently.

- M. Baldi, M. Bodrato, F. Chiaraluce, "A new analysis of the McEliece cryptosystem based on QC-LDPC codes", Proc. SCN 2008, vol. 5229 of LNCS, pp. 246–262, 2008.
- R. Misoczki, J.-P. Tillich, N. Sendrier, P.S.L.M. Barreto, "MDPC-McEliece: new McEliece variants from moderate density parity-check codes", Proc. IEEE ISIT 2013, pp. 2069–2073, July 2013.

QC-LDPC code-based cryptosystems

- QC-LDPC codes bring important advantages in the framework of McEliece/Niederreiter cryptosystems:
  - The sparsity of their matrices enables very efficient decoding.
  - Quasi-cyclicity enables very compact keys.
- QC-LDPC code-based systems cryptanalyzed for more than 10 years.
- QC-MDPC code-based variants introduced more recently.

- M. Baldi, M. Bodrato, F. Chiaraluce, "A new analysis of the McEliece cryptosystem based on QC-LDPC codes", Proc. SCN 2008, vol. 5229 of LNCS, pp. 246–262, 2008.
- R. Misoczki, J.-P. Tillich, N. Sendrier, P.S.L.M. Barreto, "MDPC-McEliece: new McEliece variants from moderate density parity-check codes", Proc. IEEE ISIT 2013, pp. 2069–2073, July 2013.

QC-LDPC code-based cryptosystems

- QC-LDPC codes bring important advantages in the framework of McEliece/Niederreiter cryptosystems:
  - The sparsity of their matrices enables very efficient decoding.
  - Quasi-cyclicity enables very compact keys.
- QC-LDPC code-based systems cryptanalyzed for more than 10 years.
- QC-MDPC code-based variants introduced more recently.

- M. Baldi, M. Bodrato, F. Chiaraluce, "A new analysis of the McEliece cryptosystem based on QC-LDPC codes", Proc. SCN 2008, vol. 5229 of LNCS, pp. 246–262, 2008.
- R. Misoczki, J.-P. Tillich, N. Sendrier, P.S.L.M. Barreto, "MDPC-McEliece: new McEliece variants from moderate density parity-check codes", Proc. IEEE ISIT 2013, pp. 2069–2073, July 2013.

QC-LDPC/MDPC codes in the NIST contest

- LEDAcrypt (Low-dEnsity parity-check coDe-bAsed cryptographic systems), providing:
  - A Niederreiter-based KEM with IND-CPA and ephemeral keys.
  - A Niederreiter-based KEM with IND-CCA2 and long-term keys.
  - A McEliece-based PKC with IND-CCA2.
- BIKE (Bit Flipping Key Encapsulation), providing:
  - Three McEliece/Niederreiter-based KEMs with IND-CPA and ephemeral keys.
  - Three McEliece/Niederreiter-based KEMs with IND-CCA2 and long-term keys.

- https://www.ledacrypt.org/
- https://bikesuite.org/

QC-LDPC/MDPC codes in the NIST contest

- LEDAcrypt (Low-dEnsity parity-check coDe-bAsed cryptographic systems), providing:
  - A Niederreiter-based KEM with IND-CPA and ephemeral keys.
  - A Niederreiter-based KEM with IND-CCA2 and long-term keys.
  - A McEliece-based PKC with IND-CCA2.
- BIKE (Bit Flipping Key Encapsulation), providing:
  - Three McEliece/Niederreiter-based KEMs with IND-CPA and ephemeral keys.
  - Three McEliece/Niederreiter-based KEMs with IND-CCA2 and long-term keys.

- https://www.ledacrypt.org/
- https://bikesuite.org/

Private code

The private code is a QC-LDPC code with:

- rate $R = \frac{n_0 - 1}{n_0}$ (with $n_0 = 2, 3, 4$),
- redundancy r (in the order of some thousands),
- length $n = n_0 \cdot r$,
- dimension $k = (n_0 - 1) \cdot r$.

Secret parity-check matrix

$$\mathbf{H} = [\mathbf{H}_0 | \mathbf{H}_1 | \dots | \mathbf{H}_{n_0 - 1}]$$

- Each $\mathbf{H}_i$ is an $r \times r$ circulant matrix.
- Prime values for r must be chosen to avoid folding attacks.

M. Koochak Shooshtari, M. Ahmadian-Attari, T. Johansson and M. Reza Aref, "Cryptanalysis of McEliece cryptosystem variants based on quasi-cyclic low-density parity check codes," in IET Information Security, vol. 10, no. 4, pp. 194-202, 2016.

Private code

The private code is a QC-LDPC code with:

- rate $R = \frac{n_0 - 1}{n_0}$ (with $n_0 = 2, 3, 4$),
- redundancy r (in the order of some thousands),
- length $n = n_0 \cdot r$,
- dimension $k = (n_0 - 1) \cdot r$.

Secret parity-check matrix

$$\mathbf{H} = [\mathbf{H}_0 | \mathbf{H}_1 | \ldots | \mathbf{H}_{n_0 - 1}]$$

- Each $\mathbf{H}_i$ is an $r \times r$ circulant matrix.
- Prime values for r must be chosen to avoid folding attacks.

M. Koochak Shooshtari, M. Ahmadian-Attari, T. Johansson and M. Reza Aref, "Cryptanalysis of McEliece cryptosystem variants based on quasi-cyclic low-density parity check codes," in IET Information Security, vol. 10, no. 4, pp. 194-202, 2016.

Private code

The private code is a QC-LDPC code with:

- rate $R = \frac{n_0 - 1}{n_0}$ (with $n_0 = 2, 3, 4$),
- redundancy r (in the order of some thousands),
- length $n = n_0 \cdot r$,
- dimension $k = (n_0 - 1) \cdot r$.

Secret parity-check matrix

$$\mathbf{H} = [\mathbf{H}_0 | \mathbf{H}_1 | \ldots | \mathbf{H}_{n_0 - 1}]$$

- Each $\mathbf{H}_i$ is an $r \times r$ circulant matrix.
- Prime values for r must be chosen to avoid folding attacks.

M. Koochak Shooshtari, M. Ahmadian-Attari, T. Johansson and M. Reza Aref, "Cryptanalysis of McEliece cryptosystem variants based on quasi-cyclic low-density parity check codes," in IET Information Security, vol. 10, no. 4, pp. 194-202, 2016.

Key generation

Private key

- Parity-check matrix $\mathbf{H} \Rightarrow$ generator matrix $\mathbf{G}$.
- A $k \times k$ scrambling matrix $\mathbf{S}$.
- An $n \times n$ transformation matrix $\mathbf{Q}$.

- $\mathbf{Q}$ is a sparse matrix with avg row/column weight $m \geq 1$.
- Both $\mathbf{S}$ and $\mathbf{Q}$ have QC form.
- When d_v is large, $\mathbf{Q}$ boils down to an $n \times n$ permutation matrix ($m = 1$), or can even be eliminated.

Public key

$$\mathbf{G}' = \mathbf{S}^{-1} \cdot \mathbf{G} \cdot \mathbf{Q}^{-1}$$

- $\mathbf{G}'$ can be systematic $\Rightarrow$ public key size $= (n_0 - 1) \cdot r$ bits.

Key generation

Private key

- Parity-check matrix $\mathbf{H} \Rightarrow$ generator matrix $\mathbf{G}$.
- A $k \times k$ scrambling matrix $\mathbf{S}$.
- An $n \times n$ transformation matrix $\mathbf{Q}$.

- $\mathbf{Q}$ is a sparse matrix with avg row/column weight $m \geq 1$.
- Both $\mathbf{S}$ and $\mathbf{Q}$ have QC form.
- When d_v is large, $\mathbf{Q}$ boils down to an $n \times n$ permutation matrix ($m = 1$), or can even be eliminated.

Public key

$$\mathbf{G}' = \mathbf{S}^{-1} \cdot \mathbf{G} \cdot \mathbf{Q}^{-1}$$

- $\mathbf{G}'$ can be systematic $\Rightarrow$ public key size $= (n_0 - 1) \cdot r$ bits.

Encryption

- Alice has to encrypt a k-bit vector $\mathbf{u}$.
- She fetches Bob's public key $\mathbf{G}'$.
- She generates a random binary intentional error vector $\mathbf{e}$ with weight t.

Encryption map

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}' \oplus \mathbf{e} = \mathbf{c}' \oplus \mathbf{e}$$

- $\oplus$ denotes modulo-2 addition.
- Hence, all intentional errors are bit flipping errors.

Decryption

1. To recover $\mathbf{u}$ from $\mathbf{x}$, Bob first inverts the secret transformation (if used):

$$\mathbf{x}' = \mathbf{x} \cdot \mathbf{Q} = \mathbf{u} \cdot \mathbf{S}^{-1} \cdot \mathbf{G} \oplus \mathbf{e} \cdot \mathbf{Q} = \mathbf{c} \oplus \mathbf{e} \cdot \mathbf{Q}.$$

2. $\mathbf{x}'$ is the codeword $\mathbf{c}$ belonging to the private code, corrupted by the error vector $\mathbf{e}' = \mathbf{e} \cdot \mathbf{Q}$.
3. $\mathbf{e}'$ is a binary vector with weight $\leq t' = tm$.
4. Bob corrects all the errors through LDPC decoding and obtains $\mathbf{u} \cdot \mathbf{S}^{-1}$.
5. Bob recovers $\mathbf{u}$ through multiplication by $\mathbf{S}$.

Main attacks

Decoding attacks

Aimed at decrypting one or more ciphertexts without knowing the private key.

Key recovery attacks

Aimed at recovering the private key from the public key.

- Both decoding and key recovery attacks rely on information set decoding (ISD) algorithms.
  - For any linear code, the rows of the parity-check matrix are codewords of its dual code.
  - For QC-LDPC codes, these rows have low weight and can be searched through ISD algorithms.
- The quantum speedup due to Grover's algorithm must be taken into account.

Main attacks

Decoding attacks

Aimed at decrypting one or more ciphertexts without knowing the private key.

Key recovery attacks

Aimed at recovering the private key from the public key.

