Code-based Cryptography and Post-Quantum Standardization

Edoardo Persichetti

Dipartimento di Matematica, Università di Roma, Tor Vergata

14 December 2020

CHARLES E. SCHMIDT
COLLEGE OF SCIENCE
Florida Atlantic University

In This Talk

- Motivation
- Intro: a bit of Background
- Conservative Code-Based Cryptography
- Structured Codes
- Sparse-Matrix Codes
- Conclusions

Part I

MOTIVATION

Revolutionary approach (Rivest, Shamir, Adleman 1978, Diffie-Hellmann 1976)

PUBLIC-KEY CRYPTOGRAPHY

Revolutionary approach (Rivest, Shamir, Adleman 1978, Diffie-Hellmann 1976)

Asymmetric key framework based on hard math problem (trapdoor).

Public-Key Cryptography

Revolutionary approach (Rivest, Shamir, Adleman 1978, Diffie-Hellmann 1976)

Asymmetric key framework based on hard math problem (trapdoor).
Two keys (public/private) for different tasks.

Public-Key Cryptography

Revolutionary approach (Rivest, Shamir, Adleman 1978, Diffie-Hellmann 1976)

Asymmetric key framework based on hard math problem (trapdoor).
Two keys (public/private) for different tasks.

$$\boxed{\text{Plaintext}} \quad \underset{SK}{\overset{PK}{\rightleftarrows}} \quad \boxed{\text{Ciphertext}}$$

PUBLIC-KEY CRYPTOGRAPHY

Revolutionary approach (Rivest, Shamir, Adleman 1978, Diffie-Hellmann 1976)

Asymmetric key framework based on hard math problem (trapdoor).
Two keys (public/private) for different tasks.

$$\boxed{\text{Plaintext}} \quad \underset{SK}{\overset{PK}{\rightleftarrows}} \quad \boxed{\text{Ciphertext}}$$

Ideal for a variety of applications:

PUBLIC-KEY CRYPTOGRAPHY

Revolutionary approach (Rivest, Shamir, Adleman 1978, Diffie-Hellmann 1976)

Asymmetric key framework based on hard math problem (trapdoor).
Two keys (public/private) for different tasks.

$$\boxed{\text{Plaintext}} \quad \underset{SK}{\overset{PK}{\rightleftarrows}} \quad \boxed{\text{Ciphertext}}$$

Ideal for a variety of applications:

- Encryption
- Signatures
- Key Exchange
- ...

Post-Quantum Cryptography

In a few years time large-scale quantum computers might be reality.

In a few years time large-scale quantum computers might be reality. PKC largely based on number theory problems (factoring, discrete log). But then (Shor, '95):

Post-Quantum Cryptography

In a few years time large-scale quantum computers might be reality. PKC largely based on number theory problems (factoring, discrete log). But then (Shor, '95):

- RSA
- DSA
- ECC
- Diffie-Hellman key exchange
- and many others ... $\boxed{\textbf{not secure}}$!

POST-QUANTUM CRYPTOGRAPHY

In a few years time large-scale quantum computers might be reality. PKC largely based on number theory problems (factoring, discrete log). But then (Shor, '95):

- RSA
- DSA
- ECC
- Diffie-Hellman key exchange
- and many others ... $\boxed{\textbf{not secure}}$!

$\rightarrow$ NIST's Post-Quantum Cryptography Standardization Call

Post-Quantum Cryptography

In a few years time large-scale quantum computers might be reality. PKC largely based on number theory problems (factoring, discrete log). But then (Shor, '95):

- RSA
- DSA
- ECC
- Diffie-Hellman key exchange
- and many others ... **not secure** !

→ NIST's Post-Quantum Cryptography Standardization Call

Main areas of research:

- Lattice-based cryptography.
- Hash-based cryptography.
- Code-based cryptography (McEliece, Niederreiter).
- Multivariate cryptography.
- Isogeny-based cryptography.

Part II

Intro: a bit of Background

WHAT IS CODE-BASED CRYPTOGRAPHY?

What is Code-Based Cryptography?

The family of cryptographic primitives based on the following.

WHAT IS CODE-BASED CRYPTOGRAPHY?

The family of cryptographic primitives based on the following.

PROBLEM (COMPUTATIONAL SYNDROME DECODING)

Given: $H \in \mathbb{F}_q^{(n-k)\times n}$, $y \in \mathbb{F}_q^{(n-k)}$ and $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ with $wt(e) \leq w$ such that $He^T = y$.

What is Code-Based Cryptography?

The family of cryptographic primitives based on the following.

Problem (Computational Syndrome Decoding)

Given: $H \in \mathbb{F}_q^{(n-k)\times n}$, $y \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $He^T = y$.

Decisional version: NP-Complete (Berlekamp, McEliece and van Tilborg, 1978).

What is Code-Based Cryptography?

The family of cryptographic primitives based on the following.

Problem (Computational Syndrome Decoding)

Given: $H \in \mathbb{F}_q^{(n-k)\times n}$, $y \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $He^T = y$.

Decisional version: NP-Complete (Berlekamp, McEliece and van Tilborg, 1978).

Unique solution when w is below a certain threshold.

What is Code-Based Cryptography?

The family of cryptographic primitives based on the following.

Problem (Computational Syndrome Decoding)

Given: $H \in \mathbb{F}_q^{(n-k)\times n}$, $y \in \mathbb{F}_q^{(n-k)}$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $He^T = y$.

Decisional version: NP-Complete (Berlekamp, McEliece and van Tilborg, 1978).

Unique solution when w is below a certain threshold.

GV Bound

For a given finite field $\mathbb{F}_q$ and integers n, k, the Gilbert-Varshamov (GV) distance is the largest integer d_0 such that

$$|\mathcal{B}(0, d_0 - 1)| \leq q^{n-k}$$

where $\mathcal{B}(x, r) = \{y \in \mathbb{F}_q^n \mid d(x, y) \leq r\}$ is the n-dimensional ball of radius r centered in x.

$[n, k]$ LINEAR CODE OVER $\mathbb{F}_q$

A subspace of dimension k of $\mathbb{F}_q^n$.
w-error correcting: $\exists$ algorithm that corrects up to w errors.

$[n, k]$ LINEAR CODE OVER $\mathbb{F}_q$

A subspace of dimension k of $\mathbb{F}_q^n$.
w-error correcting: $\exists$ algorithm that corrects up to w errors.

HAMMING METRIC

$wt(x) = |\{i : x_i \neq 0, 1 \leq i \leq n\}|$, $d(x, y) = wt(x - y)$.

Minimum distance (of $\mathcal{C}$): $\min\{d(x, y) : x, y \in \mathcal{C}\}$.

Error-Correcting Codes

$[n, k]$ linear code over $\mathbb{F}_q$

A subspace of dimension k of $\mathbb{F}_q^n$.
w-error correcting: $\exists$ algorithm that corrects up to w errors.

Hamming metric

$wt(x) = |\{i : x_i \neq 0, 1 \leq i \leq n\}|$, $d(x, y) = wt(x - y)$.

Minimum distance (of $\mathcal{C}$): $\min\{d(x, y) : x, y \in \mathcal{C}\}$.

Generator matrix

$G \in \mathbb{F}_q^{k \times n}$ defines the code as follows: $x \in \mathcal{C}_G \iff x = \mu G$ for $\mu \in \mathbb{F}_q^k$.
Systematic form: $(I_k | M)$.

Error-Correcting Codes

$[n, k]$ linear code over $\mathbb{F}_q$

A subspace of dimension k of $\mathbb{F}_q^n$.
w-error correcting: $\exists$ algorithm that corrects up to w errors.

