An introduction to Post-Quantum Cryptography (PQC)

Jean-Christophe Deneuville

<jean-christophe.deneuville@enac.fr>

Fall 2020

TLS-SEC

Outline

Clarification

What are the alternatives to classical cryptography in presence of an adversary equipped with a large scale quantum computer?

Clarification

What are the alternatives to classical cryptography in presence of an adversary equipped with a large scale quantum computer?

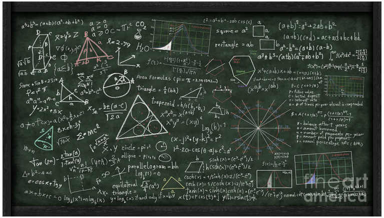
- Quantum Key Exchange (out of the scope of this course)

Clarification

What are the alternatives to classical cryptography in presence of an adversary equipped with a large scale quantum computer?

- Quantum Key Exchange (out of the scope of this course)

- Post-Quantum Cryptography



Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography
- Hash (function) - based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography
- Hash (function) - based cryptography
- Multivariate (polynomials) - based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography
- Hash (function) - based cryptography
- Multivariate (polynomials) - based cryptography
- Isogeny (over elliptic curves) - based cryptography

NIST PQC standardization process

NIST **N**ational **I**nstitute of **S**tandards and **T**echnologies

NIST PQC standardization process

National **I**nstitute of **S**tandards and **T**echnologies

- 3rd call for standardization
- Asks for post-quantum cryptographic algorithms
- 3 categories :
  - Encryption
  - Key exchange
  - Signature
- Many candidates:
  - Error correcting codes,
  - Lattices,
  - Multivariate,
  - Hash functions,
  - ...

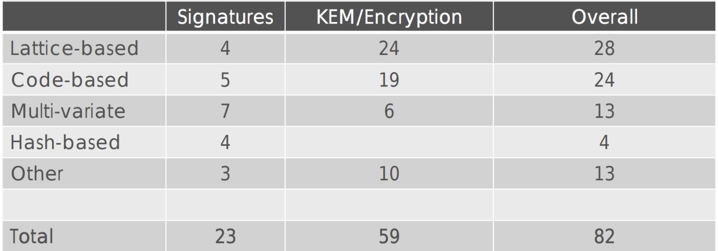
NIST PQC standardization process

NIST **N**ational **I**nstitute of **S**tandards and **T**echnologies

- 3rd call for standardization
- Asks for post-quantum cryptographic algorithms
- 3 categories :
    - Encryption
    - Key exchange
    - Signature
- Many candidates:
    - Error correcting codes,
    - Lattices,
    - Multivariate,
    - Hash functions,
    - ...

- November 2016: anouncement
- November 2017: submission deadline (82 submissions)
- December 2017: 1st round: 69 submissions
- April 2018: 1st standardization conference
- January 2019: 2nd round: 26 candidates
- March 2019: tweaks for 2nd round
- August 2019: 2nd standardization conference
- July 2020: 3rd round: 7 finalists, 8 alternates
- 2022 → 2024: draft standards ready

Hot topic!



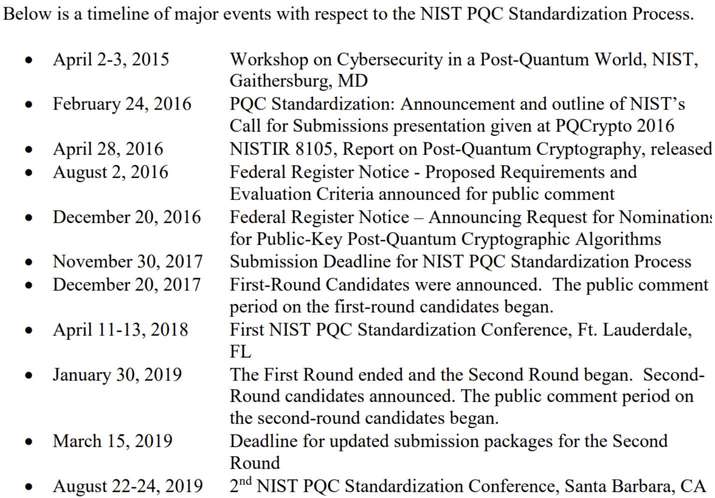
	Signatures	KEM/Encryption	Overall
Lattice-based	4	24	28
Code-based	5	19	24
Multi-variate	7	6	13
Hash-based	4		4
Other	3	10	13
Total	23	59	82

Submissions available at:

- `https://csrc.nist.gov/Projects/post-quantum-cryptography/Post-Quantum-Cryptography-Standardization`
- `https://www.safecrypto.eu/pqclounge/`

source:
Dustin Moody, NIST

Hot topic!



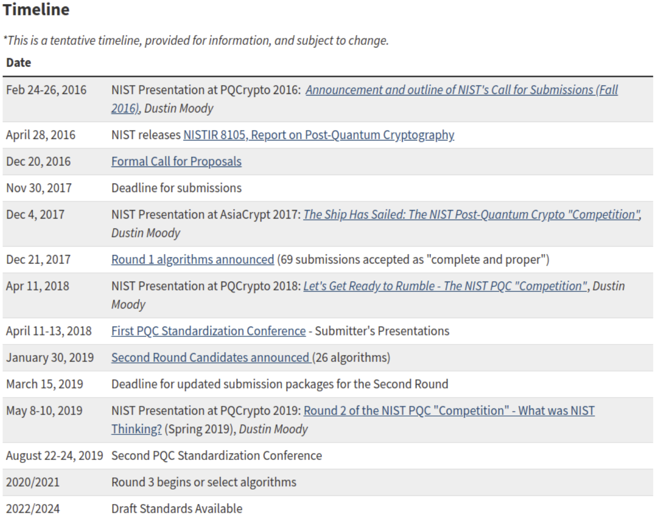
Below is a timeline of major events with respect to the NIST PQC Standardization Process.

April 2-3, 2015	Workshop on Cybersecurity in a Post-Quantum World, NIST, Gaithersburg, MD
February 24, 2016	PQC Standardization: Announcement and outline of NIST's Call for Submissions presentation given at PQCrypto 2016
April 28, 2016	NISTIR 8105, Report on Post-Quantum Cryptography, released
August 2, 2016	Federal Register Notice - Proposed Requirements and Evaluation Criteria announced for public comment
December 20, 2016	Federal Register Notice – Announcing Request for Nomination for Public-Key Post-Quantum Cryptographic Algorithms
November 30, 2017	Submission Deadline for NIST PQC Standardization Process
December 20, 2017	First-Round Candidates were announced. The public comment period on the first-round candidates began.
April 11-13, 2018	First NIST PQC Standardization Conference, Ft. Lauderdale, FL
January 30, 2019	The First Round ended and the Second Round began. Second-Round candidates announced. The public comment period on the second-round candidates began.
March 15, 2019	Deadline for updated submission packages for the Second Round
August 22-24, 2019	2nd NIST PQC Standardization Conference, Santa Barbara, CA

source:

NIST IR 8240

Hot topic!



Timeline

**This is a tentative timeline, provided for information, and subject to change.*

Date	
Feb 24-26, 2016	NIST Presentation at PQCrypto 2016: *Announcement and outline of NIST's Call for Submissions (Fall 2016), Dustin Moody*
April 28, 2016	NIST releases NISTIR 8105, Report on Post-Quantum Cryptography
Dec 20, 2016	Formal Call for Proposals
Nov 30, 2017	Deadline for submissions
Dec 4, 2017	NIST Presentation at AsiaCrypt 2017: *The Ship Has Sailed: The NIST Post-Quantum Crypto "Competition", Dustin Moody*
Dec 21, 2017	Round 1 algorithms announced (69 submissions accepted as "complete and proper")
Apr 11, 2018	NIST Presentation at PQCrypto 2018: *Let's Get Ready to Rumble - The NIST PQC "Competition", Dustin Moody*
April 11-13, 2018	First PQC Standardization Conference - Submitter's Presentations
January 30, 2019	Second Round Candidates announced (26 algorithms)
March 15, 2019	Deadline for updated submission packages for the Second Round
May 8-10, 2019	NIST Presentation at PQCrypto 2019: Round 2 of the NIST PQC "Competition" - What was NIST Thinking? (Spring 2019), *Dustin Moody*
August 22-24, 2019	Second PQC Standardization Conference
2020/2021	Round 3 begins or select algorithms
2022/2024	Draft Standards Available

Outline

6 Post-quantum cryptography
- Lattice-based cryptography
- Code-based cryptography
- Hash-based cryptography

Some background

Recalls on linear algebra

Some background

Recalls on linear algebra

- Vector space, norm, linearly independent vectors, matrix, multiplication

Some background

Recalls on linear algebra

- Vector space, norm, linearly independent vectors, matrix, multiplication
- Determinant, invertible matrix