- Both decoding and key recovery attacks rely on ISD algorithms.
  - For any linear code, the rows of the parity-check matrix are codewords of its dual code.
  - For QC-LDPC codes, these rows have low weight and can be searched through ISD algorithms.
- The quantum speedup due to Grover's algorithm must be taken into account.

Main attacks

Decoding attacks

Aimed at decrypting one or more ciphertexts without knowing the private key.

Key recovery attacks

Aimed at recovering the private key from the public key.

- Both decoding and key recovery attacks rely on ISD algorithms.
  - For any linear code, the rows of the parity-check matrix are codewords of its dual code.
  - For QC-LDPC codes, these rows have low weight and can be searched through ISD algorithms.
- The quantum speedup due to Grover's algorithm must be taken into account.

Decoding attacks

- The most dangerous decoding attacks exploit ISD algorithms.
- For code length n, dimension k and weight searched w, $WF_{\mathrm{BJMM}}(n, k, w)$ denotes the ISD work factor according to [Becker2012].
- The QC nature of these codes yields a speedup factor in the order of $\sqrt{r}$.
- ISD-based decoding attack work factor:

$$WF_{\mathrm{DA}}(t) = \frac{1}{\sqrt{r}} WF_{\mathrm{BJMM}}(n, k, t)$$

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 *Advances in Cryptology - EUROCRYPT 2012*, vol. 7237 of Springer LNCS, pp. 520–536, 2012.

Decoding attacks

- The most dangerous decoding attacks exploit ISD algorithms.
- For code length n, dimension k and weight searched w, $WF_{\mathrm{BJMM}}(n, k, w)$ denotes the ISD work factor according to [Becker2012].
- The QC nature of these codes yields a speedup factor in the order of $\sqrt{r}$.
- ISD-based decoding attack work factor:

$$WF_{\mathrm{DA}}(t) = \frac{1}{\sqrt{r}} WF_{\mathrm{BJMM}}(n, k, t)$$

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 *Advances in Cryptology - EUROCRYPT 2012*, vol. 7237 of Springer LNCS, pp. 520–536, 2012.

Decoding attacks

- The most dangerous decoding attacks exploit ISD algorithms.
- For code length n, dimension k and weight searched w, $WF_{\mathrm{BJMM}}(n, k, w)$ denotes the ISD work factor according to [Becker2012].
- The QC nature of these codes yields a speedup factor in the order of $\sqrt{r}$.
- ISD-based decoding attack work factor:

$$WF_{\mathrm{DA}}(t) = \frac{1}{\sqrt{r}} WF_{\mathrm{BJMM}}(n, k, t)$$

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 *Advances in Cryptology - EUROCRYPT 2012*, vol. 7237 of Springer LNCS, pp. 520–536, 2012.

Decoding attacks

- The most dangerous decoding attacks exploit ISD algorithms.
- For code length n, dimension k and weight searched w, $WF_{\mathrm{BJMM}}(n, k, w)$ denotes the ISD work factor according to [Becker2012].
- The QC nature of these codes yields a speedup factor in the order of $\sqrt{r}$.
- ISD-based decoding attack work factor:

$$WF_{\mathrm{DA}}(t) = \frac{1}{\sqrt{r}} WF_{\mathrm{BJMM}}(n, k, t)$$

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 *Advances in Cryptology - EUROCRYPT 2012*, vol. 7237 of Springer LNCS, pp. 520–536, 2012.

Key recovery attacks

- Aimed at recovering the private key from the public key.
- May be successful even by finding an alternative private key which allows decoding.
- The public code admits a parity-check matrix $\mathbf{H}' = \mathbf{H} \cdot \mathbf{Q}^T$ that is moderately sparse.
- Its rows have weight $d_c' = m \cdot n_0 \cdot d_v$ and can be searched as codewords in the dual of the public code.
- Then $\mathbf{H}'$ can be used to recover $\mathbf{H}$ or to perform decoding.
- The QC nature of the codes gives a speedup factor in the order of r.

$$WF_{\mathrm{KRA}} = \frac{1}{r} WF_{\mathrm{BJMM}}(n, r, d_c')$$

Key recovery attacks

- Aimed at recovering the private key from the public key.
- May be successful even by finding an alternative private key which allows decoding.
- The public code admits a parity-check matrix $\mathbf{H}' = \mathbf{H} \cdot \mathbf{Q}^T$ that is moderately sparse.
- Its rows have weight $d_c' = m \cdot n_0 \cdot d_v$ and can be searched as codewords in the dual of the public code.
- Then $\mathbf{H}'$ can be used to recover $\mathbf{H}$ or to perform decoding.
- The QC nature of the codes gives a speedup factor in the order of r.

$$WF_{\mathrm{KRA}} = \frac{1}{r} WF_{\mathrm{BJMM}}(n, r, d_c')$$

Key recovery attacks

- Aimed at recovering the private key from the public key.
- May be successful even by finding an alternative private key which allows decoding.
- The public code admits a parity-check matrix $\mathbf{H}' = \mathbf{H} \cdot \mathbf{Q}^T$ that is moderately sparse.
- Its rows have weight $d_c' = m \cdot n_0 \cdot d_v$ and can be searched as codewords in the dual of the public code.
- Then $\mathbf{H}'$ can be used to recover $\mathbf{H}$ or to perform decoding.
- The QC nature of the codes gives a speedup factor in the order of r.

$$WF_{\mathrm{KRA}} = \frac{1}{r} WF_{\mathrm{BJMM}}(n, r, d_c')$$

Key recovery attacks

- Aimed at recovering the private key from the public key.
- May be successful even by finding an alternative private key which allows decoding.
- The public code admits a parity-check matrix $\mathbf{H}' = \mathbf{H} \cdot \mathbf{Q}^T$ that is moderately sparse.
- Its rows have weight $d_c' = m \cdot n_0 \cdot d_v$ and can be searched as codewords in the dual of the public code.
- Then $\mathbf{H}'$ can be used to recover $\mathbf{H}$ or to perform decoding.
- The QC nature of the codes gives a speedup factor in the order of r.

$$WF_{\mathrm{KRA}} = \frac{1}{r} WF_{\mathrm{BJMM}}(n, r, d_c')$$

Key recovery attacks

- Aimed at recovering the private key from the public key.
- May be successful even by finding an alternative private key which allows decoding.
- The public code admits a parity-check matrix $\mathbf{H}' = \mathbf{H} \cdot \mathbf{Q}^T$ that is moderately sparse.
- Its rows have weight $d'_c = m \cdot n_0 \cdot d_v$ and can be searched as codewords in the dual of the public code.
- Then $\mathbf{H}'$ can be used to recover $\mathbf{H}$ or to perform decoding.
- The QC nature of the codes gives a speedup factor in the order of r.

$$WF_{\mathrm{KRA}} = \frac{1}{r} WF_{\mathrm{BJMM}}(n, r, d'_c)$$

System examples

Niederreiter cryptosystems (with CCA2 secure conversion) with 128-bit security.

Code	n	k	t_p	t	m	d_v	KS [KiB]
Goppa	2960	2288	56	56	-	-	188
QC-MDPC	19714	9857	134	134	1	71	1.2
QC-LDPC	24702	12351	134	501	3.74	19	1.5

- D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-Quantum Cryptography*, vol. 5299 of Springer LNCS, pp. 31–46, 2008.
- R. Misoczki, J. P. Tillich, N. Sendrier and P. S. L. M. Barreto, "MDPC-McEliece: New McEliece variants from Moderate Density Parity-Check codes," Proc. IEEE ISIT 2013, Istanbul, Turkey, pp. 2069–2073.
- M. Baldi, QC-LDPC Code-Based Cryptography, SpringerBriefs in Electrical and Computer Engineering, Springer, 2014.

Reaction attacks

- QC-LDPC and QC-MDPC codes are decoded through iterative algorithms derived from Gallager's Bit Flipping.
- Their decoding radius is not deterministic.

Warning

With iterative decoding, the decryption failure rate (DFR) is non-zero, and hard to predict.

- The residual decoding failure rate (DFR) may be exploited by an attacker to mount attacks based on Bob's reactions.

- Q. Guo, T. Johansson, and P. Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016*, vol. 10031 of *LNCS*, pages 789–815. Springer Berlin Heidelberg, 2016.
- T. Fabšič, V. Hromada, P. Stankovski, P. Zajac, Q. Guo, and T. Johansson. A reaction attack on the QC-LDPC McEliece cryptosystem. In T. Lange and T. Takagi, editors, *Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017*, pages 51–68. Springer, Utrecht, The Netherlands, June 2017.
- T. Fabšič, V. Hromada, and P. Zajac. A reaction attack on LEDApkc. *IACR Cryptology ePrint Archive*, 2018:140, 2018.

Reaction attacks

- QC-LDPC and QC-MDPC codes are decoded through iterative algorithms derived from Gallager's Bit Flipping.
- Their decoding radius is not deterministic.

Warning

With iterative decoding, the decryption failure rate (DFR) is non-zero, and hard to predict.

- The residual DFR may be exploited by an attacker to mount attacks based on Bob's reactions.

- Q. Guo, T. Johansson, and P. Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016*, vol. 10031 of *LNCS*, pages 789–815. Springer Berlin Heidelberg, 2016.
- T. Fabšič, V. Hromada, P. Stankovski, P. Zajac, Q. Guo, and T. Johansson. A reaction attack on the QC-LDPC McEliece cryptosystem. In T. Lange and T. Takagi, editors, *Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017*, pages 51–68. Springer, Utrecht, The Netherlands, June 2017.
- T. Fabšič, V. Hromada, and P. Zajac. A reaction attack on LEDApkc. *IACR Cryptology ePrint Archive*, 2018:140, 2018.

Reaction attacks

- QC-LDPC and QC-MDPC codes are decoded through iterative algorithms derived from Gallager's Bit Flipping.
- Their decoding radius is not deterministic.

Warning

With iterative decoding, the decryption failure rate (DFR) is non-zero, and hard to predict.

- The residual DFR may be exploited by an attacker to mount attacks based on Bob's reactions.