Hamming metric

$wt(x) = |\{i : x_i \neq 0, 1 \leq i \leq n\}|$, $d(x, y) = wt(x - y)$.
Minimum distance (of $\mathcal{C}$): $\min\{d(x, y) : x, y \in \mathcal{C}\}$.

Generator matrix

$G \in \mathbb{F}_q^{k \times n}$ defines the code as follows: $x \in \mathcal{C}_G \iff x = \mu G$ for $\mu \in \mathbb{F}_q^k$.
Systematic form: $(I_k | M)$.

Parity-check matrix

$H \in \mathbb{F}_q^{(n-k) \times n}$ defines the code as follows: $x \in \mathcal{C}_H \iff Hx^T = 0$.
Systematic form: $(M^T | I_{n-k})$.

In general, it is hard to decode random codes.

(De)Coding Problems

In general, it is hard to decode random codes.

Problem (General Decoding)

Given: $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ and $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ with $wt(e) \leq w$ such that $y - e \in \mathcal{C}_G$.

(DE)CODING PROBLEMS

In general, it is hard to decode random codes.

PROBLEM (GENERAL DECODING)

Given: $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e \in \mathcal{C}_G$.

Easy to see the two problems are equivalent.

(De)Coding Problems

In general, it is hard to decode random codes.

Problem (General Decoding)

Given: $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e \in \mathcal{C}_G$.

Easy to see the two problems are equivalent.

To get trapdoor, need one more ingredient.

(DE)CODING PROBLEMS

In general, it is hard to decode random codes.

PROBLEM (GENERAL DECODING)

Given: $G \in \mathbb{F}_q^{k \times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e \in \mathcal{C}_G$.

Easy to see the two problems are equivalent.

To get trapdoor, need one more ingredient.

ASSUMPTION (CODE INDISTINGUISHABILITY)

Let M be a matrix defining a code. Then M is indistinguishable from a randomly generated matrix of the same size.

(De)Coding Problems

In general, it is hard to decode random codes.

Problem (General Decoding)

Given: $G \in \mathbb{F}_q^{k\times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e \in \mathcal{C}_G$.

Easy to see the two problems are equivalent.

To get trapdoor, need one more ingredient.

Assumption (Code Indistinguishability)

Let M be a matrix defining a code. Then M is indistinguishable from a randomly generated matrix of the same size.

Choose a code family with efficient decoding algorithm associated to description Δ and hide the structure.

In general, it is hard to decode random codes.

PROBLEM (GENERAL DECODING)

Given: $G \in \mathbb{F}_q^{k\times n}$, $y \in \mathbb{F}_q^n$ *and* $w \in \mathbb{N}$.
Goal: find a word $e \in \mathbb{F}_q^n$ *with* $wt(e) \leq w$ *such that* $y - e \in \mathcal{C}_G$.

Easy to see the two problems are equivalent.

To get trapdoor, need one more ingredient.

ASSUMPTION (CODE INDISTINGUISHABILITY)

Let M be a matrix defining a code. Then M is indistinguishable from a randomly generated matrix of the same size.

Choose a code family with efficient decoding algorithm associated to description Δ and hide the structure.

Hardness of assumption depends on chosen code family.

McEliece: first proposal (1978), based on GDP.

Code-based Cryptosystems

McEliece: first proposal (1978), based on GDP.

Chosen code family: binary Goppa codes.

Code-based Cryptosystems

McEliece: first proposal (1978), based on GDP.

Chosen code family: binary Goppa codes.

KeyGen chooses generator matrix G and forms public key as SGP.

Code-based Cryptosystems

McEliece: first proposal (1978), based on GDP.

Chosen code family: binary Goppa codes.

KeyGen chooses generator matrix G and forms public key as SGP.

Plaintext is encrypted as noisy codeword (scheme is probabilistic).

McEliece: first proposal (1978), based on GDP.

Chosen code family: binary Goppa codes.

KeyGen chooses generator matrix G and forms public key as SGP.

Plaintext is encrypted as noisy codeword (scheme is probabilistic).

Niederreiter: “dual”/equivalent version (1985), based on SDP.

CODE-BASED CRYPTOSYSTEMS

McEliece: first proposal (1978), based on GDP.

Chosen code family: binary Goppa codes.

KeyGen chooses generator matrix G and forms public key as SGP.

Plaintext is encrypted as noisy codeword (scheme is probabilistic).

Niederreiter: "dual"/equivalent version (1985), based on SDP.

Chosen code family: Generalized Reed-Solomon (GRS) codes.

Code-based Cryptosystems

McEliece: first proposal (1978), based on GDP.

Chosen code family: binary Goppa codes.

KeyGen chooses generator matrix G and forms public key as SGP.

Plaintext is encrypted as noisy codeword (scheme is probabilistic).

Niederreiter: "dual"/equivalent version (1985), based on SDP.

Chosen code family: Generalized Reed-Solomon (GRS) codes.

KeyGen chooses parity-check matrix H and forms public key as SHP.

McEliece: first proposal (1978), based on GDP.

Chosen code family: binary Goppa codes.

KeyGen chooses generator matrix G and forms public key as SGP.

Plaintext is encrypted as noisy codeword (scheme is probabilistic).

Niederreiter: "dual"/equivalent version (1985), based on SDP.

Chosen code family: Generalized Reed-Solomon (GRS) codes.

KeyGen chooses parity-check matrix H and forms public key as SHP.

Plaintext is encrypted as low-weight vector (scheme is deterministic).

McEliece PKE (Modern)

Key Generation

- Choose w-error correcting code $\mathcal{C}$.
- SK: code description Δ for $\mathcal{C}$.
- PK: generator matrix G in systematic form for $\mathcal{C}$.

McEliece PKE (Modern)

Key Generation

- Choose w-error correcting code $\mathcal{C}$.
- SK: code description Δ for $\mathcal{C}$.
- PK: generator matrix G in systematic form for $\mathcal{C}$.

Encryption

- Plaintext is a word $\mu \in \mathbb{F}_2^k$.
- Select random error vector $e \in \mathbb{F}_2^n$ of weight w.
- $c = \mu G + e$.

McEliece PKE (Modern)

Key Generation

- Choose w-error correcting code $\mathcal{C}$.
- SK: code description Δ for $\mathcal{C}$.
- PK: generator matrix G in systematic form for $\mathcal{C}$.

Encryption

- Plaintext is a word $\mu \in \mathbb{F}_2^k$.
- Select random error vector $e \in \mathbb{F}_2^n$ of weight w.
- $c = \mu G + e$.

Decryption

- Set $\mu = Decode_{\Delta}(c)$ and return μ.
- Return $\perp$ if decoding fails.

NIEDERREITER PKE (MODERN)

KEY GENERATION

- Choose w-error correcting code $\mathcal{C}$.
- SK: code description Δ for $\mathcal{C}$.
- PK: parity-check matrix H in systematic form for $\mathcal{C}$.

Niederreiter PKE (Modern)

Key Generation

- Choose w-error correcting code $\mathcal{C}$.
- SK: code description Δ for $\mathcal{C}$.
- PK: parity-check matrix H in systematic form for $\mathcal{C}$.

Encryption

- Plaintext is a word $e \in \mathbb{F}_2^n$ of weight w.
- $c = He^T$.

Niederreiter PKE (Modern)

Key Generation

- Choose w-error correcting code $\mathcal{C}$.
- SK: code description Δ for $\mathcal{C}$.
- PK: parity-check matrix H in systematic form for $\mathcal{C}$.

Encryption

- Plaintext is a word $e \in \mathbb{F}_2^n$ of weight w.
- $c = He^T$.

Decryption

- Set $e = Decode_\Delta(c)$ and return e.
- Return $\perp$ if decoding fails.

SECURITY

Both encryption schemes are only OW-CPA (OW-Passive) secure.

SECURITY

Both encryption schemes are only OW-CPA (OW-Passive) secure.