Some background

Recalls on linear algebra

- Vector space, norm, linearly independent vectors, matrix, multiplication
- Determinant, invertible matrix
- Polynomials, quotient ring, relationship with matrices

Definitions

Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Definitions

Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Vocabulary

- rank n **(main security parameter)**
- dimension m **($m = \mathcal{O}(n \cdot \log n)$)**
- basis $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n)$ **(multiple basis)**

Definitions

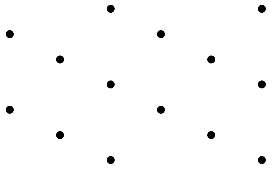
Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Vocabulary

- rank n **(main security parameter)**
- dimension m **($m = \mathcal{O}(n \cdot \log n)$)**
- basis $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n)$ **(multiple basis)**



Definitions

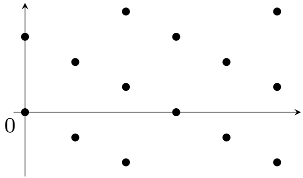
Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Vocabulary

- rank n **(main security parameter)**
- dimension m **($m = \mathcal{O}(n \cdot \log n)$)**
- basis $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n)$ **(multiple basis)**



Definitions

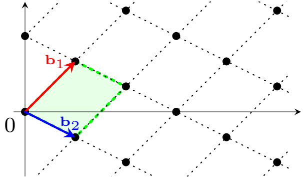
Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Vocabulary

- rank n **(main security parameter)**
- dimension m **($m = \mathcal{O}(n \cdot \log n)$)**
- basis $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n)$ **(multiple basis)**



Definitions

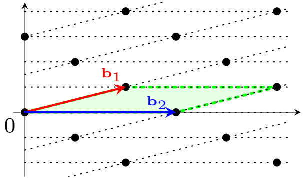
Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Vocabulary

- rank n **(main security parameter)**
- dimension m **($m = \mathcal{O}(n \cdot \log n)$)**
- basis $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n)$ **(multiple basis)**



Definitions

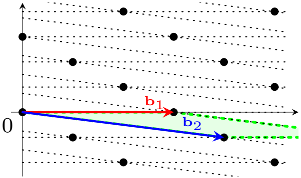
Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Vocabulary

- rank n **(main security parameter)**
- dimension m **($m = \mathcal{O}(n \cdot \log n)$)**
- basis $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n)$ **(multiple basis)**



Definitions

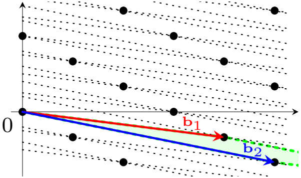
Lattice

An m-dimensional lattice is a discrete subgroup of $\mathbb{R}^m$. Formally, if $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice $\Lambda(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is the set

$$\Lambda = \{\sum_{i=1}^{n} x_i \mathbf{b}_i ; x_i \in \mathbb{Z}\} \subset \mathbb{R}^m$$

Vocabulary

- rank n **(main security parameter)**
- dimension m **($m = \mathcal{O}(n \cdot \log n)$)**
- basis $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n)$ **(multiple basis)**

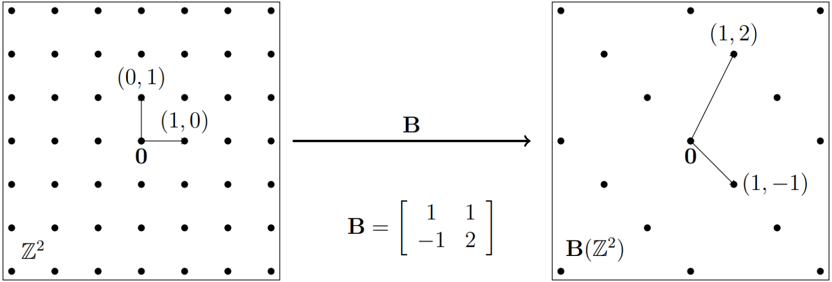


Lattice examples

The simplest lattice in dimension 2: $\mathbb{Z}^2$, and a twisted version of it.

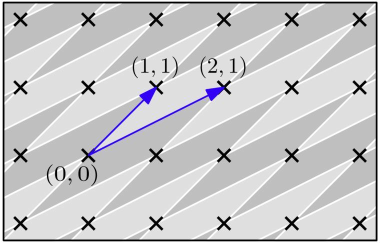
Lattice examples

The simplest lattice in dimension 2: $\mathbb{Z}^2$, and a twisted version of it.



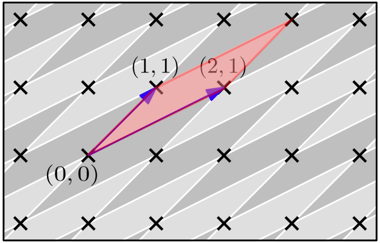
Lattice, Span, Fundamental Parallelepiped

Lattices and spans should not be confused.



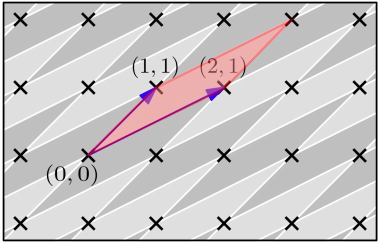
Lattice, Span, Fundamental Parallelepiped

Lattices and spans should not be confused.



Lattice, Span, Fundamental Parallelepiped

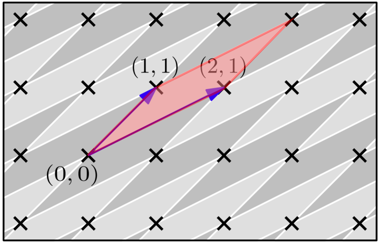
Lattices and spans should not be confused.



Basis vectors define the *fundamental parallelepiped*. (in red above)

Lattice, Span, Fundamental Parallelepiped

Lattices and spans should not be confused.



Basis vectors define the *fundamental parallelepiped*. (in red above)
The volume of this parallelepiped is called the *volume* or *determinant* of the lattice.

Matrix Representation and q-ary Lattices

Matrix Representation

Given $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n) \in \mathbb{Z}^{m \times n}$, the lattice generated by $\mathbf{B}$ is

$$\Lambda(\mathbf{B}) = \{\mathbf{B} \cdot \mathbf{x} ; \mathbf{x} \in \mathbb{Z}^n\}$$

Matrix Representation and q-ary Lattices

Matrix Representation

Given $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n) \in \mathbb{Z}^{m \times n}$, the lattice generated by $\mathbf{B}$ is

$$\Lambda(\mathbf{B}) = \{\mathbf{B} \cdot \mathbf{x}; \mathbf{x} \in \mathbb{Z}^n\}$$

q-ary Lattices

Let $\mathbf{B} = (\mathbf{b}_1 | \cdots | \mathbf{b}_n) \in \mathbb{Z}_q^{m \times n}$ for some prime q, and let

$$\Lambda_q(\mathbf{B}) = \{\mathbf{B} \cdot \mathbf{x} \mod q : \mathbf{x} \in \mathbb{Z}^n\}, \text{ and}$$

$$\Lambda_q^{\perp}(\mathbf{B}) = \{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y}^t \mathbf{B} = \mathbf{0} \mod q\}.$$

Unimodular matrices and lattice bases

- A (square) matrix is said *unimodular* if its determinant is ± 1.

Unimodular matrices and lattice bases

- A (square) matrix is said *unimodular* if its determinant is ± 1.
- If $\mathbf{B}' = \mathbf{B}\mathbf{U}$ for some unimodular matrix $\mathbf{U}$, then $\mathcal{L}\left(\mathbf{B}'\right) = \mathcal{L}\left(\mathbf{B}\right)$.

Unimodular matrices and lattice bases

- A (square) matrix is said *unimodular* if its determinant is ± 1.

- If $\mathbf{B}' = \mathbf{B}\mathbf{U}$ for some unimodular matrix $\mathbf{U}$, then $\mathcal{L}(\mathbf{B}') = \mathcal{L}(\mathbf{B})$.

  $\rightarrow$ exercise: prove it?

Unimodular matrices and lattice bases

- A (square) matrix is said *unimodular* if its determinant is ± 1.
- If $\mathbf{B}' = \mathbf{B}\mathbf{U}$ for some unimodular matrix $\mathbf{U}$, then $\mathcal{L}(\mathbf{B}') = \mathcal{L}(\mathbf{B})$.

  $\rightarrow$ exercise: prove it?
- For $n \geq 2$, any n-dimensional lattice has infinitely many bases.

Unimodular matrices and lattice bases

- A (square) matrix is said *unimodular* if its determinant is ± 1.
- If $\mathbf{B}' = \mathbf{B}\mathbf{U}$ for some unimodular matrix $\mathbf{U}$, then $\mathcal{L}\left(\mathbf{B}'\right) = \mathcal{L}\left(\mathbf{B}\right)$.