- Q. Guo, T. Johansson, and P. Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016*, vol. 10031 of *LNCS*, pages 789–815. Springer Berlin Heidelberg, 2016.
- T. Fabšič, V. Hromada, P. Stankovski, P. Zajac, Q. Guo, and T. Johansson. A reaction attack on the QC-LDPC McEliece cryptosystem. In T. Lange and T. Takagi, editors, *Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017*, pages 51–68. Springer, Utrecht, The Netherlands, June 2017.
- T. Fabšič, V. Hromada, and P. Zajac. A reaction attack on LEDApkc. *IACR Cryptology ePrint Archive*, 2018:140, 2018.

Reaction attacks (2)

- The attack is built upon two observations:
  1. The decoding failure probability is > 0 and depends on the structure of the secret key.
  2. Eve can estimate such a probability by observing Bob's reactions.
- This weakness affects both the plain versions (with only CPA security) and their conversions achieving CCA2 security.

- Q. Guo, T. Johansson, and P. Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016*, vol. 10031 of *LNCS*, pages 789–815. Springer Berlin Heidelberg, 2016.
- T. Fabšič, V. Hromada, P. Stankovski, P. Zajac, Q. Guo, and T. Johansson. A reaction attack on the QC-LDPC McEliece cryptosystem. In T. Lange and T. Takagi, editors, *Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017*, pages 51–68. Springer, Utrecht, The Netherlands, June 2017.
- T. Fabšič, V. Hromada, and P. Zajac. A reaction attack on LEDApkc. *IACR Cryptology ePrint Archive*, 2018:140, 2018.

Reaction attacks (2)

- The attack is built upon two observations:
  1. The decoding failure probability is > 0 and depends on the structure of the secret key.
  2. Eve can estimate such a probability by observing Bob's reactions.
- This weakness affects both the plain versions (with only CPA security) and their conversions achieving CCA2 security.

- Q. Guo, T. Johansson, and P. Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016*, vol. 10031 of *LNCS*, pages 789–815. Springer Berlin Heidelberg, 2016.
- T. Fabšič, V. Hromada, P. Stankovski, P. Zajac, Q. Guo, and T. Johansson. A reaction attack on the QC-LDPC McEliece cryptosystem. In T. Lange and T. Takagi, editors, *Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017*, pages 51–68. Springer, Utrecht, The Netherlands, June 2017.
- T. Fabšič, V. Hromada, and P. Zajac. A reaction attack on LEDApkc. *IACR Cryptology ePrint Archive*, 2018:140, 2018.

Reaction attacks (3)

- These attacks aim at recovering the distance spectrum of the secret matrix.
- The distance spectrum is the set of all distances d_i existing between any two ones in a binary vector (cyclically closed), together with their multiplicities $m(d_i)$.
- In the CPA case, for each d_i Eve performs many encryptions with suitably chosen error vectors and observe Bob's reactions during decryption.
- In the CCA2 case, the error vectors cannot be chosen by Eve, who must exploit those resulting from encryption of each message to make deductions about $m(d_i)$.

Reaction attacks (3)

- These attacks aim at recovering the distance spectrum of the secret matrix.
- The distance spectrum is the set of all distances d_i existing between any two ones in a binary vector (cyclically closed), together with their multiplicities $m(d_i)$.
- In the CPA case, for each d_i Eve performs many encryptions with suitably chosen error vectors and observe Bob's reactions during decryption.
- In the CCA2 case, the error vectors cannot be chosen by Eve, who must exploit those resulting from encryption of each message to make deductions about $m(d_i)$.

Reaction attacks (3)

- These attacks aim at recovering the distance spectrum of the secret matrix.
- The distance spectrum is the set of all distances d_i existing between any two ones in a binary vector (cyclically closed), together with their multiplicities $m(d_i)$.
- In the CPA case, for each d_i Eve performs many encryptions with suitably chosen error vectors and observe Bob's reactions during decryption.
- In the CCA2 case, the error vectors cannot be chosen by Eve, who must exploit those resulting from encryption of each message to make deductions about $m(d_i)$.

Reaction attacks (4)

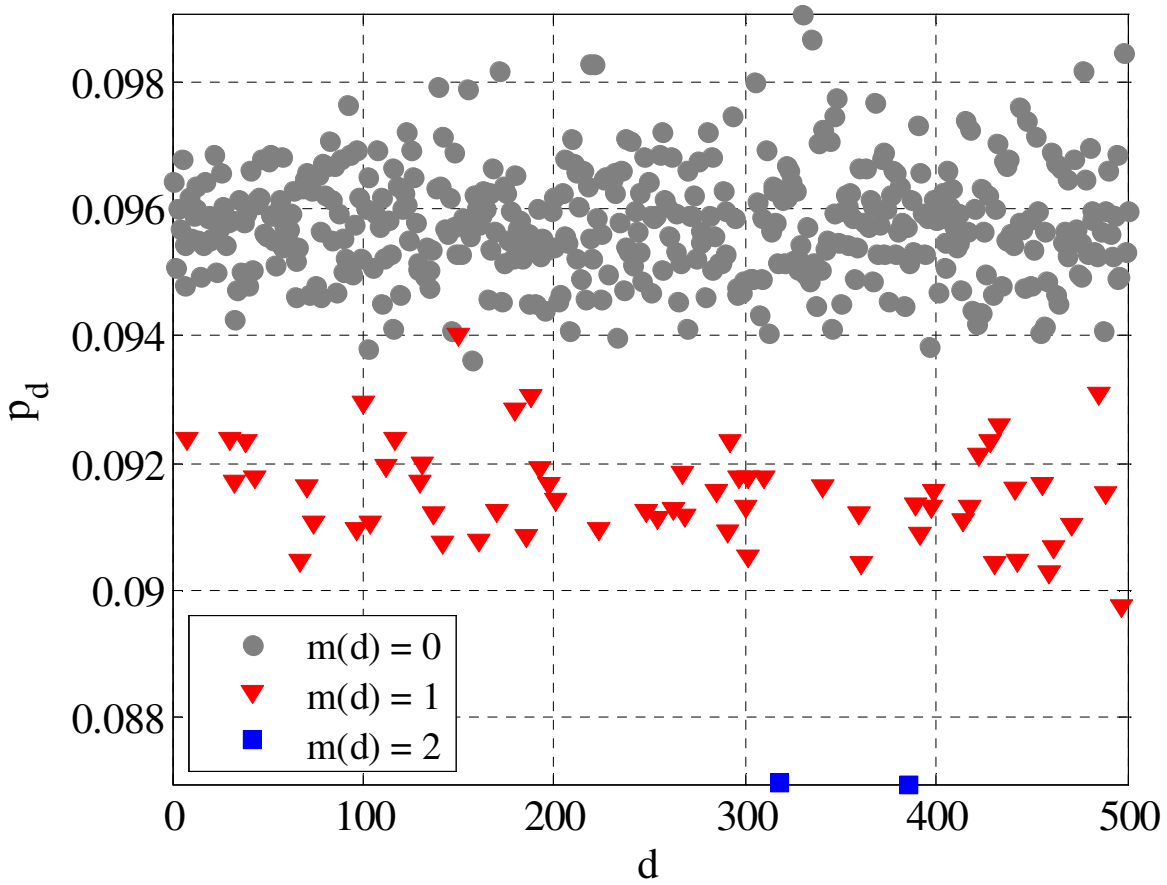


Figure: DFR conditioned on the presence of distance d in the error vector (p_d) for a system (with IND-CCA2) based on QC-MDPC codes with $n = 2000$ and $R = 1/2$.

Reaction and timing attacks

- The probabilistic nature of iterative decoding algorithms entails some information leakage.
- A general family of statistical attacks has been devised, including reaction and timing attacks.
- They exploit side-channel information (like decoding time, power consumption and DFR) to recover the cardinality of the support of sums of columns of H.
- They not only work against QC codes, but can be generalized to broader classes of codes.
- To counter these attacks we need:
  1. a constant-time and constant-power decoder implementation,
  2. a negligible DFR.

P. Santini, M. Battaglioni, F. Chiaraluce, and M. Baldi, "Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes," in Code-Based Cryptography Workshop (CBC 2019), ser. Lecture Notes in Computer Science, Springer, Cham, 2019, vol. 11666, pp. 115–136.

Reaction and timing attacks

- The probabilistic nature of iterative decoding algorithms entails some information leakage.
- A general family of statistical attacks has been devised, including reaction and timing attacks.
- They exploit side-channel information (like decoding time, power consumption and DFR) to recover the cardinality of the support of sums of columns of H.
- They not only work against QC codes, but can be generalized to broader classes of codes.
- To counter these attacks we need:
  1. a constant-time and constant-power decoder implementation,
  2. a negligible DFR.

P. Santini, M. Battaglioni, F. Chiaraluce, and M. Baldi, "Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes," in Code-Based Cryptography Workshop (CBC 2019), ser. Lecture Notes in Computer Science, Springer, Cham, 2019, vol. 11666, pp. 115–136.

Reaction and timing attacks

- The probabilistic nature of iterative decoding algorithms entails some information leakage.
- A general family of statistical attacks has been devised, including reaction and timing attacks.
- They exploit side-channel information (like decoding time, power consumption and DFR) to recover the cardinality of the support of sums of columns of H.
- They not only work against QC codes, but can be generalized to broader classes of codes.
- To counter these attacks we need:
  1. a constant-time and constant-power decoder implementation,
  2. a negligible DFR.

P. Santini, M. Battaglioni, F. Chiaraluce, and M. Baldi, "Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes," in Code-Based Cryptography Workshop (CBC 2019), ser. Lecture Notes in Computer Science, Springer, Cham, 2019, vol. 11666, pp. 115–136.

Reaction and timing attacks

- The probabilistic nature of iterative decoding algorithms entails some information leakage.
- A general family of statistical attacks has been devised, including reaction and timing attacks.
- They exploit side-channel information (like decoding time, power consumption and DFR) to recover the cardinality of the support of sums of columns of H.
- They not only work against QC codes, but can be generalized to broader classes of codes.
- To counter these attacks we need:
  1. a constant-time and constant-power decoder implementation,
  2. a negligible DFR.

P. Santini, M. Battaglioni, F. Chiaraluce, and M. Baldi, "Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes," in Code-Based Cryptography Workshop (CBC 2019), ser. Lecture Notes in Computer Science, Springer, Cham, 2019, vol. 11666, pp. 115–136.