Given that assumption is true, best attack is generic search on random codes.

SECURITY

Both encryption schemes are only OW-CPA (OW-Passive) secure.

Given that assumption is true, best attack is generic search on random codes.

Paradigm: Information Set Decoding (ISD)(Prange,1962).

Both encryption schemes are only OW-CPA (OW-Passive) secure.

Given that assumption is true, best attack is generic search on random codes.

Paradigm: Information Set Decoding (ISD)(Prange,1962).

In a nutshell: look for Information Set (set of columns carrying the information symbols) which is error-free.

SECURITY

Both encryption schemes are only OW-CPA (OW-Passive) secure.

Given that assumption is true, best attack is generic search on random codes.

Paradigm: Information Set Decoding (ISD)(Prange,1962).

In a nutshell: look for Information Set (set of columns carrying the information symbols) which is error-free.

Several variants use Birthday Paradox and other tricks to obtain some speed-ups.

SECURITY

Both encryption schemes are only OW-CPA (OW-Passive) secure.

Given that assumption is true, best attack is generic search on random codes.

Paradigm: Information Set Decoding (ISD)(Prange,1962).

In a nutshell: look for Information Set (set of columns carrying the information symbols) which is error-free.

Several variants use Birthday Paradox and other tricks to obtain some speed-ups.

Complexity $2^{w(c+o(1))}$, constant c depending on algorithm, code and error rate.

SECURITY

Both encryption schemes are only OW-CPA (OW-Passive) secure.

Given that assumption is true, best attack is generic search on random codes.

Paradigm: Information Set Decoding (ISD)(Prange,1962).

In a nutshell: look for Information Set (set of columns carrying the information symbols) which is error-free.

Several variants use Birthday Paradox and other tricks to obtain some speed-ups.

Complexity $2^{w(c+o(1))}$, constant c depending on algorithm, code and error rate.

Use ISD as a tool to assess security level.

Part III

CONSERVATIVE CODE-BASED CRYPTOGRAPHY

Overall Strategy

1. Code family.

OVERALL STRATEGY

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

Overall Strategy

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

$\rightarrow$ Plain binary Goppa codes secure for 40 years.

Overall Strategy

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

$\rightarrow$ Plain binary Goppa codes secure for 40 years.

2. Protocol.

Overall Strategy

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

$\rightarrow$ Plain binary Goppa codes secure for 40 years.

2. Protocol.

Ideal use of PKC: exchange a key for symmetric cipher.

OVERALL STRATEGY

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

$\rightarrow$ Plain binary Goppa codes secure for 40 years.

2. Protocol.

Ideal use of PKC: exchange a key for symmetric cipher.

$\rightarrow$ Focus on designing KEM.

Overall Strategy

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

$\rightarrow$ Plain binary Goppa codes secure for 40 years.

2. Protocol.

Ideal use of PKC: exchange a key for symmetric cipher.

$\rightarrow$ Focus on designing KEM.

3. Framework.

OVERALL STRATEGY

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

$\rightarrow$ Plain binary Goppa codes secure for 40 years.

2. Protocol.

Ideal use of PKC: exchange a key for symmetric cipher.

$\rightarrow$ Focus on designing KEM.

3. Framework.

Since we use a KEM, “plaintext” is randomly generated.

1. Code family.

Nearly every code choice has been shown to be insecure

- GRS
- Reed-Muller
- Concatenated
- Elliptic
- ...

$\rightarrow$ Plain binary Goppa codes secure for 40 years.

2. Protocol.

Ideal use of PKC: exchange a key for symmetric cipher.

$\rightarrow$ Focus on designing KEM.

3. Framework.

Since we use a KEM, “plaintext” is randomly generated.

$\rightarrow$ More practical to use Niederreiter.

The KEM-DEM framework

Introduced by Shoup (2000), combines the actions of two independent mechanisms.

The KEM-DEM framework

Introduced by Shoup (2000), combines the actions of two independent mechanisms.

Key Encapsulation Mechanism (KEM)

- KeyGen: generates private key SK and public key PK.
- $\mathrm{Enc}^{KEM}(PK)$: produces a symmetric key K and a ciphertext c_0.
- $\mathrm{Dec}^{KEM}(SK, c_0)$: returns the symmetric key K (or $\perp$).

The KEM-DEM framework

Introduced by Shoup (2000), combines the actions of two independent mechanisms.

Key Encapsulation Mechanism (KEM)

- KeyGen: generates private key SK and public key PK.
- $\mathrm{Enc}^{KEM}(PK)$: produces a symmetric key K and a ciphertext c_0.
- $\mathrm{Dec}^{KEM}(SK, c_0)$: returns the symmetric key K (or $\perp$).

Data Encapsulation Mechanism (DEM)

- $\mathrm{Enc}^{DEM}(K, m)$: produces the ciphertext c_1.
- $\mathrm{Dec}^{DEM}(K, c_1)$: returns the plaintext m (or $\perp$).

HYBRID ENCRYPTION SCHEME

- KeyGen: generates private key SK and public key PK.
- $\mathrm{Enc}^{HY}(PK, m)$:
  - Run $\mathrm{Enc}^{KEM}(PK)$ and get (K, c_0).
  - Run $\mathrm{Enc}^{DEM}(K, m)$ and get c_1.
  - Final ciphertext $c = (c_0, c_1)$.
- $\mathrm{Dec}^{HY}(SK, c)$:
  - Run $\mathrm{Dec}^{KEM}(SK, c_0)$ and get K.
  - Run $\mathrm{Dec}^{DEM}(K, c_1)$ and recover m.

Key Exchange from KEMs

Standard procedure to convert KEM into a key exchange.

Standard procedure to convert KEM into a key exchange.

Alice		Bob
$(SK, PK) \leftarrow \text{KeyGen}$	$\xrightarrow{\quad PK \quad}$	
	$\xleftarrow{\quad c_0 \quad}$	$(K, c_0) \leftarrow \text{Enc}^{KEM}(PK)$
$K/\perp \leftarrow \text{Dec}^{KEM}(SK, c_0)$		

Key Exchange from KEMs

Standard procedure to convert KEM into a key exchange.

Alice		Bob
$(SK, PK) \leftarrow \text{KeyGen}$	$\xrightarrow{\quad PK \quad}$	
	$\xleftarrow{\quad c_0 \quad}$	$(K, c_0) \leftarrow \text{Enc}^{KEM}(PK)$
$K/\perp \leftarrow \text{Dec}^{KEM}(SK, c_0)$		

The shared key is K.

Key Exchange from KEMs

Standard procedure to convert KEM into a key exchange.

Alice		Bob
$(SK, PK) \leftarrow \text{KeyGen}$	$\xrightarrow{\quad PK \quad}$	
	$\xleftarrow{\quad c_0 \quad}$	$(K, c_0) \leftarrow \text{Enc}^{KEM}(PK)$
$K/\perp \leftarrow \text{Dec}^{KEM}(SK, c_0)$		

The shared key is K.

"Asymmetric" structure key exchange suitable for different scenarios.

Select hash functions $\mathbf{H}, \mathbf{K}$ (in practice, just use SHAKE-256).

Classic McEliece: a binary Goppa-based KEM

Select hash functions $\mathbf{H}, \mathbf{K}$ (in practice, just use SHAKE-256).

Key Generation

- Choose a Goppa code $\mathcal{C}$.
- SK: description $(g, \alpha_1, \ldots \alpha_n)$ for $\mathcal{C}$ plus random string s.
- PK: parity-check matrix H in systematic form for $\mathcal{C}$.

Classic McEliece: a binary Goppa-based KEM

Select hash functions $\mathbf{H}, \mathbf{K}$ (in practice, just use SHAKE-256).