  → exercise: prove it?
- For $n \geq 2$, any n-dimensional lattice has infinitely many bases.

  → exercise: give intuition for n=2?

Successive minima

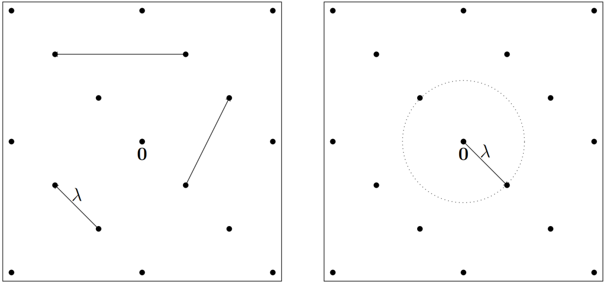
For any lattice $\mathcal{L}$, the minimum distance of $\mathcal{L}$, denoted $\lambda_1\left(\mathcal{L}\right)$ is the smallest distance between any two distinct lattice points:

$$\lambda_1\left(\mathcal{L}\right) = \inf\left\{\|\mathbf{x} - \mathbf{y}\|, \mathbf{x} \neq \mathbf{y} \in \mathcal{L}\right\}.$$

Successive minima

For any lattice $\mathcal{L}$, the minimum distance of $\mathcal{L}$, denoted $\lambda_1(\mathcal{L})$ is the smallest distance between any two distinct lattice points:

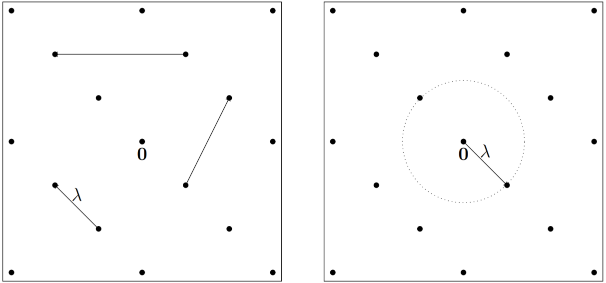
$$\lambda_1(\mathcal{L}) = \inf\{\|\mathbf{x}-\mathbf{y}\|, \mathbf{x} \neq \mathbf{y} \in \mathcal{L}\}.$$



Successive minima

For any lattice $\mathcal{L}$, the minimum distance of $\mathcal{L}$, denoted $\lambda_1(\mathcal{L})$ is the smallest distance between any two distinct lattice points:

$$\lambda_1(\mathcal{L}) = \inf\left\{\|\mathbf{x}-\mathbf{y}\|, \mathbf{x} \neq \mathbf{y} \in \mathcal{L}\right\}.$$



Equivalently, $\lambda_1(\mathcal{L})$ is the length of the shortest vector in $\mathcal{L}$.

Successive minima

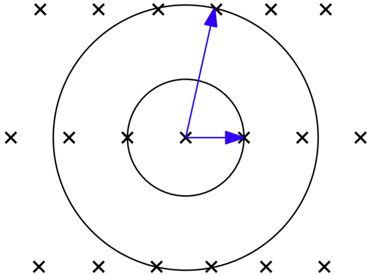
Alternatively, the minimum distance (or first minimum) of lattice $\mathcal{L}$ can be defined as the radius of the smallest ball containing a non-zero lattice point.

Successive minima

Alternatively, the minimum distance (or first minimum) of lattice $\mathcal{L}$ can be defined as the radius of the smallest ball containing a non-zero lattice point.
This definition is easily generalized to define a sequence of parameters $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$, called the successive minima of the $\mathcal{L}$

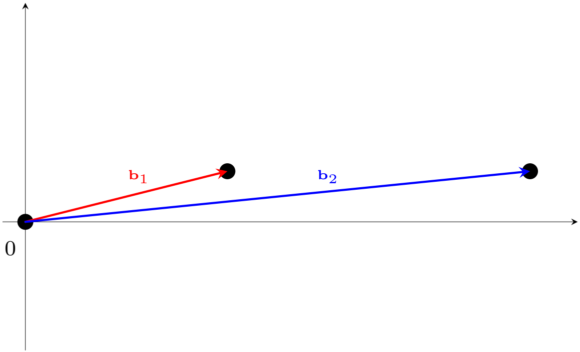
Successive minima

Alternatively, the minimum distance (or first minimum) of lattice $\mathcal{L}$ can be defined as the radius of the smallest ball containing a non-zero lattice point.
This definition is easily generalized to define a sequence of parameters $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$, called the successive minima of the $\mathcal{L}$

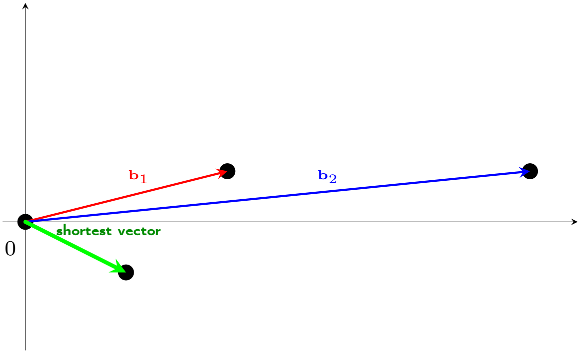


Hard problems: the Shortest Vector Problem

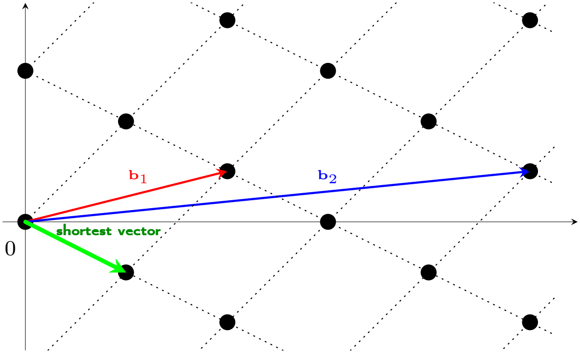
Hard problems: SVP example



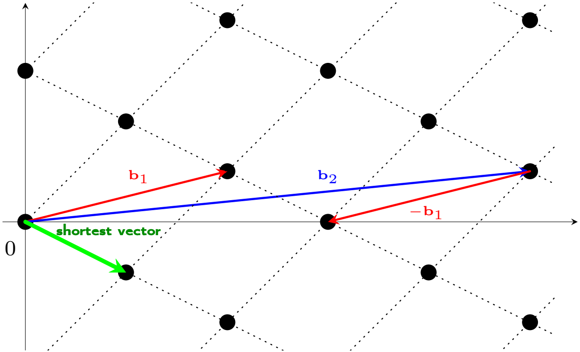
Hard problems: SVP example



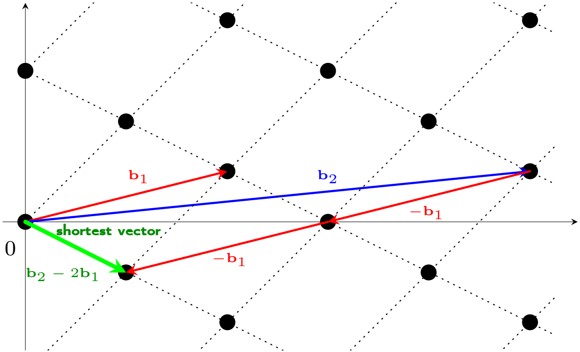
Hard problems: SVP example



Hard problems: SVP example

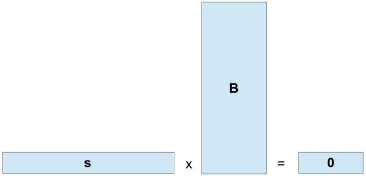


Hard problems: SVP example



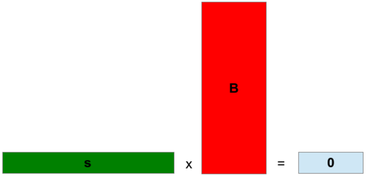
Hard problems: the Small Integer Solution

Given $\mathbf{B} \overset{\$}{\leftarrow} \mathbb{Z}_q^{m \times n}$, find “small” $\mathbf{s} \in \mathbb{Z}^m$ such that $\mathbf{s}^t\mathbf{B} = \mathbf{0} \mod q$



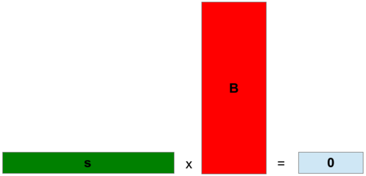
Hard problems: the Small Integer Solution

Given $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, find “small” $\mathbf{s} \in \mathbb{Z}^m$ such that $\mathbf{s}^t\mathbf{B} = \mathbf{0} \mod q$