Reaction and timing attacks

- The probabilistic nature of iterative decoding algorithms entails some information leakage.
- A general family of statistical attacks has been devised, including reaction and timing attacks.
- They exploit side-channel information (like decoding time, power consumption and DFR) to recover the cardinality of the support of sums of columns of H.
- They not only work against QC codes, but can be generalized to broader classes of codes.
- To counter these attacks we need:
  1. a constant-time and constant-power decoder implementation,
  2. a negligible DFR.

▶ P. Santini, M. Battaglioni, F. Chiaraluce, and M. Baldi, "Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes," in Code-Based Cryptography Workshop (CBC 2019), ser. Lecture Notes in Computer Science, Springer, Cham, 2019, vol. 11666, pp. 115–136.

The DFR problem

Issue 1

Iterative decoders are algorithmic $\Rightarrow$ no closed form formula for their error correction capability.

Issue 2

Mathematical models of iterative decoding algorithms work under some idealistic assumptions (e.g., i.i.d. variables).

Issue 3

Performance curves may be simulated (Monte Carlo) down to DFR $\approx 10^{-9}$.

The DFR problem

Issue 1

Iterative decoders are algorithmic $\Rightarrow$ no closed form formula for their error correction capability.

Issue 2

Mathematical models of iterative decoding algorithms work under some idealistic assumptions (e.g., i.i.d. variables).

Issue 3

Performance curves may be simulated (Monte Carlo) down to DFR $\approx 10^{-9}$.

The DFR problem

Issue 1

Iterative decoders are algorithmic $\Rightarrow$ no closed form formula for their error correction capability.

Issue 2

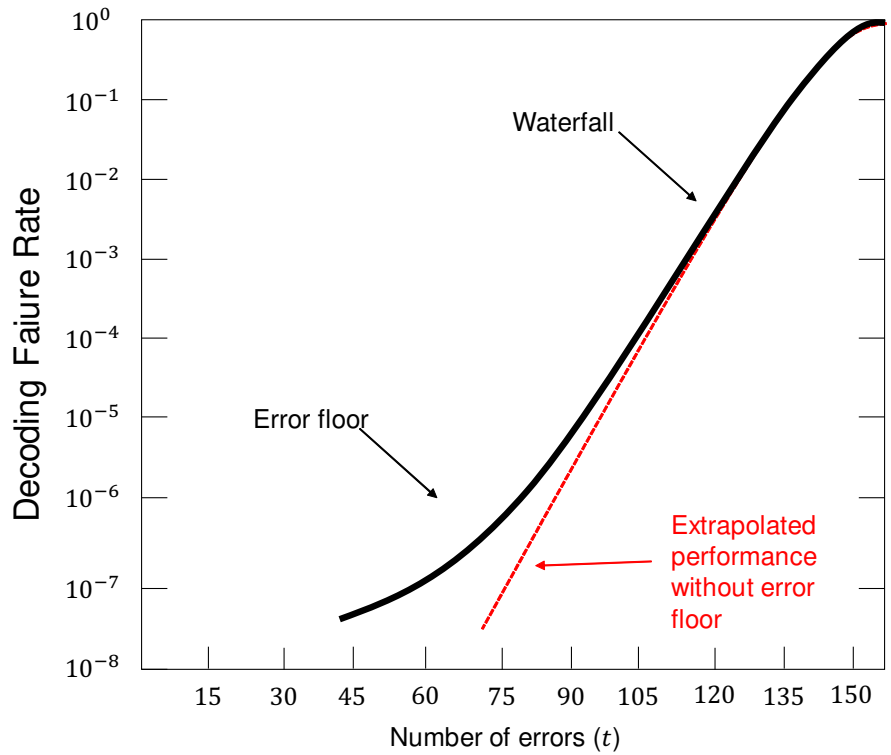
Mathematical models of iterative decoding algorithms work under some idealistic assumptions (e.g., i.i.d. variables).

Issue 3

Performance curves may be simulated (Monte Carlo) down to DFR $\approx 10^{-9}$.

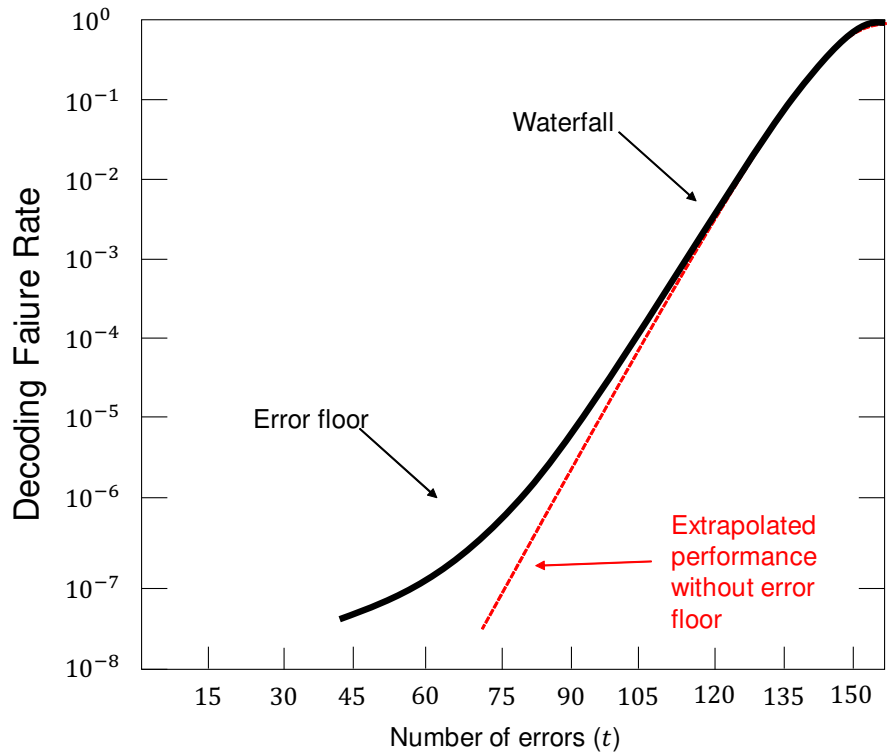
DFR extrapolation?

- One way to extend DFR curves is by extrapolation.
- But their slope may change (error floor) due to:
  - code structural properties,
  - code representation properties,
  - decoding algorithm properties.
- Hence, performance curves can hardly be extrapolated outside the simulated region.



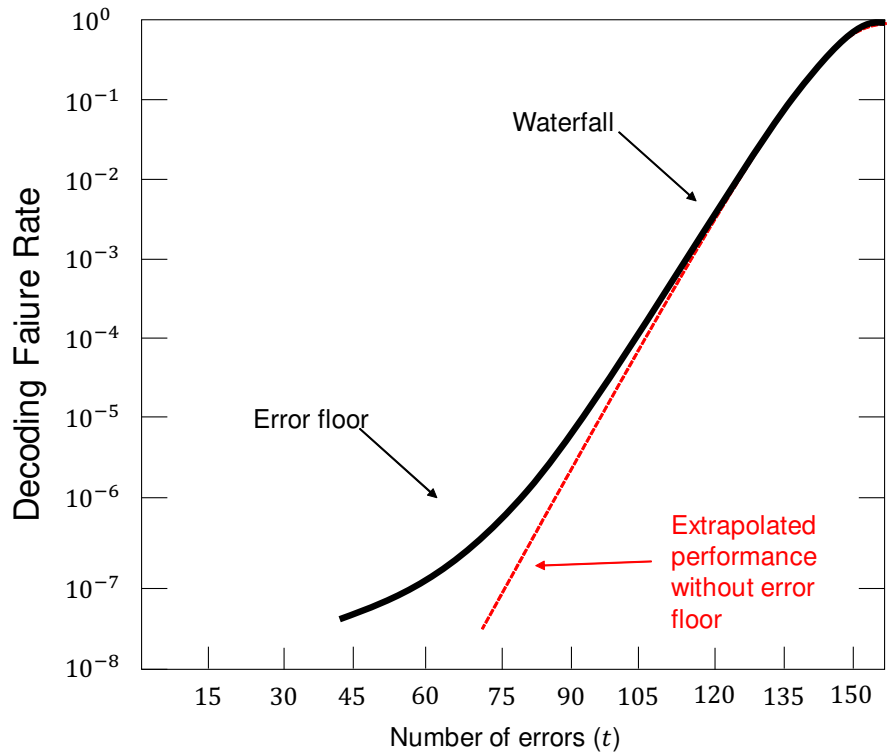
DFR extrapolation?

- One way to extend DFR curves is by extrapolation.
- But their slope may change (error floor) due to:
  - code structural properties,
  - code representation properties,
  - decoding algorithm properties.
- Hence, performance curves can hardly be extrapolated outside the simulated region.



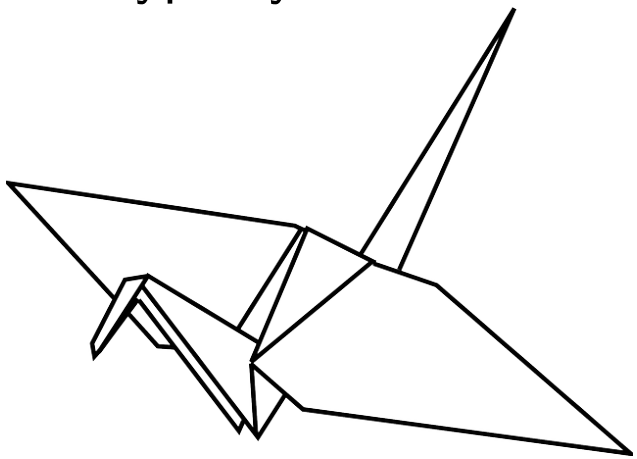
DFR extrapolation?

- One way to extend DFR curves is by extrapolation.
- But their slope may change (error floor) due to:
  - code structural properties,
  - code representation properties,
  - decoding algorithm properties.
- Hence, performance curves can hardly be extrapolated outside the simulated region.