Key Generation

- Choose a Goppa code $\mathcal{C}$.
- SK: description $(g, \alpha_1, \dots \alpha_n)$ for $\mathcal{C}$ plus random string s.
- PK: parity-check matrix H in systematic form for $\mathcal{C}$.

Encapsulation

- Sample a word $e \in \mathbb{F}_2^n$ of weight w.
- $c = (c_0, c_1)$ where $c_0 = He^T$, $c_1 = \mathbf{H}(e)$.
- $K = \mathbf{K}(c, e)$

Classic McEliece: a binary Goppa-based KEM

Select hash functions $\mathbf{H}, \mathbf{K}$ (in practice, just use SHAKE-256).

Key Generation

- Choose a Goppa code $\mathcal{C}$.
- SK: description $(g, \alpha_1, \ldots \alpha_n)$ for $\mathcal{C}$ plus random string s.
- PK: parity-check matrix H in systematic form for $\mathcal{C}$.

Encapsulation

- Sample a word $e \in \mathbb{F}_2^n$ of weight w.
- $c = (c_0, c_1)$ where $c_0 = He^T$, $c_1 = \mathbf{H}(e)$.
- $K = \mathbf{K}(c, e)$

Decapsulation

- Set $e' = Decode(c_0)$.
- $c' = (c'_0, c'_1)$ where $c'_0 = He'^T$, $c'_1 = \mathbf{H}(e')$.
- Return $K = \mathbf{K}(c', s)$ if decoding fails or $c \neq c'$.
- Else return $K = \mathbf{K}(c', e')$.

About Classic McEliece

Classic McEliece parameters (bytes):

m	n	w	PK Size	SK Size	Ciph Size	Security
13	8,192	128	1,357,824	14,080	240	5
13	6,960	119	1,046,739	13,908	226	5
13	6,688	128	1,044,992	13,892	240	5
13	4,608	96	524,160	13,568	188	3
12	3,488	64	261,120	6,452	128	1

Classic McEliece parameters (bytes):

m	n	w	PK Size	SK Size	Ciph Size	Security
13	8,192	128	1,357,824	14,080	240	5
13	6,960	119	1,046,739	13,908	226	5
13	6,688	128	1,044,992	13,892	240	5
13	4,608	96	524,160	13,568	188	3
12	3,488	64	261,120	6,452	128	1

Joint effort with large group of renowned researchers in CBC (D. J. Bernstein, T. Lange, N. Sendrier, T. Chou etc.)

19 people, 13 institutions, 8 countries

Classic McEliece parameters (bytes):

m	n	w	PK Size	SK Size	Ciph Size	Security
13	8,192	128	1,357,824	14,080	240	5
13	6,960	119	1,046,739	13,908	226	5
13	6,688	128	1,044,992	13,892	240	5
13	4,608	96	524,160	13,568	188	3
12	3,488	64	261,120	6,452	128	1

Joint effort with large group of renowned researchers in CBC (D. J. Bernstein, T. Lange, N. Sendrier, T. Chou etc.)

19 people, 13 institutions, 8 countries

Classic McEliece is now a finalist in Round 3 of NIST's PQC Standardization Process and likely to become a standard in the next few months.

Classic McEliece parameters (bytes):

m	n	w	PK Size	SK Size	Ciph Size	Security
13	8,192	128	1,357,824	14,080	240	5
13	6,960	119	1,046,739	13,908	226	5
13	6,688	128	1,044,992	13,892	240	5
13	4,608	96	524,160	13,568	188	3
12	3,488	64	261,120	6,452	128	1

Joint effort with large group of renowned researchers in CBC (D. J. Bernstein, T. Lange, N. Sendrier, T. Chou etc.)

19 people, 13 institutions, 8 countries

Classic McEliece is now a finalist in Round 3 of NIST's PQC Standardization Process and likely to become a standard in the next few months.

https://classic.mceliece.org/

Inherent Aspects of Conservative CBC

Very simple description (binary objects, low-weight XOR).

Very simple description (binary objects, low-weight XOR).

Very fast implementation (encapsulation/decapsulation).

Inherent Aspects of Conservative CBC

Very simple description (binary objects, low-weight XOR).

Very fast implementation (encapsulation/decapsulation).

Very small ciphertext size.

Inherent Aspects of Conservative CBC

Very simple description (binary objects, low-weight XOR).

Very fast implementation (encapsulation/decapsulation).

Very small ciphertext size.

No decryption failures.

Inherent Aspects of Conservative CBC

Very simple description (binary objects, low-weight XOR).

Very fast implementation (encapsulation/decapsulation).

Very small ciphertext size.

No decryption failures.

Long-term static keys + easy, tight IND-CCA reduction
(Bernstein, P., 2018).

INHERENT ASPECTS OF CONSERVATIVE CBC

Very simple description (binary objects, low-weight XOR).

Very fast implementation (encapsulation/decapsulation).

Very small ciphertext size.

No decryption failures.

Long-term static keys + easy, tight IND-CCA reduction
(Bernstein, P., 2018).

40 years of security history.

Inherent Aspects of Conservative CBC

Very simple description (binary objects, low-weight XOR).

Very fast implementation (encapsulation/decapsulation).

Very small ciphertext size.

No decryption failures.

Long-term static keys + easy, tight IND-CCA reduction
(Bernstein, P., 2018).

40 years of security history.

Very large key and slow key generation.

Part IV

STRUCTURED CODES

Traditional approach at current security levels produces very large keys: several Kb to $\approx$ 1Mb.

Traditional approach at current security levels produces very large keys: several Kb to $\approx$ 1Mb.

The problem is: public key is a large matrix, size $\mathcal{O}(n^2)$.

STRUCTURED CODES

Traditional approach at current security levels produces very large keys: several Kb to $\approx$ 1Mb.

The problem is: public key is a large matrix, size $\mathcal{O}(n^2)$.

Idea: public matrix with compact description (Gaborit '05).

Traditional approach at current security levels produces very large keys: several Kb to $\approx$ 1Mb.

The problem is: public key is a large matrix, size $\mathcal{O}(n^2)$.

Idea: public matrix with compact description (Gaborit '05).

This would allow to describe public key more efficiently.

Traditional approach at current security levels produces very large keys: several Kb to $\approx$ 1Mb.

The problem is: public key is a large matrix, size $\mathcal{O}(n^2)$.

Idea: public matrix with compact description (Gaborit '05).

This would allow to describe public key more efficiently.

Need families of codes with particular automorphism group.

Quasi-Cyclic Codes (Berger, Cayrel, Gaborit, Otmani '09).

$$G = \left[\begin{array}{cc|c} 1 & & \boxed{g} \\ & 1 & \end{array}\right]$$

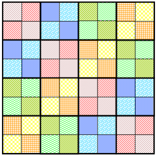
$$G = \left[\begin{array}{cc|cc|c|c|c} 1 & & & & \boxed{g_{0,0}} & \boxed{g_{0,1}} & \boxed{g_{0,2}} \\ & 1 & & & & & \\ \hline & & 1 & & \boxed{g_{1,0}} & \boxed{g_{1,1}} & \boxed{g_{1,2}} \\ & & & 1 & & & \end{array}\right]$$

Quasi-Cyclic Codes (Berger, Cayrel, Gaborit, Otmani '09).

$$G = \left[\begin{array}{cc|c} 1 & & \boxed{g} \\ & 1 & \circlearrowright \end{array}\right]$$

$$G = \left[\begin{array}{cc|cc|c|c|c} 1 & & & & \boxed{g_{0,0}} & \boxed{g_{0,1}} & \boxed{g_{0,2}} \\ & 1 & & & \circlearrowright & \circlearrowright & \circlearrowright \\ \hline & & 1 & & \boxed{g_{1,0}} & \boxed{g_{1,1}} & \boxed{g_{1,2}} \\ & & & 1 & \circlearrowright & \circlearrowright & \circlearrowright \end{array}\right]$$

Quasi-Dyadic Codes (Misoczki, Barreto '09).