Hard problems: the Small Integer Solution

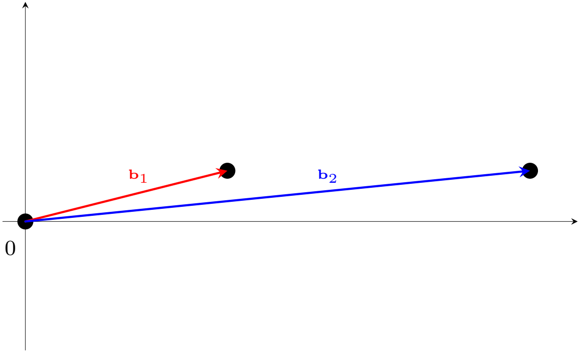
Given $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, find "small" $\mathbf{s} \in \mathbb{Z}^m$ such that $\mathbf{s}^t\mathbf{B} = \mathbf{0} \mod q$



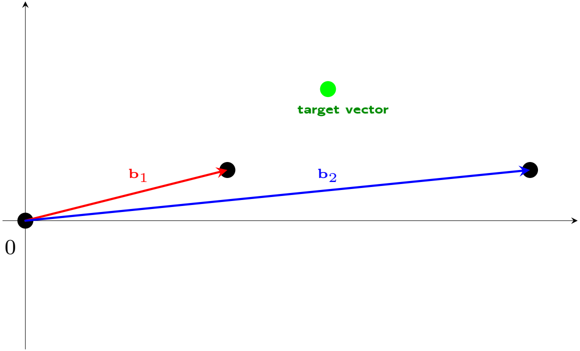
Relationship to Lattices

Solving **SIS** in random lattices $\mathbf{B}$ is "close" to solving **SVP** in $\Lambda_q^{\perp}(\mathbf{B})$

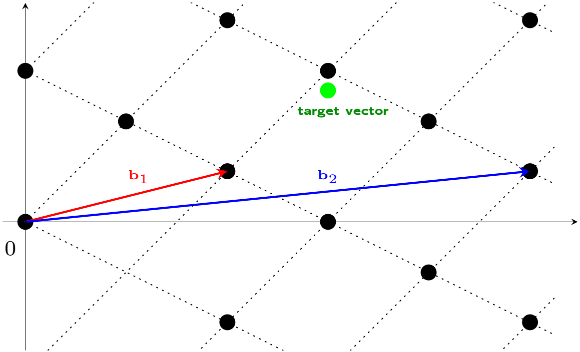
Hard problems: the Closest Vector Problem



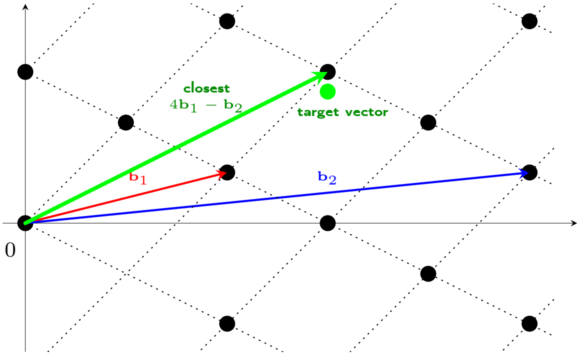
Hard problems: the Closest Vector Problem



Hard problems: the Closest Vector Problem



Hard problems: the Closest Vector Problem



Hard problems: the Learning with Errors

The Learning with Errors (LWE) problem was defined by Regev.
Given $(\mathbf{A}, \mathbf{c})$ with $\mathbf{c} \in \mathbb{Z}_q^m$, $\mathbf{A} \in \mathbb{Z}_q^{mn}$, $\mathbf{s} \in \mathbb{Z}_q^n$ and small $\mathbf{e} \in \mathbb{Z}^m$ is

$$\begin{pmatrix} \\ \\ \mathbf{c} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} \leftarrow & n & \rightarrow \\ & & \\ & \mathbf{A} & \\ & & \\ & & \end{pmatrix} \cdot \begin{pmatrix} \mathbf{s} \end{pmatrix} + \begin{pmatrix} \\ \\ \mathbf{e} \\ \\ \\ \end{pmatrix}$$

or $\mathbf{c} \leftarrow_{\$} \mathcal{U}\left(\mathbb{Z}_q^m\right)$.

Relation to other problems

Solving LWE in random lattices is close to to solving CVP in $\Lambda_q(\mathbf{B})$.

Lattice problems

Idea behind lattice-based cryptography: these problems are

Lattice problems

Idea behind lattice-based cryptography: these problems are

- hard given a "bad" basis (constituted of long and almost parallel vectors) $\rightarrow$ pk

Lattice problems

Idea behind lattice-based cryptography: these problems are

- hard given a "bad" basis (constituted of long and almost parallel vectors) → pk
- (yet it is easy to verify that a candidate vector is a solution to SVP or CVP with pk).

Lattice problems

Idea behind lattice-based cryptography: these problems are

- hard given a “bad” basis (constituted of long and almost parallel vectors) → pk
- (yet it is easy to verify that a candidate vector is a solution to SVP or CVP with pk).
- easy given a “good” basis (of short and almost orthogonal vectors) → sk

Lattice problems

Idea behind lattice-based cryptography: these problems are

- hard given a “bad” basis (constituted of long and almost parallel vectors) $\rightarrow$ pk
- (yet it is easy to verify that a candidate vector is a solution to SVP or CVP with pk).
- easy given a “good” basis (of short and almost orthogonal vectors) $\rightarrow$ sk

All these problems do not seem hard in dimension 2... Lattice cryptography operates in dimension ≥ 512.

Lattice problems

Idea behind lattice-based cryptography: these problems are

- hard given a "bad" basis (constituted of long and almost parallel vectors) $\rightarrow$ pk
- (yet it is easy to verify that a candidate vector is a solution to SVP or CVP with pk).
- easy given a "good" basis (of short and almost orthogonal vectors) $\rightarrow$ sk

All these problems do not seem hard in dimension 2... Lattice cryptography operates in dimension ≥ 512.

Question: how hard is it to obtain a good basis given a bad basis?

Good basis: optimal goals

How orthogonal can a basis be?

How short can a vector be?

Good basis: optimal goals

How orthogonal can a basis be?

$$\delta\left(\mathcal{L}\right) = \frac{\prod_{i=1}^{n} \|\mathbf{b}_i\|}{\det\left(\mathcal{L}\right) = \sqrt{\det\left(\mathbf{B}^\top \mathbf{B}\right)}} \geq 1$$

How short can a vector be?

Good basis: optimal goals

How orthogonal can a basis be?

$$\delta\left(\mathcal{L}\right) = \frac{\prod_{i=1}^{n} \left\|\mathbf{b}_i\right\|}{\det\left(\mathcal{L}\right) = \sqrt{\det\left(\mathbf{B}^\top \mathbf{B}\right)}} \geq 1$$

Notice that equality holds $\Longleftrightarrow$ basis is orthogonal.

How short can a vector be?

Good basis: optimal goals

How orthogonal can a basis be?

$$\delta\left(\mathcal{L}\right) = \frac{\prod_{i=1}^{n} \|\mathbf{b}_i\|}{\det\left(\mathcal{L}\right) = \sqrt{\det\left(\mathbf{B}^\top \mathbf{B}\right)}} \geq 1$$

Notice that equality holds $\iff$ basis is orthogonal.

How short can a vector be?

$$\lambda_1\left(\mathcal{L}\right) \approx \frac{\Gamma\left(n/2+1\right)^{1/n}}{\sqrt{\pi}} \cdot \det\left(\mathcal{L}\right)^{1/n} \approx \sqrt{\frac{n}{2\pi e}} \cdot \det\left(\mathcal{L}\right)^{1/n}$$

Good basis: optimal goals

How orthogonal can a basis be?

$$\delta\left(\mathcal{L}\right) = \frac{\prod_{i=1}^{n} \left\|\mathbf{b}_i\right\|}{\det\left(\mathcal{L}\right) = \sqrt{\det\left(\mathbf{B}^\top \mathbf{B}\right)}} \geq 1$$

Notice that equality holds $\iff$ basis is orthogonal.

How short can a vector be?

$$\lambda_1\left(\mathcal{L}\right) \approx \frac{\Gamma\left(n/2+1\right)^{1/n}}{\sqrt{\pi}} \cdot \det\left(\mathcal{L}\right)^{1/n} \approx \sqrt{\frac{n}{2\pi e}} \cdot \det\left(\mathcal{L}\right)^{1/n}$$

Gaussian heuristic predicts the length of the shortest vector in a random lattice.

Bad to good basis: lattice reduction

Given a bad basis $\mathcal{B}$, find linear combinations of its vector to obtain a reduced and almost orthogonal good basis $\mathcal{B}'$.