LEDAcrypt

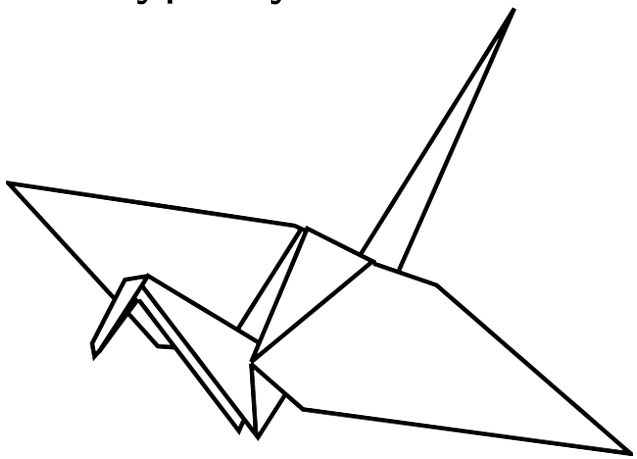
- Suite of low-density parity-check code-based cryptosystems.
- Among the 26 second round candidates of the NIST pqcrypto competition.



- Proposing team:
  - Marco Baldi (Univpm, Italy)
  - Alessandro Barenghi (Polimi, Italy)
  - Franco Chiaraluce (Univpm, Italy)
  - Gerardo Pelosi (Polimi, Italy)
  - Paolo Santini (Univpm, Italy)
- Official website (`https://www.ledacrypt.org/`):
  - First and second round specifications.
  - Full ANSI-C99 codebase.
  - Upcoming updates.
- Upcoming hardware implementation.

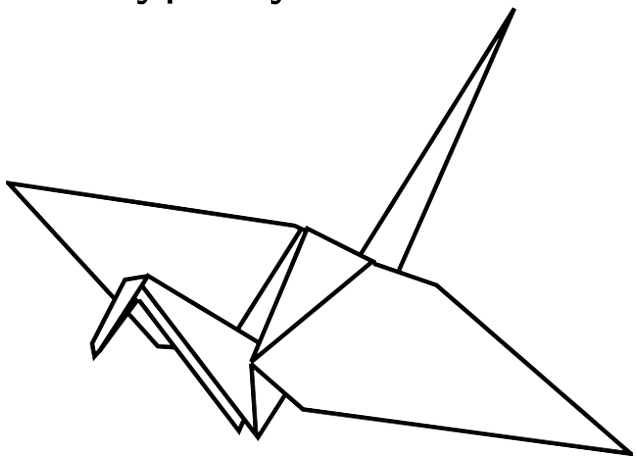
LEDAcrypt

- Suite of low-density parity-check code-based cryptosystems.
- Among the 26 second round candidates of the NIST pqcrypto competition.
- Proposing team:
  - Marco Baldi (Univpm, Italy)
  - Alessandro Barenghi (Polimi, Italy)
  - Franco Chiaraluce (Univpm, Italy)
  - Gerardo Pelosi (Polimi, Italy)
  - Paolo Santini (Univpm, Italy)
- Official website (`https://www.ledacrypt.org/`):
  - First and second round specifications.
  - Full ANSI-C99 codebase.
  - Upcoming updates.
- Upcoming hardware implementation.



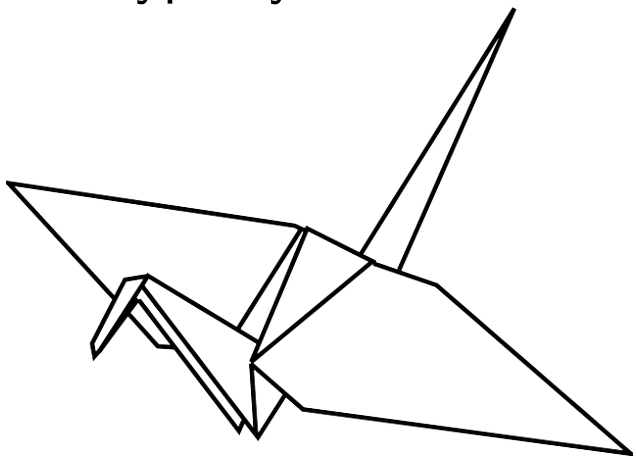
LEDAcrypt

- Suite of low-density parity-check code-based cryptosystems.
- Among the 26 second round candidates of the NIST pqcrypto competition.
- Proposing team:
  - Marco Baldi (Univpm, Italy)
  - Alessandro Barenghi (Polimi, Italy)
  - Franco Chiaraluce (Univpm, Italy)
  - Gerardo Pelosi (Polimi, Italy)
  - Paolo Santini (Univpm, Italy)
- Official website (`https://www.ledacrypt.org/`):
  - First and second round specifications.
  - Full ANSI-C99 codebase.
  - Upcoming updates.
- Upcoming hardware implementation.



LEDAcrypt

- Suite of low-density parity-check code-based cryptosystems.
- Among the 26 second round candidates of the NIST pqcrypto competition.
- Proposing team:
  - Marco Baldi (Univpm, Italy)
  - Alessandro Barenghi (Polimi, Italy)
  - Franco Chiaraluce (Univpm, Italy)
  - Gerardo Pelosi (Polimi, Italy)
  - Paolo Santini (Univpm, Italy)
- Official website (`https://www.ledacrypt.org/`):
  - First and second round specifications.
  - Full ANSI-C99 codebase.
  - Upcoming updates.
- Upcoming hardware implementation.



LEDAcrypt features

1. Both KEM and PKC modes.
2. Closed-form upper bound on the DFR.
3. Algorithmic approach to the design of parameter sets.
4. Instances with:
   - Ephemeral keys and a DFR in the order of 10^{-9}, or
   - Long-term keys and a DFR of 2^{-64} or smaller than $2^{-\lambda}$, with $\lambda = 128, 192, 256$.

- M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, P. Santini. LEDAcrypt: QC-LDPC Code-Based Cryptosystems with Bounded Decryption Failure Rate. *Code-Based Cryptography Workshop, CBC 2019*, pages 11–43. Springer LNCS, vol. 11666, 2019.

LEDAcrypt features

1. Both KEM and PKC modes.
2. Closed-form upper bound on the DFR.
3. Algorithmic approach to the design of parameter sets.
4. Instances with:
   - Ephemeral keys and a DFR in the order of 10^{-9}, or
   - Long-term keys and a DFR of 2^{-64} or smaller than $2^{-\lambda}$, with $\lambda = 128, 192, 256$.

▶ M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, P. Santini. LEDAcrypt: QC-LDPC Code-Based Cryptosystems with Bounded Decryption Failure Rate. *Code-Based Cryptography Workshop, CBC 2019*, pages 11–43. Springer LNCS, vol. 11666, 2019.

LEDAcrypt features

1. Both KEM and PKC modes.
2. Closed-form upper bound on the DFR.
3. Algorithmic approach to the design of parameter sets.
4. Instances with:
   - Ephemeral keys and a DFR in the order of 10^{-9}, or
   - Long-term keys and a DFR of 2^{-64} or smaller than $2^{-\lambda}$, with $\lambda = 128, 192, 256$.

▶ M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, P. Santini. LEDAcrypt: QC-LDPC Code-Based Cryptosystems with Bounded Decryption Failure Rate. *Code-Based Cryptography Workshop, CBC 2019*, pages 11–43. Springer LNCS, vol. 11666, 2019.

LEDAcrypt features

1. Both KEM and PKC modes.
2. Closed-form upper bound on the DFR.
3. Algorithmic approach to the design of parameter sets.
4. Instances with:
   - Ephemeral keys and a DFR in the order of 10^{-9}, or
   - Long-term keys and a DFR of 2^{-64} or smaller than $2^{-\lambda}$, with $\lambda = 128, 192, 256$.

▶ M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, P. Santini. LEDAcrypt: QC-LDPC Code-Based Cryptosystems with Bounded Decryption Failure Rate. *Code-Based Cryptography Workshop, CBC 2019*, pages 11–43. Springer LNCS, vol. 11666, 2019.

Security level goals

Target set by NIST (for categories 1, 3, and 5)

Computational effort required on either a classical or a quantum computer to break the AES with a key size of λ bits, $\lambda \in \{128, 192, 256\}$, through an exhaustive key search.

- We ignore categories 2 and 4 (recommending parameters for categories 3 and 5, respectively).
- On a classical computer we have complexity $2^{\lambda} C_{\mathrm{AES}}$, where C_{AES} is the amount of binary operations required to compute AES on a small set of plaintexts.
- C_{AES} is estimated considering the most performing AES implementations.

▶ R. Ueno, S. Morioka, N. Homma, and T. Aoki. A High Throughput/Gate AES Hardware Architecture by Compressing Encryption and Decryption Datapaths - Toward Efficient CBC-Mode Implementation. In B. Gierlichs and A. Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016*, vol. 9813 of LNCS, pages 538–558. Springer, 2016.

Security level goals

Target set by NIST (for categories 1, 3, and 5)

Computational effort required on either a classical or a quantum computer to break the AES with a key size of λ bits, $\lambda \in \{128, 192, 256\}$, through an exhaustive key search.

- We ignore categories 2 and 4 (recommending parameters for categories 3 and 5, respectively).
- On a classical computer we have complexity $2^{\lambda}\mathrm{C}_{\mathtt{AES}}$, where $\mathrm{C}_{\mathtt{AES}}$ is the amount of binary operations required to compute AES on a small set of plaintexts.
- $\mathrm{C}_{\mathtt{AES}}$ is estimated considering the most performing AES implementations.

▶ R. Ueno, S. Morioka, N. Homma, and T. Aoki. A High Throughput/Gate AES Hardware Architecture by Compressing Encryption and Decryption Datapaths - Toward Efficient CBC-Mode Implementation. In B. Gierlichs and A. Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016*, vol. 9813 of LNCS, pages 538–558. Springer, 2016.

Security level goals (2)

- Reference to the AES breaker implementation by Grassl et al.
- Grover's algorithm seeking the zeros of the function given by the binary comparison of a set of AES ciphertexts with the encryption of their corresponding plaintexts for all the possible key values.
- Only the strictly needed Clifford and T gates are counted (since they are the most expensive).