Security

Several families admit QC/QD description:
GRS, Goppa, Generalized Srivastava (P. '11).

Several families admit QC/QD description:
GRS, Goppa, Generalized Srivastava (P. '11).

Problem: extra structure = extra info for attacker.

SECURITY

Several families admit QC/QD description:
GRS, Goppa, Generalized Srivastava (P. '11).

Problem: extra structure = extra info for attacker.

Critical algebraic attack.
(Faugère, Otmani, Perret, Tillich '10).

SECURITY

Several families admit QC/QD description:
GRS, Goppa, Generalized Srivastava (P. '11).

Problem: extra structure = extra info for attacker.

Critical algebraic attack.
(Faugère, Otmani, Perret, Tillich '10).

Solve system of equations from $H \cdot G^T = 0$ to recover private key.

SECURITY

Several families admit QC/QD description:
GRS, Goppa, Generalized Srivastava (P. '11).

Problem: extra structure = extra info for attacker.

Critical algebraic attack.
(Faugère, Otmani, Perret, Tillich '10).

Solve system of equations from $H \cdot G^T = 0$ to recover private key.

QC/QD + algebraic structure crucial to reduce number of unknowns of system.

Security

Several families admit QC/QD description:
GRS, Goppa, Generalized Srivastava (P. '11).

Problem: extra structure = extra info for attacker.

Critical algebraic attack.
(Faugère, Otmani, Perret, Tillich '10).

Solve system of equations from $H \cdot G^T = 0$ to recover private key.

QC/QD + algebraic structure crucial to reduce number of unknowns of system.

Original QC/QD Goppa proposals severely broken, but QD-GS perform better.

Generalized Srivastava (GS) Codes

Alternant codes with non-trivial intersection with Goppa codes.

Generalized Srivastava (GS) Codes

Alternant codes with non-trivial intersection with Goppa codes.

Admit parity-check which is superposition of s blocks of size $t \times n$.

Generalized Srivastava (GS) Codes

Alternant codes with non-trivial intersection with Goppa codes.

Admit parity-check which is superposition of s blocks of size $t \times n$.

Each block H_ℓ has ij-th element $\dfrac{z_j}{(v_j - u_\ell)^i}$ (nonzero field elements).

Generalized Srivastava (GS) Codes

Alternant codes with non-trivial intersection with Goppa codes.

Admit parity-check which is superposition of s blocks of size $t \times n$.

Each block H_ℓ has ij-th element $\dfrac{z_j}{(v_j - u_\ell)^i}$ (nonzero field elements).

If $t = 1$ this is a Goppa code.

Generalized Srivastava (GS) Codes

Alternant codes with non-trivial intersection with Goppa codes.

Admit parity-check which is superposition of s blocks of size $t \times n$.

Each block H_ℓ has ij-th element $\dfrac{z_j}{(v_j - u_\ell)^i}$ (nonzero field elements).

If $t = 1$ this is a Goppa code.

Can generate QD-GS codes using (modified) algorithm for QD Goppa.

Generalized Srivastava (GS) Codes

Alternant codes with non-trivial intersection with Goppa codes.

Admit parity-check which is superposition of s blocks of size $t \times n$.

Each block H_ℓ has ij-th element $\dfrac{z_j}{(v_j - u_\ell)^i}$ (nonzero field elements).

If $t = 1$ this is a Goppa code.

Can generate QD-GS codes using (modified) algorithm for QD Goppa.

Improved resistance against FOPT led to NIST submission DAGS.

DAGS: A QD-GS BASED KEM

Select hash functions $\mathbf{G}, \mathbf{H}, \mathbf{K}$.

DAGS: A QD-GS BASED KEM

Select hash functions $\mathbf{G}, \mathbf{H}, \mathbf{K}$.

KEY GENERATION

- Generate a QD-GS code $\mathcal{C}$.
- SK: description $(\bar{u}, \bar{v}, \bar{z})$ for $\mathcal{C}$ (in alternant form) over $\mathbb{F}_{q^m}$.
- PK: generator matrix G in systematic form for $\mathcal{C}$ over $\mathbb{F}_q$.

Select hash functions $\mathbf{G}, \mathbf{H}, \mathbf{K}$.

KEY GENERATION

- Generate a QD-GS code $\mathcal{C}$.
- SK: description $(\bar{u}, \bar{v}, \bar{z})$ for $\mathcal{C}$ (in alternant form) over $\mathbb{F}_{q^m}$.
- PK: generator matrix G in systematic form for $\mathcal{C}$ over $\mathbb{F}_q$.

ENCAPSULATION

- Choose random word $\mu \in \mathbb{F}_q^k$.
- Compute $(\rho \parallel \sigma) = \mathbf{G}(\mu)$, $d = \mathbf{H}(\mu)$.
- Generate $e \in \mathbb{F}_2^n$ of weight w from seed σ.
- Output $c = (c_0, d)$ where $c_0 = (\rho \parallel \mu)G + e$ and $K = \mathbf{K}(\mu)$.

DAGS: A QD-GS BASED KEM

Select hash functions $\mathbf{G}, \mathbf{H}, \mathbf{K}$.

KEY GENERATION

- Generate a QD-GS code $\mathcal{C}$.
- SK: description $(\bar{u}, \bar{v}, \bar{z})$ for $\mathcal{C}$ (in alternant form) over $\mathbb{F}_{q^m}$.
- PK: generator matrix G in systematic form for $\mathcal{C}$ over $\mathbb{F}_q$.

ENCAPSULATION

- Choose random word $\mu \in \mathbb{F}_q^k$.
- Compute $(\rho \parallel \sigma) = \mathbf{G}(\mu)$, $d = \mathbf{H}(\mu)$.
- Generate $e \in \mathbb{F}_2^n$ of weight w from seed σ.
- Output $c = (c_0, d)$ where $c_0 = (\rho \parallel \mu)G + e$ and $K = \mathbf{K}(\mu)$.

DECAPSULATION

- Set $(\rho' \parallel \mu'), e' = Decode(c_0)$.
- Recompute $\mathbf{G}(\mu')$, $d = \mathbf{H}(\mu')$ and e, then compare.
- Return $\perp$ if decoding fails or any check fails.
- Else return $K = \mathbf{K}(\mu')$.

About DAGS

DAGS parameters (bytes):

q	m	n	w	PK Size	SK Size	Ciph Size	Security
2^8	2	1,600	176	19,712	6,400	1,632	5
2^8	2	1,216	176	11,264	4,864	1,248	3
2^8	2	704	88	7,744	2,816	736	1

DAGS parameters (bytes):

q	m	n	w	PK Size	SK Size	Ciph Size	Security
2^8	2	1,600	176	19,712	6,400	1,632	5
2^8	2	1,216	176	11,264	4,864	1,248	3
2^8	2	704	88	7,744	2,816	736	1

Joint project with P.-L. Cayrel, C. T. Gueye, G. Banegas et al.

13 people, 7 institutions, 5 countries

About DAGS

DAGS parameters (bytes):

q	m	n	w	PK Size	SK Size	Ciph Size	Security
2^8	2	1,600	176	19,712	6,400	1,632	5
2^8	2	1,216	176	11,264	4,864	1,248	3
2^8	2	704	88	7,744	2,816	736	1

Joint project with P.-L. Cayrel, C. T. Gueye, G. Banegas et al.

13 people, 7 institutions, 5 countries

DAGS was admitted to Round 2 but didn't make it to final round due to security concerns.

(Barelli-Couvreur '18).