Bad to good basis: lattice reduction

Given a bad basis $\mathcal{B}$, find linear combinations of its vector to obtain a reduced and almost orthogonal good basis $\mathcal{B}'$.
First idea: Gram-Schmidt performs basis orthogonalization!

Bad to good basis: lattice reduction

Given a bad basis $\mathcal{B}$, find linear combinations of its vector to obtain a reduced and almost orthogonal good basis $\mathcal{B}'$.

First idea: Gram-Schmidt performs basis orthogonalization!

→ right, but the resulting set of vectors no longer spans the same lattice.

Bad to good basis: lattice reduction

Given a bad basis $\mathcal{B}$, find linear combinations of its vector to obtain a reduced and almost orthogonal good basis $\mathcal{B}'$.

First idea: Gram-Schmidt performs basis orthogonalization!

→ right, but the resulting set of vectors no longer spans the same lattice. Why ?

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

```
1 $\mathbf{b}_0^* \leftarrow \mathbf{b}_0$
```

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

```
1  $\mathbf{b}_0^* \leftarrow \mathbf{b}_0$
2  for $i$ from $1$ to $n-1$, do
```

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

```
$\mathbf{b}_0^* \leftarrow \mathbf{b}_0$
for $i$ from $1$ to $n-1$, do
    $\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=0}^{i-1} \frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \mathbf{b}_j^*$
```

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

1. $\mathbf{b}_0^* \leftarrow \mathbf{b}_0$
2. for i from 1 to $n-1$, do
3. $\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=0}^{i-1} \frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \mathbf{b}_j^*$

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

```
1 $\mathbf{b}_0^* \leftarrow \mathbf{b}_0$
2 for $i$ from $1$ to $n-1$, do
3         $\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=0}^{i-1} \frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \mathbf{b}_j^*$
```

Not an integer!

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

```
1 $\mathbf{b}_0^* \leftarrow \mathbf{b}_0$
2 for $i$ from $1$ to $n-1$, do
3     $\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=0}^{i-1} \frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \mathbf{b}_j^*$
```

Not an integer!

LLL [LLL82] solution:

Best known attacks: lattice reduction

Gram-Schmidt algorithm:

```
1  $\mathbf{b}_0^* \leftarrow \mathbf{b}_0$
2  for $i$ from $1$ to $n-1$, do
3      $\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=0}^{i-1} \boxed{\frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}} \mathbf{b}_j^*$
```

Not an integer!

LLL [LLL82] solution:

- Replace $\frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}$ by $\left\lfloor \frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \right\rceil$, the nearest integer

LLL algorithm (1982)

```
Algorithm 2: LLL$(\mathbf{B}, \delta)$
Input: (Bad) Basis $\mathbf{B}$ of $\mathcal{L}$, reduction parameter $\delta \in ]1/4, 1[$ (default=$3/4$)
Output: $\delta$-LLL-reduced basis of $\mathcal{L}$
```

LLL algorithm (1982)

```
Algorithm 2: LLL($\mathbf{B}, \delta$)
Input: (Bad) Basis $\mathbf{B}$ of $\mathcal{L}$, reduction parameter $\delta \in ]1/4, 1[$ (default=$3/4$)
Output: $\delta$-LLL-reduced basis of $\mathcal{L}$
Compute the GS orthogonalization $\tilde{\mathbf{B}}$ of $\mathbf{B}$;
```

LLL algorithm (1982)

```
Algorithm 2: LLL($\mathbf{B}, \delta$)
Input: (Bad) Basis $\mathbf{B}$ of $\mathcal{L}$, reduction parameter $\delta \in ]1/4, 1[$ (default=3/4)
Output: $\delta$-LLL-reduced basis of $\mathcal{L}$
Compute the GS orthogonalization $\tilde{\mathbf{B}}$ of $\mathbf{B}$;
Reduction step:
```

LLL algorithm (1982)

```
Algorithm 2: LLL($\mathbf{B}, \delta$)
Input: (Bad) Basis $\mathbf{B}$ of $\mathcal{L}$, reduction parameter $\delta \in ]1/4, 1[$ (default=$3/4$)
Output: $\delta$-LLL-reduced basis of $\mathcal{L}$
Compute the GS orthogonalization $\tilde{\mathbf{B}}$ of $\mathbf{B}$;
Reduction step:
for i from 2 to $n$ do
    for j from $i-1$ down to 1 do
        $\mathbf{b}_i \leftarrow \mathbf{b}_i - \left\lceil \frac{\langle \mathbf{b}_i, \tilde{\mathbf{b}_j} \rangle}{\langle \tilde{\mathbf{b}_j}, \tilde{\mathbf{b}_j} \rangle} \right\rfloor \mathbf{b}_j$
```

LLL algorithm (1982)

Algorithm 2: LLL$(\mathbf{B}, \delta)$

Input: (Bad) Basis $\mathbf{B}$ of $\mathcal{L}$, reduction parameter $\delta \in]1/4, 1[$ (default$=3/4$)

Output: δ-LLL-reduced basis of $\mathcal{L}$

```
Compute the GS orthogonalization $\tilde{\mathbf{B}}$ of $\mathbf{B}$;
Reduction step:
for i from 2 to $n$ do
    for j from $i-1$ down to 1 do
        $\mathbf{b}_i \leftarrow \mathbf{b}_i - \left\lceil \frac{\langle \mathbf{b}_i, \tilde{\mathbf{b}_j} \rangle}{\langle \tilde{\mathbf{b}_j}, \tilde{\mathbf{b}_j} \rangle} \right\rfloor \mathbf{b}_j$
Swap step:
```

LLL algorithm (1982)

Algorithm 2: LLL$(\mathbf{B}, \delta)$

```
Input: (Bad) Basis $\mathbf{B}$ of $\mathcal{L}$, reduction parameter $\delta \in ]1/4, 1[$ (default=$3/4$)
Output: $\delta$-LLL-reduced basis of $\mathcal{L}$
1 Compute the GS orthogonalization $\tilde{\mathbf{B}}$ of $\mathbf{B}$;
  Reduction step:
2 for i from 2 to $n$ do
3     for j from $i-1$ down to 1 do
4         $\mathbf{b}_i \leftarrow \mathbf{b}_i - \left\lceil \frac{\langle \mathbf{b}_i, \tilde{\mathbf{b}_j} \rangle}{\langle \tilde{\mathbf{b}_j}, \tilde{\mathbf{b}_j} \rangle} \right\rfloor \mathbf{b}_j$
  Swap step:
5 if $\exists i$ such that $\delta \|\tilde{\mathbf{b}_i}\|^2 > \| \frac{\langle \mathbf{b}_i, \tilde{\mathbf{b}_j} \rangle}{\langle \tilde{\mathbf{b}_j}, \tilde{\mathbf{b}_j} \rangle} \tilde{\mathbf{b}_i} + \tilde{\mathbf{b}_{i+1}} \|^2$ then
6     $\mathbf{b}_i \leftrightarrow \mathbf{b}_j$, go to 1 ;
```

LLL algorithm (1982)

```
Algorithm 2: LLL($\mathbf{B}, \delta$)
Input: (Bad) Basis $\mathbf{B}$ of $\mathcal{L}$, reduction parameter $\delta \in ]1/4, 1[$ (default=$3/4$)
Output: $\delta$-LLL-reduced basis of $\mathcal{L}$
1 Compute the GS orthogonalization $\tilde{\mathbf{B}}$ of $\mathbf{B}$;
  Reduction step:
2 for i from 2 to $n$ do
3     for j from $i-1$ down to 1 do
4         $\mathbf{b}_i \leftarrow \mathbf{b}_i - \left\lceil \frac{\langle \mathbf{b}_i, \tilde{\mathbf{b}}_j \rangle}{\langle \tilde{\mathbf{b}}_j, \tilde{\mathbf{b}}_j \rangle} \right\rfloor \mathbf{b}_j$
  Swap step:
5 if $\exists i$ such that $\delta \|\tilde{\mathbf{b}}_i\|^2 > \| \frac{\langle \mathbf{b}_i, \tilde{\mathbf{b}}_j \rangle}{\langle \tilde{\mathbf{b}}_j, \tilde{\mathbf{b}}_j \rangle} \tilde{\mathbf{b}}_i + \tilde{\mathbf{b}}_{i+1} \|^2$ then
6     $\mathbf{b}_i \leftrightarrow \mathbf{b}_j$, go to 1 ;
7 return $\mathbf{B}$
```

Best known attacks: lattice reduction

LLL algorithm:

- Polynomial-time algorithm, but...
- Exponential approximation factor (the resulting basis $\mathcal{B}'$ is not that good)...