NIST Category	AES Key Size (bits)	Classical Cost (binary operations)	Quantum Cost (quantum gates)
1	128	$2^{128} \cdot 2^{14} \cdot 3 = 2^{143.5}$	$1.16 \cdot 2^{81}$
3	192	$2^{192} \cdot 2^{14} \cdot 4 = 2^{208}$	$1.33 \cdot 2^{113}$
5	256	$2^{256} \cdot 2^{14} \cdot 5 = 2^{272.3}$	$1.57 \cdot 2^{145}$

M. Grassl, B. Langenberg, M. Roetteler, and R. Steinwandt. Applying Grover's Algorithm to AES: Quantum Resource Estimates. In T. Takagi, editor, *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016*, vol. 9606 of *LNCS*, pages 29–43. Springer, 2016.

Security level goals (2)

- Reference to the AES breaker implementation by Grassl et al.
- Grover's algorithm seeking the zeros of the function given by the binary comparison of a set of AES ciphertexts with the encryption of their corresponding plaintexts for all the possible key values.
- Only the strictly needed Clifford and T gates are counted (since they are the most expensive).

NIST Category	AES Key Size (bits)	Classical Cost (binary operations)	Quantum Cost (quantum gates)
1	128	$2^{128} \cdot 2^{14} \cdot 3 = 2^{143.5}$	$1.16 \cdot 2^{81}$
3	192	$2^{192} \cdot 2^{14} \cdot 4 = 2^{208}$	$1.33 \cdot 2^{113}$
5	256	$2^{256} \cdot 2^{14} \cdot 5 = 2^{272.3}$	$1.57 \cdot 2^{145}$

M. Grassl, B. Langenberg, M. Roetteler, and R. Steinwandt. Applying Grover's Algorithm to AES: Quantum Resource Estimates. In T. Takagi, editor, *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016*, vol. 9606 of *LNCS*, pages 29–43. Springer, 2016.

Evaluated attacks

- In the automated design procedure, we consider:
  1. Attacks based on exhaustive key search (H and Q)
  2. Message recovery attacks based on ISD
  3. Key recovery attacks based on ISD
- As ISD algorithms on classical computers, we consider (finite regime complexity for) Prange, Lee-Brickell, Leon, Stern, Finiasz-Sendrier, and Becker-Joux-May-Meurer.
- On quantum computers, we consider (finite regime complexity for) Lee-Brickell and Stern.
- Open source software for estimating the complexity of attacks and automatically designing LEDAcrypt parameters available at

`https://github.com/LEDAcrypt/LEDAtools`

Evaluated attacks

- In the automated design procedure, we consider:
  1. Attacks based on exhaustive key search (H and Q)
  2. Message recovery attacks based on ISD
  3. Key recovery attacks based on ISD
- As ISD algorithms on classical computers, we consider (finite regime complexity for) Prange, Lee-Brickell, Leon, Stern, Finiasz-Sendrier, and Becker-Joux-May-Meurer.
- On quantum computers, we consider (finite regime complexity for) Lee-Brickell and Stern.
- Open source software for estimating the complexity of attacks and automatically designing LEDAcrypt parameters available at

`https://github.com/LEDAcrypt/LEDAtools`

Evaluated attacks

- In the automated design procedure, we consider:
  1. Attacks based on exhaustive key search (H and Q)
  2. Message recovery attacks based on ISD
  3. Key recovery attacks based on ISD
- As ISD algorithms on classical computers, we consider (finite regime complexity for) Prange, Lee-Brickell, Leon, Stern, Finiasz-Sendrier, and Becker-Joux-May-Meurer.
- On quantum computers, we consider (finite regime complexity for) Lee-Brickell and Stern.
- Open source software for estimating the complexity of attacks and automatically designing LEDAcrypt parameters available at

`https://github.com/LEDAcrypt/LEDAtools`

Evaluated attacks

- In the automated design procedure, we consider:
  1. Attacks based on exhaustive key search (H and Q)
  2. Message recovery attacks based on ISD
  3. Key recovery attacks based on ISD
- As ISD algorithms on classical computers, we consider (finite regime complexity for) Prange, Lee-Brickell, Leon, Stern, Finiasz-Sendrier, and Becker-Joux-May-Meurer.
- On quantum computers, we consider (finite regime complexity for) Lee-Brickell and Stern.
- Open source software for estimating the complexity of attacks and automatically designing LEDAcrypt parameters available at

`https://github.com/LEDAcrypt/LEDAtools`

LEDAcrypt automated design procedure

Input:

- Desired security level
  - λ_c: base-2 logarithm of the desired attack work factor on a classical computer
  - λ_q: base-2 logarithm of the desired attack work factor on a quantum computer
- $n_0 \in \{2, 3, 4\}$: number of circulant blocks forming H
- ϵ: tuning parameter for the DFR ($\epsilon = 0.3 \Rightarrow$ DFR in the range 10^{-9}–10^{-8})

Output:

- p: size of the circulant blocks
- d_v: column weight of H
- t: number of intentional errors
- $\{m_0, m_1, \ldots, m_{n_0-1}\}$, with $\sum_{i=0}^{n_0-1} m_i = m$: weights of the n_0 circulant blocks forming each row of Q

LEDAcrypt automated design procedure

Input:

- Desired security level
  - λ_c: base-2 logarithm of the desired attack work factor on a classical computer
  - λ_q: base-2 logarithm of the desired attack work factor on a quantum computer
- $n_0 \in \{2, 3, 4\}$: number of circulant blocks forming H
- ϵ: tuning parameter for the DFR ($\epsilon = 0.3 \Rightarrow$ DFR in the range 10^{-9}–10^{-8})

Output:

- p: size of the circulant blocks
- d_v: column weight of H
- t: number of intentional errors
- $\{m_0, m_1, \ldots, m_{n_0-1}\}$, with $\sum_{i=0}^{n_0-1} m_i = m$: weights of the n_0 circulant blocks forming each row of Q

Parameters for LEDAcrypt KEM (ephemeral)

NIST Cat.	n_0	p	t	d_v	m	errors out of decodes
1	2	$14,939$	136	11	$[4,3]$	14 out of $1.2 \cdot 10^9$
	3	$7,853$	86	9	$[4,3,2]$	0 out of $1 \cdot 10^9$
	4	$7,547$	69	13	$[2,2,2,1]$	0 out of $1 \cdot 10^9$
3	2	$25,693$	199	13	$[5,3]$	2 out of $1 \cdot 10^9$
	3	$16,067$	127	11	$[4,4,3]$	0 out of $1 \cdot 10^9$
	4	$14,341$	101	15	$[3,2,2,2]$	0 out of $1 \cdot 10^9$
5	2	$36,877$	267	11	$[7,6]$	0 out of $1 \cdot 10^9$
	3	$27,437$	169	15	$[4,4,3]$	0 out of $1 \cdot 10^9$
	4	$22,691$	134	13	$[4,3,3,3]$	0 out of $1 \cdot 10^9$

Deriving them took approximately a day for all the parameter sets with $n_0 \in \{3,4\}$ and approximately a month for all the parameter sets with $n_0 = 2$ on a dual socket AMD EPYC 7551 32-Core CPU. The memory footprint for each parameter seeking process was below 100 MiB.

Message recovery attack WF for LEDAcrypt KEM (ephemeral)

NIST Cat.	n_0	Classical computer ($\log_2$ ♯binary op.s)						Quantum computer ($\log_2$ ♯quantum gates)	
		Prange	L-B	Leon	Stern	F-S	BJMM	Q-LB	Q-Stern
1	2	169.05	158.23	156.35	148.59	148.57	144.37	97.26	98.67
	3	167.72	157.37	154.51	147.67	147.65	144.29	96.14	97.55
	4	169.62	159.40	155.86	149.32	149.31	145.98	97.47	98.22
3	2	234.11	222.19	220.26	210.42	210.41	207.17	130.22	131.62
	3	235.32	223.84	220.82	211.91	211.90	208.71	130.67	132.07
	4	235.98	224.66	220.97	212.39	212.39	209.10	131.26	132.66
5	2	303.56	290.79	288.84	277.40	277.39	274.54	165.18	166.58
	3	303.84	291.53	288.42	277.98	277.98	274.34	165.48	166.88
	4	303.68	291.54	287.78	277.67	277.67	274.91	165.52	166.92

Key recovery attack WF for LEDAcrypt KEM (ephemeral)

NIST Cat.	n_0	Classical computer ($\log_2$ ♯binary op.s)						Quantum computer ($\log_2$ ♯quantum gates)	
		Prange	L-B	Leon	Stern	F-S	BJMM	Q-LB	Q-Stern
1	2	180.25	169.06	167.24	158.76	158.75	154.94	99.21	100.62
	3	169.36	157.78	156.53	147.71	147.68	144.08	93.93	95.34
	4	179.86	167.79	165.69	157.13	157.10	153.01	99.71	101.12
3	2	237.85	225.77	223.87	213.72	213.71	210.64	128.35	129.75
	3	241.70	228.98	227.03	216.59	216.57	213.18	130.56	131.96
	4	254.92	241.73	238.97	228.80	228.78	224.76	137.60	139.01
5	2	315.08	302.11	300.19	288.35	288.34	285.71	167.04	168.44
	3	320.55	306.93	304.48	292.78	292.77	289.00	170.31	171.71
	4	312.68	298.84	295.66	284.59	284.58	280.91	166.82	168.22

Performance of LEDAcrypt KEM (ephemeral)

Software running on an Intel i5-6500, 3.2 GHz

NIST Category	n_0	KeyGen (ms)	Encap. (ms)	Decap. (ms)	Total exec. time (ms)	Ctx+kpub Size (kiB)
1	2	1.32	0.06	0.24	1.62	3.65
	3	0.50	0.03	0.23	0.77	3.04
	4	0.47	0.02	0.26	0.76	3.68
3	2	3.63	0.12	0.61	4.37	6.28
	3	1.72	0.07	0.54	2.33	5.91
	4	1.50	0.07	0.69	2.27	7.03
5	2	7.18	0.20	0.95	8.35	9.01
	3	4.64	0.16	1.05	5.86	10.05
	4	3.83	0.13	1.05	5.02	11.09

Bounding the DFR of iterative decoders

- LEDAcrypt uses a very fast decoding procedure known as Q-decoder.
- The Q-decoder exploits the fact that the public code is defined by $H' = HQ$, where H defines the private code and Q is a secret transformation matrix.
- We define
  - $\mathcal{H}' \leftarrow \texttt{Lift}(H')$
  - $\mathcal{H} \leftarrow \texttt{Lift}(H)$
  - $\mathcal{Q} \leftarrow \texttt{Lift}(Q)$

  matrices obtained through lifting their values from $\mathbb{Z}_2$ to $\mathbb{Z}$.
- A BF procedure acting on H' and a Q-decoding procedure acting on H and Q are equivalent if $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.
- We introduce rejection sampling during key generation to ensure that $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.