About DAGS

DAGS parameters (bytes):

q	m	n	w	PK Size	SK Size	Ciph Size	Security
2^8	2	1,600	176	19,712	6,400	1,632	5
2^8	2	1,216	176	11,264	4,864	1,248	3
2^8	2	704	88	7,744	2,816	736	1

Joint project with P.-L. Cayrel, C. T. Gueye, G. Banegas et al.

13 people, 7 institutions, 5 countries

DAGS was admitted to Round 2 but didn't make it to final round due to security concerns.

(Barelli-Couvreur '18).

https://www.dags-project.org/

Inherent Aspects of Structured Algebraic CBC

Potential for significant reduction in public-key size.

Potential for significant reduction in public-key size.

Very small ciphertext size.

Potential for significant reduction in public-key size.

Very small ciphertext size.

No decryption failures.

Inherent Aspects of Structured Algebraic CBC

Potential for significant reduction in public-key size.

Very small ciphertext size.

No decryption failures.

Long-term static keys + easy, tight IND-CCA reduction.

Inherent Aspects of Structured Algebraic CBC

Potential for significant reduction in public-key size.

Very small ciphertext size.

No decryption failures.

Long-term static keys + easy, tight IND-CCA reduction.

Slower arithmetic due to different basic tools (e.g. larger finite fields).

Inherent Aspects of Structured Algebraic CBC

Potential for significant reduction in public-key size.

Very small ciphertext size.

No decryption failures.

Long-term static keys + easy, tight IND-CCA reduction.

Slower arithmetic due to different basic tools (e.g. larger finite fields).

Newer approach requires long time to build confidence...

INHERENT ASPECTS OF STRUCTURED ALGEBRAIC CBC

Potential for significant reduction in public-key size.

Very small ciphertext size.

No decryption failures.

Long-term static keys + easy, tight IND-CCA reduction.

Slower arithmetic due to different basic tools (e.g. larger finite fields).

Newer approach requires long time to build confidence...

...after a few years of fixes and new attacks: keys getting bigger, confidence/interest getting smaller.

Part V

Sparse-Matrix Codes

Family of codes characterized by very sparse parity-check matrix.

.

Sparse-Matrix Codes

Family of codes characterized by very sparse parity-check matrix.

Definition (LDPC Code)

An $[n, k]$ binary linear code which admits a parity-check matrix of constant row weight $\omega \in O(1)$.

.

Sparse-Matrix Codes

Family of codes characterized by very sparse parity-check matrix.

Definition (LDPC Code)

An $[n, k]$ binary linear code which admits a parity-check matrix of constant row weight $\omega \in O(1)$.

If we write $H = (H_0 \mid H_1)$ resp. $r \times k$ and $r \times r$ then $G = (I_k \mid H_0^T H_1^{-T})$.

Sparse-Matrix Codes

Family of codes characterized by very sparse parity-check matrix.

Definition (LDPC Code)

An $[n, k]$ binary linear code which admits a parity-check matrix of constant row weight $\omega \in O(1)$.

If we write $H = (H_0 \mid H_1)$ resp. $r \times k$ and $r \times r$ then $G = (I_k \mid H_0^T H_1^{-T})$.

The non-trivial block is dense, so this is a natural choice of public key for McEliece.

Sparse-Matrix Codes

Family of codes characterized by very sparse parity-check matrix.

Definition (LDPC Code)

An $[n, k]$ binary linear code which admits a parity-check matrix of constant row weight $\omega \in O(1)$.

If we write $H = (H_0 \mid H_1)$ resp. $r \times k$ and $r \times r$ then $G = (I_k \mid H_0^T H_1^{-T})$.

The non-trivial block is dense, so this is a natural choice of public key for McEliece.

Decodable with very efficient probabilistic "bit flipping" algorithm (Gallager, '63), small decoding failure rate (DFR).

SECURITY

Distinguish public matrix $\approxeq$ look for low-weight codewords in the dual.

Security

Distinguish public matrix $\approxeq$ look for low-weight codewords in the dual.

This is also a decoding problem! So we have essentially one assumption.

SECURITY

Distinguish public matrix $\approxeq$ look for low-weight codewords in the dual.

This is also a decoding problem! So we have essentially one assumption.

Best attacks: generic “search” algorithms like Information-Set Decoding (ISD).

SECURITY

Distinguish public matrix $\approxeq$ look for low-weight codewords in the dual.

This is also a decoding problem! So we have essentially one assumption.

Best attacks: generic "search" algorithms like Information-Set Decoding (ISD).

MDPC: "relaxed" version of LDPC (Misoczki, Tillich, Sendrier and Barreto '12).

SECURITY

Distinguish public matrix $\approxeq$ look for low-weight codewords in the dual.

This is also a decoding problem! So we have essentially one assumption.

Best attacks: generic "search" algorithms like Information-Set Decoding (ISD).

MDPC: "relaxed" version of LDPC (Misoczki, Tillich, Sendrier and Barreto '12).

Change weight ω from very low (≈ 10) to "moderate" ($O(\sqrt{n})$).

Security

Distinguish public matrix $\approxeq$ look for low-weight codewords in the dual.

This is also a decoding problem! So we have essentially one assumption.

Best attacks: generic "search" algorithms like Information-Set Decoding (ISD).

MDPC: "relaxed" version of LDPC (Misoczki, Tillich, Sendrier and Barreto '12).

Change weight ω from very low (≈ 10) to "moderate" ($O(\sqrt{n})$).

Still decodable, gain in security makes up for degradation.

Structured Sparse-Matrix Codes

Using “plain” LDPC/MDPC is not practical due to long code lengths.

Structured Sparse-Matrix Codes

Using “plain” LDPC/MDPC is not practical due to long code lengths.

Possible to build QC-LDPC/MDPC codes and have compact keys.

Using “plain” LDPC/MDPC is not practical due to long code lengths.

Possible to build QC-LDPC/MDPC codes and have compact keys.

Matrices formed by circulant blocks

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{r-1} \\ a_{r-1} & a_0 & \dots & a_{r-2} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 \end{bmatrix}$$

Structured Sparse-Matrix Codes

Using "plain" LDPC/MDPC is not practical due to long code lengths.

Possible to build QC-LDPC/MDPC codes and have compact keys.

Matrices formed by circulant blocks

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{r-1} \\ a_{r-1} & a_0 & \dots & a_{r-2} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 \end{bmatrix}$$

Correspond to ideals of $\mathcal{R} = \mathbb{F}_2[x]/(x^r - 1)$: describe using ring arithmetic.

Structured Sparse-Matrix Codes

Using "plain" LDPC/MDPC is not practical due to long code lengths.

Possible to build QC-LDPC/MDPC codes and have compact keys.

Matrices formed by circulant blocks

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{r-1} \\ a_{r-1} & a_0 & \dots & a_{r-2} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 \end{bmatrix}$$

Correspond to ideals of $\mathcal{R} = \mathbb{F}_2[x]/(x^r - 1)$: describe using ring arithmetic.

Sparse-matrix codes don't possess inherent algebraic structure.

Structured Sparse-Matrix Codes

Using "plain" LDPC/MDPC is not practical due to long code lengths.

Possible to build QC-LDPC/MDPC codes and have compact keys.

Matrices formed by circulant blocks

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{r-1} \\ a_{r-1} & a_0 & \dots & a_{r-2} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 \end{bmatrix}$$

Correspond to ideals of $\mathcal{R} = \mathbb{F}_2[x]/(x^r - 1)$: describe using ring arithmetic.

Sparse-matrix codes don't possess inherent algebraic structure.

QC property alone does not provide a structural attack.

Sparse-matrix McEliece

Key Generation

- Choose h_0, h_1 in $\mathcal{R}$ of combined weight ω.
- SK: parity-check matrix formed by circulant blocks h_0, h_1.
- PK: generator matrix formed by identity and $g = h_0 h_1^{-1}$.