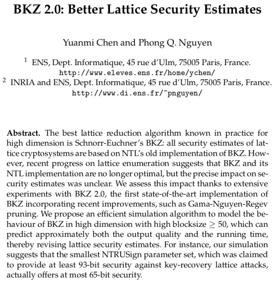
Other algorithms that trade memory/time for quality exist:

- blockwise generalization of LLL: BKZ
- Sieving
- Enumeration

They are out of the scope of this course.

Security Level

- $\delta = \left(\frac{\lambda_1}{\det(\Lambda)^{1/n}}\right)^{1/n}$ [CN11]



BKZ 2.0: Better Lattice Security Estimates

Yuanmi Chen and Phong Q. Nguyen

[1] ENS, Dept. Informatique, 45 rue d'Ulm, 75005 Paris, France.
http://www.eleves.ens.fr/home/ychen/
[2] INRIA and ENS, Dept. Informatique, 45 rue d'Ulm, 75005 Paris, France.
http://www.di.ens.fr/~pnguyen/

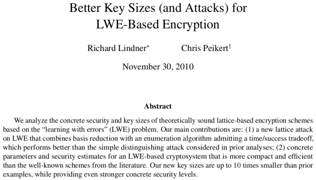
Abstract. The best lattice reduction algorithm known in practice for high dimension is Schnorr-Euchner's BKZ: all security estimates of lattice cryptosystems are based on NTL's old implementation of BKZ. However, recent progress on lattice enumeration suggests that BKZ and its NTL implementation are no longer optimal, but the precise impact on security estimates was unclear. We assess this impact thanks to extensive experiments with BKZ 2.0, the first state-of-the-art implementation of BKZ incorporating recent improvements, such as Gama-Nguyen-Regev pruning. We propose an efficient simulation algorithm to model the behaviour of BKZ in high dimension with high blocksize ≥ 50, which can predict approximately both the output quality and the running time, thereby revising lattice security estimates. For instance, our simulation suggests that the smallest NTRUSign parameter set, which was claimed to provide at least 93-bit security against key-recovery lattice attacks, actually offers at most 65-bit security.

Security Level

- $\delta = \left(\frac{\lambda_1}{\det(\Lambda)^{1/n}}\right)^{1/n}$ [CN11]
- "Exact" bitlevel correpsondance [LP11]

k	δ
80	1.00783
100	1.00696
128	1.00602

$$\log_2(\delta) := \frac{1.8}{\log_2(\frac{T_{BKZ}(\delta)}{2^{30}}) + 110} = \frac{1.8}{k - 30 + 110} = \frac{1.8}{k + 80}$$



Better Key Sizes (and Attacks) for
LWE-Based Encryption

Richard Lindner[*] Chris Peikert[†]

November 30, 2010

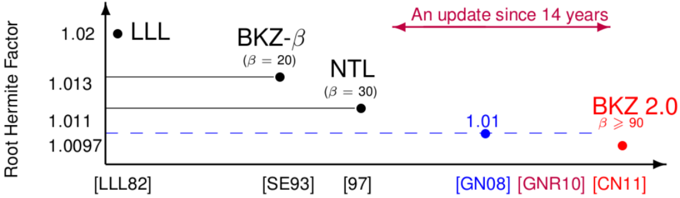
Abstract

We analyze the concrete security and key sizes of theoretically sound lattice-based encryption schemes based on the "learning with errors" (LWE) problem. Our main contributions are: (1) a new lattice attack on LWE that combines basis reduction with an enumeration algorithm admitting a time/success tradeoff, which performs better than the simple distinguishing attack considered in prior analyses; (2) concrete parameters and security estimates for an LWE-based cryptosystem that is more compact and efficient than the well-known schemes from the literature. Our new key sizes are up to 10 times smaller than prior examples, while providing even stronger concrete security levels.

Security Level

- $\delta = \left(\frac{\lambda_1}{\det(\Lambda)^{1/n}}\right)^{1/n}$ [CN11]
- "Exact" bitlevel correpsondance [LP11]
- Depends on the algorithm

k	δ
80	1.00783
100	1.00696
128	1.00602



LBC: what about encryption

In 2005, Regev proposed a lattice-based encryption scheme.

KeyGen

Given n, m, q, α, generate $\mathbf{e} \leftarrow D_\alpha$ output $sk = \mathbf{s} \in \{-1, 0, 1\}^n$ and $pk = (\mathbf{A}, \mathbf{b})$ where $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$

Decrypt

Compute $\ell = v - \mathbf{u}^\top \mathbf{s}$. If ℓ is close to 0 output 0, otherwise, output 1.

Encrypt $m \in \{0, 1\}$

$\mathbf{r} \leftarrow \{0, 1\}$ and output $\mathbf{u} = \mathbf{r}^\top \mathbf{A}$ and $v = \mathbf{r}^\top \mathbf{b} + \lfloor q/2 \rfloor \times m$

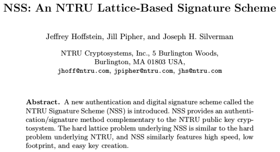
Regev's cryptosystem relies on a lattice-related problem called LWE.
Notice that there exist other cryptosystems that improve upon this one.

NTRUSign: lattice-based signature

NTRUSign: lattice-based signature

History

- Originally NSS [HPS01]



NSS: An NTRU Lattice-Based Signature Scheme

Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman

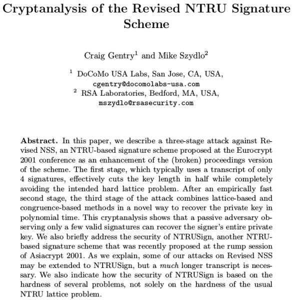
NTRU Cryptosystems, Inc., 5 Burlington Woods,
Burlington, MA 01803 USA,
`jhoff@ntru.com`, `jpipher@ntru.com`, `jhs@ntru.com`

Abstract. A new authentication and digital signature scheme called the NTRU Signature Scheme (NSS) is introduced. NSS provides an authentication/signature method complementary to the NTRU public key cryptosystem. The hard lattice problem underlying NSS is similar to the hard problem underlying NTRU, and NSS similarly features high speed, low footprint, and easy key creation.

NTRUSign: lattice-based signature

History

- Originally NSS [HPS01]
  Quickly broken [GS02]



Cryptanalysis of the Revised NTRU Signature Scheme

Craig Gentry[1] and Mike Szydlo[2]

[1] DoCoMo USA Labs, San Jose, CA, USA,
cgentry@docomolabs-usa.com
[2] RSA Laboratories, Bedford, MA, USA,
mszydlo@rsasecurity.com

Abstract. In this paper, we describe a three-stage attack against Revised NSS, an NTRU-based signature scheme proposed at the Eurocrypt 2001 conference as an enhancement of the (broken) proceedings version of the scheme. The first stage, which typically uses a transcript of only 4 signatures, effectively cuts the key length in half while completely avoiding the intended hard lattice problem. After an empirically fast second stage, the third stage of the attack combines lattice-based and congruence-based methods in a novel way to recover the private key in polynomial time. This cryptanalysis shows that a passive adversary observing only a few valid signatures can recover the signer's entire private key. We also briefly address the security of NTRUSign, another NTRU-based signature scheme that was recently proposed at the rump session of Asiacrypt 2001. As we explain, some of our attacks on Revised NSS may be extended to NTRUSign, but a *much* longer transcript is necessary. We also indicate how the security of NTRUSign is based on the hardness of several problems, not solely on the hardness of the usual NTRU lattice problem.

NTRUSign: lattice-based signature

History

- Originally NSS [HPS01]
  Quickly broken [GS02]
- NTRUSign [HPSW02]

$$\mathbf{f}, \mathbf{g} = \begin{cases} d \text{ coefficients } & +1 \\ N - d \text{ coefficients } & 0 \end{cases}$$

$$\mathbf{F}, \mathbf{G} \text{ st. } \mathbf{f} * \mathbf{G} - \mathbf{F} * \mathbf{g} = q$$

$$\mathbf{h} = \mathbf{g} * \mathbf{f}^{-1} \overset{\$}{\leftsquigarrow} \mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^N + 1\rangle$$

NTRUSign: lattice-based signature

History

- Originally NSS [HPS01]
  Quickly broken [GS02]
- NTRUSign [HPSW02]

$$\mathbf{f},\ \mathbf{g} = \begin{cases} d \text{ coefficients } & +1 \\ N-d \text{ coefficients } & 0 \end{cases}$$

$$\mathbf{F},\ \mathbf{G} \text{ st. } \mathbf{f} * \mathbf{G} - \mathbf{F} * \mathbf{g} = q$$

$$\mathbf{h} = \mathbf{g} * \mathbf{f}^{-1} \overset{\$}{\leftsquigarrow} \mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^N + 1 \rangle$$

$$\mathbf{P} = \left(\begin{array}{c|c} \mathbf{1} \circlearrowleft & \mathbf{h} \circlearrowleft \\ \hline \mathbf{0} \circlearrowleft & \mathbf{q} \circlearrowleft \end{array} \right) \qquad \mathbf{S} = \left(\begin{array}{c|c} \mathbf{f} \circlearrowleft & \mathbf{g} \circlearrowleft \\ \hline \mathbf{F} \circlearrowleft & \mathbf{G} \circlearrowleft \end{array} \right)$$