Bounding the DFR of iterative decoders

- LEDAcrypt uses a very fast decoding procedure known as Q-decoder.
- The Q-decoder exploits the fact that the public code is defined by $H' = HQ$, where H defines the private code and Q is a secret transformation matrix.
- We define
  - $\mathcal{H}' \leftarrow \texttt{Lift}(H')$
  - $\mathcal{H} \leftarrow \texttt{Lift}(H)$
  - $\mathcal{Q} \leftarrow \texttt{Lift}(Q)$

  matrices obtained through lifting their values from $\mathbb{Z}_2$ to $\mathbb{Z}$.
- A BF procedure acting on H' and a Q-decoding procedure acting on H and Q are equivalent if $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.
- We introduce rejection sampling during key generation to ensure that $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.

Bounding the DFR of iterative decoders

- LEDAcrypt uses a very fast decoding procedure known as Q-decoder.
- The Q-decoder exploits the fact that the public code is defined by $H' = HQ$, where H defines the private code and Q is a secret transformation matrix.
- We define
  - $\mathcal{H}' \leftarrow \texttt{Lift}(H')$
  - $\mathcal{H} \leftarrow \texttt{Lift}(H)$
  - $\mathcal{Q} \leftarrow \texttt{Lift}(Q)$

  matrices obtained through lifting their values from $\mathbb{Z}_2$ to $\mathbb{Z}$.
- A BF procedure acting on H' and a Q-decoding procedure acting on H and Q are equivalent if $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.
- We introduce rejection sampling during key generation to ensure that $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.

Bounding the DFR of iterative decoders

- LEDAcrypt uses a very fast decoding procedure known as Q-decoder.
- The Q-decoder exploits the fact that the public code is defined by $H' = HQ$, where H defines the private code and Q is a secret transformation matrix.
- We define
  - $\mathcal{H}' \leftarrow \texttt{Lift}(H')$
  - $\mathcal{H} \leftarrow \texttt{Lift}(H)$
  - $\mathcal{Q} \leftarrow \texttt{Lift}(Q)$

  matrices obtained through lifting their values from $\mathbb{Z}_2$ to $\mathbb{Z}$.
- A BF procedure acting on H' and a Q-decoding procedure acting on H and Q are equivalent if $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.
- We introduce rejection sampling during key generation to ensure that $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.

Bounding the DFR of iterative decoders

- LEDAcrypt uses a very fast decoding procedure known as Q-decoder.
- The Q-decoder exploits the fact that the public code is defined by $H' = HQ$, where H defines the private code and Q is a secret transformation matrix.
- We define
  - $\mathcal{H}' \leftarrow \texttt{Lift}(H')$
  - $\mathcal{H} \leftarrow \texttt{Lift}(H)$
  - $\mathcal{Q} \leftarrow \texttt{Lift}(Q)$

  matrices obtained through lifting their values from $\mathbb{Z}_2$ to $\mathbb{Z}$.
- A BF procedure acting on H' and a Q-decoding procedure acting on H and Q are equivalent if $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.
- We introduce rejection sampling during key generation to ensure that $\mathcal{H}' = \mathcal{H}\mathcal{Q}$.

Bounding the DFR of iterative decoders (2)

- We consider a Q-decoder performing only two iterations.
- This is also intrinsically prone to constant-time implementation (opposed to decoders performing many iterations).

First iteration

- We use arguments similar to Gallager's statistical analysis of bit flipping.
- We characterize the probability of having a certain number of residual errors after the first iteration.
- This is like considering the average DFR over all possible codes.

Bounding the DFR of iterative decoders (2)

- We consider a Q-decoder performing only two iterations.
- This is also intrinsically prone to constant-time implementation (opposed to decoders performing many iterations).

First iteration

- We use arguments similar to Gallager's statistical analysis of bit flipping.
- We characterize the probability of having a certain number of residual errors after the first iteration.
- This is like considering the average DFR over all possible codes.

Bounding the DFR of iterative decoders (3)

Second iteration

- Aimed at correcting all residual errors.
- Unsatisfied parity-check count: $\mathtt{upc}_z$, $z \in \{0, \ldots, n-1\}$.
- All flips are correct if all $\mathtt{upc}_z$ match or exceed the flipping threshold b for $e_z = 1$, and all $\mathtt{upc}_z$ are below b for $e_z = 0$.
- If the largest value of $\mathtt{upc}_z$ for $e_z = 0$ ($\mathtt{max\_upc}_{no\,flip}$) is smaller than the smallest value of $\mathtt{upc}_z$ for $e_z = 1$ ($\mathtt{min\_upc}_{flip}$), then all errors are corrected when $b = \mathtt{min\_upc}_{flip}$.

- An upper bound can be found on the maximum admissible number of errors t guaranteeing that

$$\mathtt{max\_upc}_{no\,flip} < \mathtt{min\_upc}_{flip} .$$

- This upper bound is specific for each code.

Bounding the DFR of iterative decoders (3)

Second iteration

- Aimed at correcting all residual errors.
- Unsatisfied parity-check count: $\mathtt{upc}_z$, $z \in \{0, \ldots, n-1\}$.
- All flips are correct if all $\mathtt{upc}_z$ match or exceed the flipping threshold b for $e_z = 1$, and all $\mathtt{upc}_z$ are below b for $e_z = 0$.
- If the largest value of $\mathtt{upc}_z$ for $e_z = 0$ ($\mathtt{max\_upc}_{no\,flip}$) is smaller than the smallest value of $\mathtt{upc}_z$ for $e_z = 1$ ($\mathtt{min\_upc}_{flip}$), then all errors are corrected when $b = \mathtt{min\_upc}_{flip}$.

- An upper bound can be found on the maximum admissible number of errors t guaranteeing that

$$\mathtt{max\_upc}_{no\,flip} < \mathtt{min\_upc}_{flip} .$$

- This upper bound is specific for each code.

Upper bound for the second iteration

Theorem

Let:

- H be an $r \times n$ parity-check matrix with constant column weight equal to d_v.
- Q be an $n \times n$ matrix with constant row weight equal to m.
- $\mathbf{I}(e) \subset \{0, \ldots, n-1\}$ define the support of e.
- e be a $1 \times n$ binary error vector with weight t, composed as $e = \sum_{i \in \mathbf{I}(e)} u_i$, where $u_i \in \mathbb{Z}_2^n$, and $wt(u_i) = 1$.
- $\tilde{e}$ be the $1 \times n$ expanded vector $\tilde{e} = eQ^T$.

Upper bound for the second iteration (2)

Theorem (cont.)

One iteration of the Q-decoder, taking as input the $1 \times r$ syndrome $s = \tilde{e}H^T$, retrieves the values of all the bits of the actual error vector e if $t < \frac{\alpha+\beta}{\gamma+\beta}$, where

$$\begin{cases} \alpha = \min_{z \in \mathbf{I}(e)} \left\{ wt\left(P_{(z)} \left(u_z Q^T \right)^T \right) \right\}, \\ \beta = \max_{z,i \in \mathbf{I}(e), z \neq i} \left\{ wt\left(P_{(z)} \left(u_z Q^T \right)^T \wedge P_{(z)} \left(u_i Q^T \right)^T \right) \right\}, \\ \gamma = \max_{z,i \in \mathbf{I}(e), z \neq i} \left\{ wt\left(P_{(z)} \left(u_i Q^T \right)^T \right) \right\}, \end{cases}$$

where $P_{(z)}(\cdot)$ gives a matrix containing a set of parity-check equations (i.e., rows of H) that contribute to upc_z and allows to compute the contribution to the value upc_z given by its argument.

Long-term keys and IND-CCA

Further rejection sampling

A second rejection sampling is performed during key generation to ensure that the second decoder iteration corrects all errors of weight $\leq \bar{t}$ left by the first decoder iteration.

Target DFR

Code parameters are chosen such that the first iteration of the Q-decoder results in at most $\bar{t}$ residual errors with probability $> 1 - \rho$, where ρ is the target DFR value.

Long-term keys and IND-CCA (2)

- We start from the parameters obtained through the automated parameter optimization procedure for instances with ephemeral keys.
- We keep md_v constant (or slightly increased), and increase the size of the circulant blocks until the probability that the number of residual errors after the first iteration is $\leq \bar{t}$ becomes larger than the given target.
- The optimal flipping threshold (corresponding to the maximum value of the probability) is found exhaustively.
- These changes may only impact positively on the security margin against ISD attacks and key recovery attacks.

Long-term keys and IND-CCA (2)

- We start from the parameters obtained through the automated parameter optimization procedure for instances with ephemeral keys.
- We keep md_v constant (or slightly increased), and increase the size of the circulant blocks until the probability that the number of residual errors after the first iteration is $\leq \overline{t}$ becomes larger than the given target.
- The optimal flipping threshold (corresponding to the maximum value of the probability) is found exhaustively.
- These changes may only impact positively on the security margin against ISD attacks and key recovery attacks.

Long-term keys and IND-CCA (2)

- We start from the parameters obtained through the automated parameter optimization procedure for instances with ephemeral keys.
- We keep md_v constant (or slightly increased), and increase the size of the circulant blocks until the probability that the number of residual errors after the first iteration is $\leq \overline{t}$ becomes larger than the given target.
- The optimal flipping threshold (corresponding to the maximum value of the probability) is found exhaustively.
- These changes may only impact positively on the security margin against ISD attacks and key recovery attacks.