Sparse-matrix McEliece

Key Generation

- Choose h_0, h_1 in $\mathcal{R}$ of combined weight ω.
- SK: parity-check matrix formed by circulant blocks h_0, h_1.
- PK: generator matrix formed by identity and $g = h_0 h_1^{-1}$.

Encryption

- Take message $\mu \in \mathcal{R}$.
- Sample vectors e_0, e_1 in $\mathcal{R}$ of combined weight w.
- Output $c = (\mu + e_0, \mu \cdot g + e_1)$.

Sparse-matrix McEliece

Key Generation

- Choose h_0, h_1 in $\mathcal{R}$ of combined weight ω.
- SK: parity-check matrix formed by circulant blocks h_0, h_1.
- PK: generator matrix formed by identity and $g = h_0 h_1^{-1}$.

Encryption

- Take message $\mu \in \mathcal{R}$.
- Sample vectors e_0, e_1 in $\mathcal{R}$ of combined weight w.
- Output $c = (\mu + e_0, \mu \cdot g + e_1)$.

Decryption

- Set $(e_0, e_1) = Decode_{BitFlipping}(c)$.
- Return $\perp$ if decoding fails.
- Else recover μ (truncate).

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

Initially three variants, independently published.

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

Initially three variants, independently published.

$1, 2$: CAKE (Barreto, Gueron, Güneysu, Misoczki, P., Sendrier, Tillich, '17).

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

Initially three variants, independently published.

$1, 2$: CAKE (Barreto, Gueron, Güneysu, Misoczki, P., Sendrier, Tillich, '17).

3: Ouroboros (Deneuville, Gaborit, Zémor, '17).

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

Initially three variants, independently published.

$1, 2$: CAKE (Barreto, Gueron, Güneysu, Misoczki, P., Sendrier, Tillich, '17).

3: Ouroboros (Deneuville, Gaborit, Zémor, '17).

BIKE-1: use McEliece and non-systematic generator matrix to avoid polynomial inversion and optimize time (latency).

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

Initially three variants, independently published.

$1, 2$: CAKE (Barreto, Gueron, Güneysu, Misoczki, P., Sendrier, Tillich, '17).

3: Ouroboros (Deneuville, Gaborit, Zémor, '17).

BIKE-1: use McEliece and non-systematic generator matrix to avoid polynomial inversion and optimize time (latency).

BIKE-2: use Niederreiter and systematic parity-check to optimize space (bandwidth).

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

Initially three variants, independently published.

$1, 2$: CAKE (Barreto, Gueron, Güneysu, Misoczki, P., Sendrier, Tillich, '17).

3: Ouroboros (Deneuville, Gaborit, Zémor, '17).

BIKE-1: use McEliece and non-systematic generator matrix to avoid polynomial inversion and optimize time (latency).

BIKE-2: use Niederreiter and systematic parity-check to optimize space (bandwidth).

BIKE-3: use "noisy" decoder to have simpler security reduction.

BIKE

Suite of KEM schemes based on the bit-flipping decoder and QC-MDPC codes.

Initially three variants, independently published.

$1, 2$: CAKE (Barreto, Gueron, Güneysu, Misoczki, P., Sendrier, Tillich, '17).

3: Ouroboros (Deneuville, Gaborit, Zémor, '17).

BIKE-1: use McEliece and non-systematic generator matrix to avoid polynomial inversion and optimize time (latency).

BIKE-2: use Niederreiter and systematic parity-check to optimize space (bandwidth).

BIKE-3: use "noisy" decoder to have simpler security reduction.

In Round 3, narrowed scope to BIKE-2 as most promising.

DECODING FAILURES ARE BAD!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Decoding Failures are Bad!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Problem 1: reaction attacks (Guo, Johansson, Stankovski, '16).

Decoding Failures are Bad!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Problem 1: reaction attacks (Guo, Johansson, Stankovski, '16).

Observe decryption of several ($\approx$ 300 million) ciphertexts: analyze decoding failures to reconstruct private key (distance spectrum).

Decoding Failures are Bad!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Problem 1: reaction attacks (Guo, Johansson, Stankovski, '16).

Observe decryption of several ($\approx$ 300 million) ciphertexts: analyze decoding failures to reconstruct private key (distance spectrum).

Solution: use ephemeral keys.

Decoding Failures are Bad!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Problem 1: reaction attacks (Guo, Johansson, Stankovski, '16).

Observe decryption of several ($\approx$ 300 million) ciphertexts: analyze decoding failures to reconstruct private key (distance spectrum).

Solution: use ephemeral keys.

Problem 2: IND-CCA security.

Decoding Failures are Bad!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Problem 1: reaction attacks (Guo, Johansson, Stankovski, '16).

Observe decryption of several ($\approx$ 300 million) ciphertexts: analyze decoding failures to reconstruct private key (distance spectrum).

Solution: use ephemeral keys.

Problem 2: IND-CCA security.

IND-CCA conversions require perfect correctness or at least trivial DFR ($\approx 2^{-128}$).

Decoding Failures are Bad!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Problem 1: reaction attacks (Guo, Johansson, Stankovski, '16).

Observe decryption of several ($\approx$ 300 million) ciphertexts: analyze decoding failures to reconstruct private key (distance spectrum).

Solution: use ephemeral keys.

Problem 2: IND-CCA security.

IND-CCA conversions require perfect correctness or at least trivial DFR ($\approx 2^{-128}$).

Solution: security is formally restricted to IND-CPA security.

Decoding Failures are Bad!

"Basic" decoding algorithms have (usually) DFR around 10^{-7} to 10^{-9}.

Problem 1: reaction attacks (Guo, Johansson, Stankovski, '16).

Observe decryption of several ($\approx$ 300 million) ciphertexts: analyze decoding failures to reconstruct private key (distance spectrum).

Solution: use ephemeral keys.

Problem 2: IND-CCA security.

IND-CCA conversions require perfect correctness or at least trivial DFR ($\approx 2^{-128}$).

Solution: security is formally restricted to IND-CPA security.

Challenge: design an IND-CCA secure variant with static keys.

BIKE

Select hash functions $\mathbf{H}, \mathbf{K}, \mathbf{L}$.

BIKE

Select hash functions $\mathbf{H}, \mathbf{K}, \mathbf{L}$.

Key Generation

- Choose h_0, h_1 in $\mathcal{R}$ of combined weight w.
- SK: parity-check matrix formed by circulant blocks h_0, h_1.
- PK: parity-check matrix formed by identity and $h_1 h_0^{-1}$, plus random string σ.

BIKE

Select hash functions $\mathbf{H}, \mathbf{K}, \mathbf{L}$.

Key Generation

- Choose h_0, h_1 in $\mathcal{R}$ of combined weight w.
- SK: parity-check matrix formed by circulant blocks h_0, h_1.
- PK: parity-check matrix formed by identity and $h_1 h_0^{-1}$, plus random string σ.

Encapsulation

- Sample message $\mu \in \{0, 1\}^*$.
- Compute $(e_0, e_1) = \mathbf{H}(\mu)$ in $\mathcal{R}$ of combined weight t.
- Output $c = (c_0, c_1) = (e_0 + e_1 h, \mu \oplus \mathbf{L}(e_0, e_1))$ and $K = \mathbf{K}(\mu, c)$.

BIKE

Select hash functions $\mathbf{H}, \mathbf{K}, \mathbf{L}$.

Key Generation

- Choose h_0, h_1 in $\mathcal{R}$ of combined weight w.
- SK: parity-check matrix formed by circulant blocks h_0, h_1.
- PK: parity-check matrix formed by identity and $h_1 h_0^{-1}$, plus random string σ.