NTRUSign: lattice-based signature

History

- Originally NSS [HPS01]
  Quickly broken [GS02]
- NTRUSign [HPSW02]

$$\mathbf{f},\ \mathbf{g} = \begin{cases} d \text{ coefficients } & +1 \\ N-d \text{ coefficients } & 0 \end{cases}$$

$$\mathbf{F},\ \mathbf{G} \text{ st. } \mathbf{f} * \mathbf{G} - \mathbf{F} * \mathbf{g} = q$$

$$\mathbf{h} = \mathbf{g} * \mathbf{f}^{-1} \overset{\$}{\leftsquigarrow} \mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^N + 1\rangle$$

$$\mathbf{P} = \left(\begin{array}{c|c} \mathbf{1} \circlearrowleft & \mathbf{h} \circlearrowleft \\ \hline \mathbf{0} \circlearrowleft & \mathbf{q} \circlearrowleft \end{array}\right) \qquad \mathbf{S} = \left(\begin{array}{c|c} \mathbf{f} \circlearrowleft & \mathbf{g} \circlearrowleft \\ \hline \mathbf{F} \circlearrowleft & \mathbf{G} \circlearrowleft \end{array}\right)$$

NTRU lattice: $\Lambda_{\mathbf{h},q} = \{(\mathbf{u}, \mathbf{u} * \mathbf{h} \mod q), \mathbf{u} \in \mathcal{R}_q\}$

NTRUSign

Sign

Given $\mu \in \{0,1\}^*$ to sign:

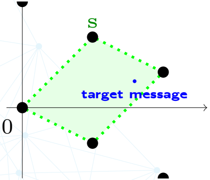
- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

NTRUSign

Sign

Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

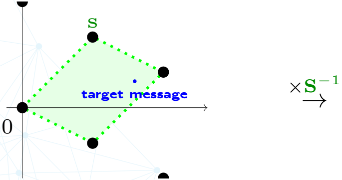


NTRUSign

Sign

Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

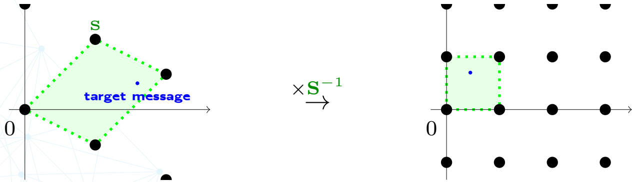


NTRUSign

Sign

Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

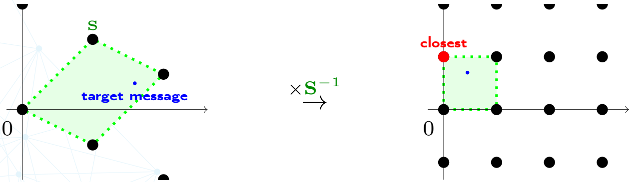


NTRUSign

Sign

Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

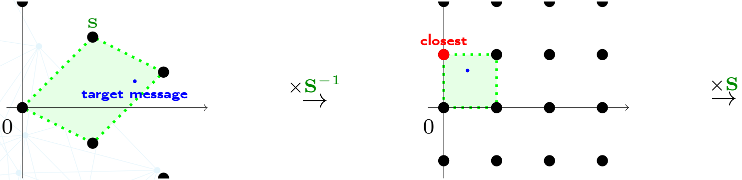


NTRUSign

Sign

Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

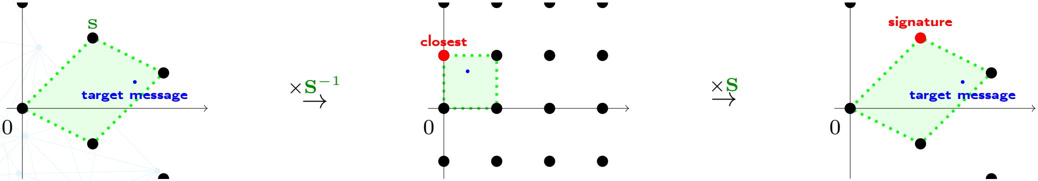


NTRUSign

Sign

Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$



NTRUSign

Sign

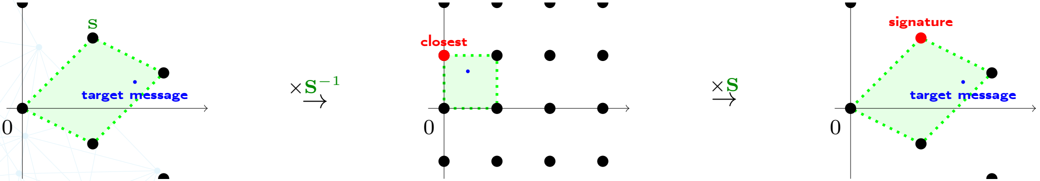
Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

Verify

Given the signature s, check:

- It's a lattice point (using bad basis $\mathbf{P}$)
- Not far from $(\mathbf{0}, \mathbf{m})$



NTRUSign

Sign

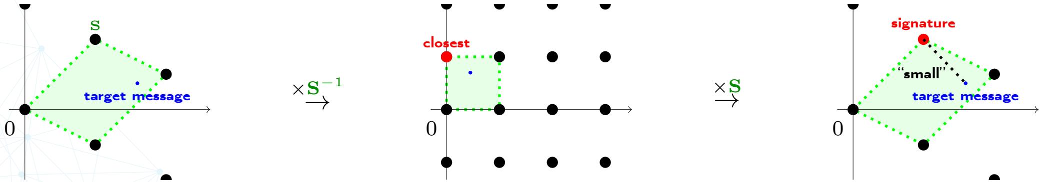
Given $\mu \in \{0,1\}^*$ to sign:

- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

Verify

Given the signature s, check:

- It's a lattice point (using bad basis $\mathbf{P}$)
- Not far from $(\mathbf{0}, \mathbf{m})$



NTRUSign

Signature Size (in bits)

security	80	112	128	160
NTRUSign	1256	1576	1784	2367
ECDSA_{sign}	320	448	512	640
RSA	1024	2048	3072	4096

NTRUSign

Signature Size (in bits)

security	80	112	128	160
NTRUSign	1256	1576	1784	2367
ECDSA$_{sign}$	320	448	512	640
RSA	1024	2048	3072	4096

NTRUSign runs faster !

NTRUSign

Signature Size (in bits)

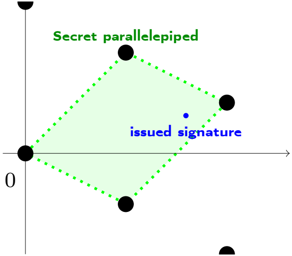
security	80	112	128	160
NTRUSign	1256	1576	1784	2367
ECDSA$_{sign}$	320	448	512	640
RSA	1024	2048	3072	4096

NTRUSign runs faster !

But...

Problem : Not Zero-Knowledge

Key-recovery attacks

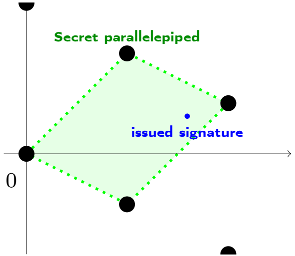


Number of signature issued : 1

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]