Long-term keys and IND-CCA (2)

- We start from the parameters obtained through the automated parameter optimization procedure for instances with ephemeral keys.
- We keep md_v constant (or slightly increased), and increase the size of the circulant blocks until the probability that the number of residual errors after the first iteration is $\leq \overline{t}$ becomes larger than the given target.
- The optimal flipping threshold (corresponding to the maximum value of the probability) is found exhaustively.
- These changes may only impact positively on the security margin against ISD attacks and key recovery attacks.

LEDAcrypt instances with long-term keys

NIST Cat.	n_0	DFR	p	t	d_v	m	$\bar{t}$	keys out of 100 with target DFR	b_0
1	2	2^{-64}	35,899	136	9	[5, 4]	4	95	44
	2	2^{-128}	52,147	136	9	[5, 4]	4	95	43
3	2	2^{-64}	57,899	199	11	[6, 5]	5	92	64
	2	2^{-192}	96,221	199	11	[6, 5]	5	92	64
5	2	2^{-64}	89,051	267	13	[7, 6]	6	93	89
	2	2^{-256}	152,267	267	13	[7, 6]	6	93	88

Weak keys in LEDAcrypt

- Presented by Daniel Apon, Corbin McNeill, Ray Perlner and Angela Robinson at the 2019 Quantum Cryptanalysis Dagstuhl Seminar (Oct. 2019).
- Leverage the product structure of the public code parity-check matrix ($H' = HQ$).

Rationale

Making guesses separately on H and Q (and projecting them onto H') accelerates ISD with respect to making them directly on H'.

- One key is weak if
  - Occurs with probability 2^{-x}.
  - Requires the equivalent of 2^y AES operations for ISD.
  - $x + y < \lambda$, being λ the claimed security level.

Weak keys in LEDAcrypt

- Presented by Daniel Apon, Corbin McNeill, Ray Perlner and Angela Robinson at the 2019 Quantum Cryptanalysis Dagstuhl Seminar (Oct. 2019).
- Leverage the product structure of the public code parity-check matrix ($H' = HQ$).

Rationale

Making guesses separately on H and Q (and projecting them onto H') accelerates ISD with respect to making them directly on H'.

- One key is weak if
  - Occurs with probability 2^{-x}.
  - Requires the equivalent of 2^y AES operations for ISD.
  - $x + y < \lambda$, being λ the claimed security level.

Weak keys in LEDAcrypt

- Presented by Daniel Apon, Corbin McNeill, Ray Perlner and Angela Robinson at the 2019 Quantum Cryptanalysis Dagstuhl Seminar (Oct. 2019).
- Leverage the product structure of the public code parity-check matrix ($H' = HQ$).

Rationale

Making guesses separately on H and Q (and projecting them onto H') accelerates ISD with respect to making them directly on H'.

- One key is weak if
  - Occurs with probability 2^{-x}.
  - Requires the equivalent of 2^y AES operations for ISD.
  - $x + y < \lambda$, being λ the claimed security level.

Weak key examples

$n_0 = 2$, cat. 5, IND-CPA

$x \approx 44$, $y \approx 52$

$n_0 = 4$, cat. 1, IND-CPA

$x \approx 40$, $y \approx 50$

- This attack works well when:
  - n_0 is small (tested for $n_0 = 2$),
  - the weights of H and Q are well balanced.
- Countermeasures:
  - increase n_0,
  - choose unbalanced weights for H and Q.
- Work is in progress...

Weak key examples

$n_0 = 2$, cat. 5, IND-CPA

$x \approx 44$, $y \approx 52$

$n_0 = 4$, cat. 1, IND-CPA

$x \approx 40$, $y \approx 50$

- This attack works well when:
  - n_0 is small (tested for $n_0 = 2$),
  - the weights of H and Q are well balanced.
- Countermeasures:
  - increase n_0,
  - choose unbalanced weights for H and Q.
- Work is in progress...

Weak key examples

$n_0 = 2$, cat. 5, IND-CPA

$x \approx 44$, $y \approx 52$

$n_0 = 4$, cat. 1, IND-CPA

$x \approx 40$, $y \approx 50$

- This attack works well when:
  - n_0 is small (tested for $n_0 = 2$),
  - the weights of H and Q are well balanced.
- Countermeasures:
  - increase n_0,
  - choose unbalanced weights for H and Q.
- Work is in progress...

Weak key examples

$n_0 = 2$, cat. 5, IND-CPA

$x \approx 44$, $y \approx 52$

$n_0 = 4$, cat. 1, IND-CPA

$x \approx 40$, $y \approx 50$

- This attack works well when:
  - n_0 is small (tested for $n_0 = 2$),
  - the weights of H and Q are well balanced.
- Countermeasures:
  - increase n_0,
  - choose unbalanced weights for H and Q.
- Work is in progress...

End of presentation

Thank you!

www.univpm.it/marco.baldi
m.baldi@univpm.it

CBCrypto 2020
(cbcrypto.dii.univpm.it)

← submit your papers!

LEDAcrypt KEM-LT (IND-CCA2) Performance

Software running on an Intel i5-6500, 3.2 GHz

NIST Category	n_0	DFR	KeyGen (ms)	Encap. (ms)	Decap. (ms)	Enc+Dec time (ms)	Ctx size (kiB)
1	2	2^{-64}	6.87	0.09	0.33	0.43	4.38
	2	2^{-128}	11.64	0.16	0.46	0.63	6.37
3	2	2^{-64}	14.74	0.24	0.69	0.99	7.07
	2	2^{-192}	30.17	0.42	0.99	1.42	11.75
5	2	2^{-64}	28.65	0.52	1.33	1.86	10.87
	2	2^{-256}	58.54	0.81	2.04	2.86	18.60

LEDAcrypt PKE (IND-CCA2) Performance

Software running on an Intel i5-6500, 3.2 GHz

NIST Category	n_0	DFR	KeyGen (ms)	Encap. (ms)	Decap. (ms)	Enc+Dec time (ms)
1	2	2^{-64}	6.87	0.31	0.69	1.00
	2	2^{-128}	11.64	0.44	0.99	1.42
3	2	2^{-64}	14.74	0.56	1.30	1.86
	2	2^{-192}	30.17	1.04	2.03	3.07
5	2	2^{-64}	28.65	1.03	2.49	3.52
	2	2^{-256}	58.54	1.62	3.86	5.48

Cost of adding IND-CCA2

Comparison between IND-CPA and IND-CCA2 KEMs, synthetic metric μ computed as $\mu = \text{cycles} + 1000 \times B$, (B transmitted bytes). Ratio computed as $\frac{\mu_{CCA} - \mu_{CPA}}{\mu_{CPA}}$ selecting the best performing IND-CPA option (among $n_0 \in \{2, 3, 4\}$) for the security level. Red color highlights an extra cost for IND-CCA2, green highlights a saving.

NIST Category	n_0	DFR	$\frac{\text{cycles}_{CCA} - \text{cycles}_{CPA}}{\text{cycles}_{CPA}}$	$\frac{B_{CCA} - B_{CPA}}{B_{CPA}}$	$\frac{\mu_{cca} - \mu_{cpa}}{\mu_{cpa}}$
1	2	2^{-64}	-47.5%	44.6%	6.4%
	2	2^{-128}	-24.5%	109.7%	54.0%
3	2	2^{-64}	-58.2%	20.2%	-28.3%
	2	2^{-192}	-32.4%	99.5%	18.1%
5	2	2^{-64}	-69.3%	21.1%	-41.9%
	2	2^{-256}	-48.2%	106.7%	-1.2%

IND-CPA (ephemeral key) options require more computation but less bandwidth

Non-Algebraic, Hamming metric code-based KEMs, Long Term use

`supercop-20190816, Intel Xeon E3-1220 v3 (haswell), hiphop`

Supercop tag	Time (**kcycles**)	transmitted (**B**)	cycles+1000×B
`ledakemlt10`	1512	4488	6000740
`hqc1281`	1603	6234	7837752
`ledakemlt11`	2292	6520	8812464
`ledakemlt30`	3260	7240	10500136
`hqc1921`	2789	10981	13770772
`hqc1922`	2901	11749	14650164
`ledakemlt50`	6414	11136	17550216
`ledakemlt31`	5793	12032	17825724
`hqc2561`	4309	15961	20270712
`hqc2562`	4576	16985	21561072
`hqc2563`	4695	17777	22472212
`ledakemlt51`	11393	19040	30433952

Non-Algebraic, Hamming metric code-based KEMs, Cat. 1, Eph. use

What are the best computation vs bandwidth tradeoffs? / Which n_0 should be picked?

`supercop-20190816, Intel Xeon E3-1220 v3 (haswell), hiphop`

Supercop tag	Time (**kcycles**)	transmitted (**B**)	cycles+1000×B
`ledakem13`	2635	3120	5755764
`bike1l1nc`	1596	5084	6680112
`ledakem14`	2964	3776	6740276
`bike3l1nc`	1595	5516	7111960
`bike1l1`	3407	5084	8491364
`ledakem12`	5470	3744	9214880
`bike3l1`	4302	5516	9818592
`bike1l1sc`	4797	5084	9881160
`hqc1281`	1840	9359	11199668
`bike2l1`	7326	5084	12410180
`bike3l1sc`	6949	5516	12465900

Non-Algebraic, Hamming metric code-based KEMs, Cat. 3, Eph. use

`supercop-20190811, Intel Xeon E3-1220 v5 (Skylake)`

Supercop tag	Time (kc) (**kcycles**)	transmitted (**B**)	c+1000×b
`ledakem33`	1,353	6,048	13539812
`ledakem34`	1,426	7,200	14269705
`hqc1921`	1,913	16,480	19139559
`hqc1922`	2,039	17,633	20391339
`ledakem32`	2,302	6,432	23024615

Non-Algebraic, Hamming metric code-based KEMs, Cat. 5, Eph. use

`supercop-20190811, Intel Xeon E3-1220 v5 (Skylake)`

Supercop tag	Time (kc) (**kcycles**)	transmitted (**B**)	c+1000×b
`ledakem54`	16,681	11,360	28041294
`hqc2562`	4,214	25,488	29702525
`hqc2563`	4,369	26,674	31043365
`ledakem53`	21,836	10,296	32132565
`ledakem52`	35,343	9,232	44575781