Encapsulation

- Sample message $\mu \in \{0, 1\}^*$.
- Compute $(e_0, e_1) = \mathbf{H}(\mu)$ in $\mathcal{R}$ of combined weight t.
- Output $c = (c_0, c_1) = (e_0 + e_1 h, \mu \oplus \mathbf{L}(e_0, e_1))$ and $K = \mathbf{K}(\mu, c)$.

Decapsulation

- Set $(e'_0, e'_1) = \mathit{Decode}_{\mathit{BitFlipping}}(c_0 h_0)$, or $(e'_0, e'_1) = (0, 0)$ if $\perp$.
- Compute $\mu' = c_1 \oplus \mathbf{L}(e'_0, e'_1)$.
- If $(e'_0, e'_1) = \mathbf{H}(\mu')$ then $K = \mathbf{K}(\mu, c)$, else $K = \mathbf{K}(\sigma, c)$.

BIKE parameters (bytes):

r	w	t	PK Size	SK Size	Ciph Size	Security
40,973	274	264	5,122	580	5,154	5
24,659	206	199	3,083	419	3,115	3
12,323	142	134	1,541	281	1,573	1

About BIKE

BIKE parameters (bytes):

r	w	t	PK Size	SK Size	Ciph Size	Security
40,973	274	264	5,122	580	5,154	5
24,659	206	199	3,083	419	3,115	3
12,323	142	134	1,541	281	1,573	1

Joint project with P. Barreto, R. Misoczki, N. Sendrier et al.

17 people, 11 institutions, 4 countries

About BIKE

BIKE parameters (bytes):

r	w	t	PK Size	SK Size	Ciph Size	Security
40,973	274	264	5,122	580	5,154	5
24,659	206	199	3,083	419	3,115	3
12,323	142	134	1,541	281	1,573	1

Joint project with P. Barreto, R. Misoczki, N. Sendrier et al.

17 people, 11 institutions, 4 countries

BIKE was admitted to Round 3 as an alternate finalist for possible future standardization.

About BIKE

BIKE parameters (bytes):

r	w	t	PK Size	SK Size	Ciph Size	Security
40,973	274	264	5,122	580	5,154	5
24,659	206	199	3,083	419	3,115	3
12,323	142	134	1,541	281	1,573	1

Joint project with P. Barreto, R. Misoczki, N. Sendrier et al.

17 people, 11 institutions, 4 countries

BIKE was admitted to Round 3 as an alternate finalist for possible future standardization.

https://www.bikesuite.org/

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Not subject to algebraic attacks.

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Not subject to algebraic attacks.

Efficient arithmetic and implementation.

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Not subject to algebraic attacks.

Efficient arithmetic and implementation.

Ideal for ephemeral use (e.g. TLS).

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Not subject to algebraic attacks.

Efficient arithmetic and implementation.

Ideal for ephemeral use (e.g. TLS).

Larger ciphertext size.

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Not subject to algebraic attacks.

Efficient arithmetic and implementation.

Ideal for ephemeral use (e.g. TLS).

Larger ciphertext size.

Decryption failures add extra difficulty.

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Not subject to algebraic attacks.

Efficient arithmetic and implementation.

Ideal for ephemeral use (e.g. TLS).

Larger ciphertext size.

Decryption failures add extra difficulty.

Newer decoders can reach target DFR (e.g. 2^{-128}) but only experimentally (extrapolations).

Inherent Aspects of Sparse-Matrix CBC

Very significant reduction in public-key size.

Not subject to algebraic attacks.

Efficient arithmetic and implementation.

Ideal for ephemeral use (e.g. TLS).

Larger ciphertext size.

Decryption failures add extra difficulty.

Newer decoders can reach target DFR (e.g. 2^{-128}) but only experimentally (extrapolations).

Scheme is "CCA-ready" but without a formal claim and not recommending static keys.

Part VI

CONCLUSIONS

Conclusions

Code-based cryptography is prominent candidate for standardization.

CONCLUSIONS

Code-based cryptography is prominent candidate for standardization.

Several distinctive strengths (and few well-known drawbacks).

Conclusions

Code-based cryptography is prominent candidate for standardization.

Several distinctive strengths (and few well-known drawbacks).

Suitable for KEM: key exchange + encryption.

CONCLUSIONS

Code-based cryptography is prominent candidate for standardization.

Several distinctive strengths (and few well-known drawbacks).

Suitable for KEM: key exchange + encryption.

NIST has identified two macro-areas, each with their own pros/cons:

CONCLUSIONS

Code-based cryptography is prominent candidate for standardization.

Several distinctive strengths (and few well-known drawbacks).

Suitable for KEM: key exchange + encryption.

NIST has identified two macro-areas, each with their own pros/cons:

- Conservative (binary Goppa, no structure)

CONCLUSIONS

Code-based cryptography is prominent candidate for standardization.

Several distinctive strengths (and few well-known drawbacks).

Suitable for KEM: key exchange + encryption.

NIST has identified two macro-areas, each with their own pros/cons:

- Conservative (binary Goppa, no structure)
- Low Hamming (LDPC/MDPC & HQC, QC structure)

CONCLUSIONS

Code-based cryptography is prominent candidate for standardization.

Several distinctive strengths (and few well-known drawbacks).

Suitable for KEM: key exchange + encryption.

NIST has identified two macro-areas, each with their own pros/cons:

- Conservative (binary Goppa, no structure)
- Low Hamming (LDPC/MDPC & HQC, QC structure)

Round 3: final protocol refinements, re-parametrizations, new/improved implementations.

CONCLUSIONS

Code-based cryptography is prominent candidate for standardization.

Several distinctive strengths (and few well-known drawbacks).

Suitable for KEM: key exchange + encryption.

NIST has identified two macro-areas, each with their own pros/cons:

- Conservative (binary Goppa, no structure)
- Low Hamming (LDPC/MDPC & HQC, QC structure)

Round 3: final protocol refinements, re-parametrizations, new/improved implementations.

Todo: signatures!

What About Signatures?

Long time standing open problem.

What About Signatures?

Long time standing open problem.

4 NIST submissions, 0 survivors: all withdrawn/broken.

What About Signatures?

Long time standing open problem.

4 NIST submissions, 0 survivors: all withdrawn/broken.

Inherent problem with Hamming metric.

What About Signatures?

Long time standing open problem.

4 NIST submissions, 0 survivors: all withdrawn/broken.

Inherent problem with Hamming metric.

Hash-and-sign: disjoint "balls" don't cover space.

What About Signatures?

Long time standing open problem.

4 NIST submissions, 0 survivors: all withdrawn/broken.

Inherent problem with Hamming metric.

Hash-and-sign: disjoint “balls” don’t cover space.

Fiat-Shamir: “sparse” vectors are prone to information leakage.

What About Signatures?

Long time standing open problem.

4 NIST submissions, 0 survivors: all withdrawn/broken.

Inherent problem with Hamming metric.

Hash-and-sign: disjoint “balls” don’t cover space.

Fiat-Shamir: “sparse” vectors are prone to information leakage.

Out of scope of this presentation (but happy to discuss!).

Follow the NIST Competition

FAU has been funded by NIST for PQC project.

FOLLOW THE NIST COMPETITION

FAU has been funded by NIST for PQC project.

Detailed competition wiki/database *https://pqc-wiki.fau.edu*.

Follow the NIST Competition

FAU has been funded by NIST for PQC project.

Detailed competition wiki/database *https://pqc-wiki.fau.edu*.

Includes parameters, sizes, security assumptions etc. + challenges.

FOLLOW THE NIST COMPETITION

FAU has been funded by NIST for PQC project.

Detailed competition wiki/database *https://pqc-wiki.fau.edu*.

Includes parameters, sizes, security assumptions etc. + challenges.

"Living" resource with external contributions.

Grazie per l'attenzione!