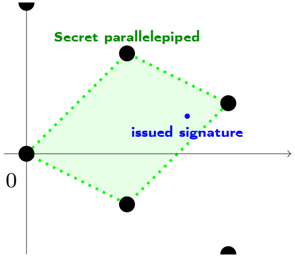


Number of signature issued : 1

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

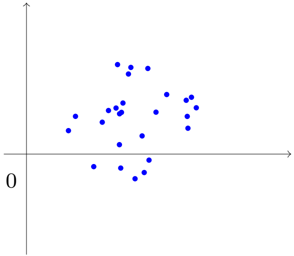


Number of signature issued : 1

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

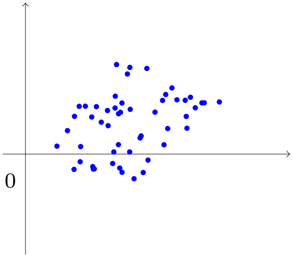


Number of signatures issued : 25

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

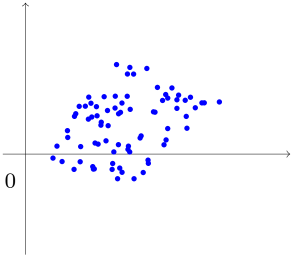


Number of signatures issued : 50

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

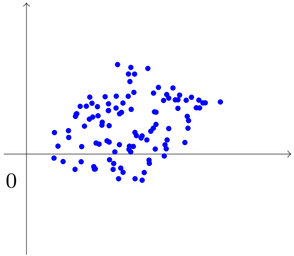


Number of signatures issued : 75

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

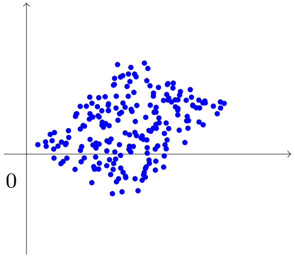


Number of signatures issued : 100

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

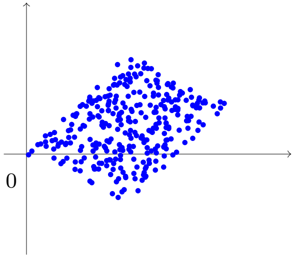


Number of signatures issued : 200

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

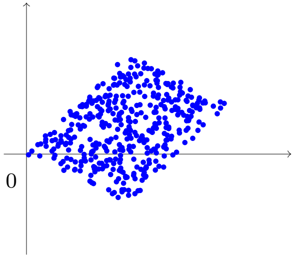


Number of signatures issued : 300

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

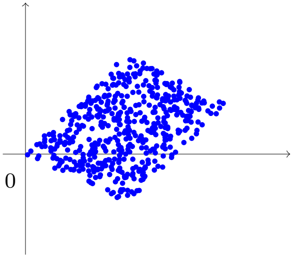


Number of signatures issued : 400

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

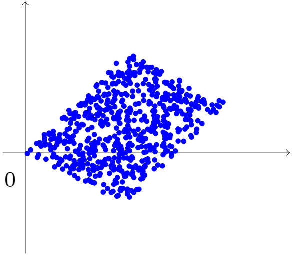


Number of signatures issued : 500

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

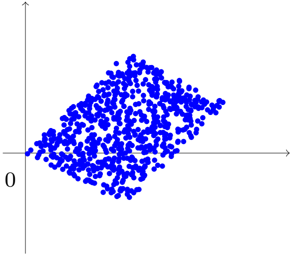


Number of signatures issued : 600

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

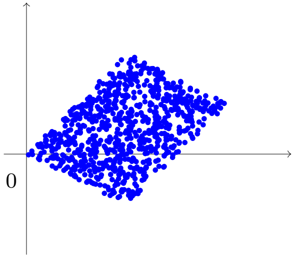


Number of signatures issued : 700

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

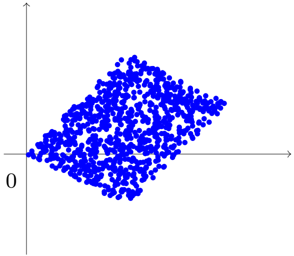


Number of signatures issued : 800

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]

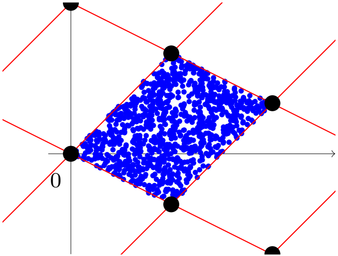


Number of signatures issued : 900

Problem : Not Zero-Knowledge

Key-recovery attacks

- Only a few signatures for original scheme [NR06]
- And a little more to break coutermeasures [DN12]



Number of signatures issued : 1000

Secure lattice based signatures [Lyu12]

KeyGen

- Secret key : $\mathbf{S} \overset{\$}{\leftarrow} \{-d, \ldots, 0, \ldots, d\}^{m \times k}$
- Public key : $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times m}$ and $\mathbf{T} = \mathbf{A} \cdot \mathbf{S} \in \mathbb{Z}_q^{n \times k}$

Sign

Secure lattice based signatures [Lyu12]

KeyGen

- Secret key : $\mathbf{S} \overset{\$}{\leftarrow} \{-d, \ldots, 0, \ldots, d\}^{m \times k}$
- Public key : $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times m}$ and $\mathbf{T} = \mathbf{A} \cdot \mathbf{S} \in \mathbb{Z}_q^{n \times k}$

Sign

First stage [Finding pre-image]

- map μ to a space element $\mathbf{c}$
- $\mathbf{Sc}$ is a short pre-image of $\mathbf{Tc}$

Secure lattice based signatures [Lyu12]

KeyGen

- Secret key : $\mathbf{S} \overset{\$}{\leftarrow} \{-d, \ldots, 0, \ldots, d\}^{m \times k}$
- Public key : $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times m}$ and $\mathbf{T} = \mathbf{A} \cdot \mathbf{S} \in \mathbb{Z}_q^{n \times k}$

Sign

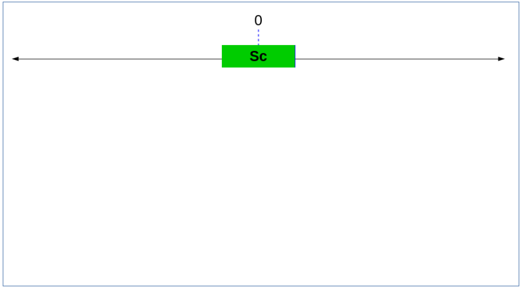
First stage [Finding pre-image]

- map μ to a space element $\mathbf{c}$
- $\mathbf{Sc}$ is a short pre-image of $\mathbf{Tc}$

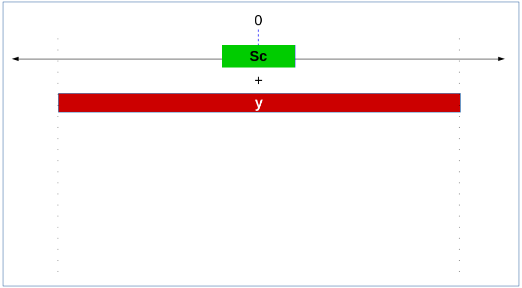
Second stage [Hiding pre-image]

- Add gaussian noise $\mathbf{y}$ to $\mathbf{Sc}$
- Apply rejection sampling to avoid leakage

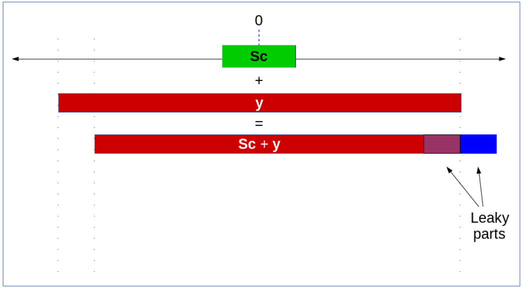
Secure lattice based signatures [Lyu12]



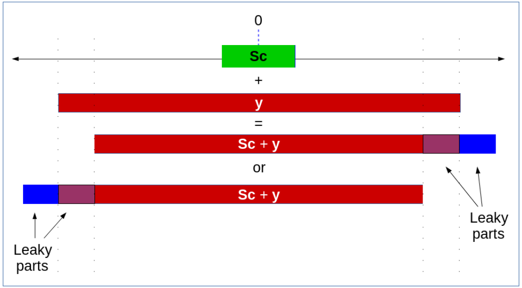
Secure lattice based signatures [Lyu12]



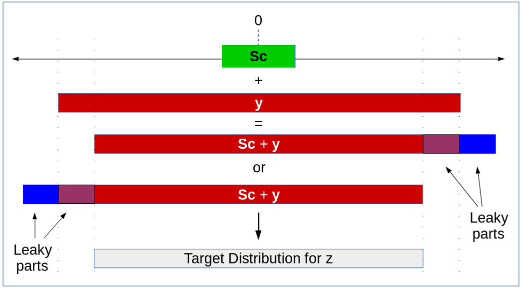
Secure lattice based signatures [Lyu12]



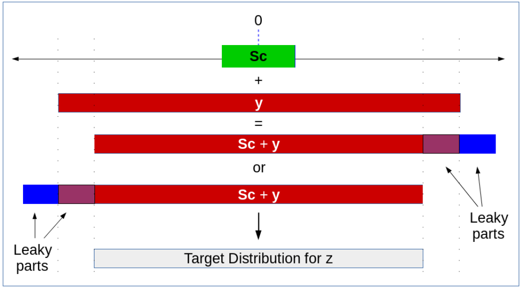
Secure lattice based signatures [Lyu12]



Secure lattice based signatures [Lyu12]



Secure lattice based signatures [Lyu12]



Verify

Given $(\mathbf{z}, \mathbf{c})$, check that :

- $H(\underbrace{\mathbf{Az} - \mathbf{Tc}}_{\mathbf{A}(\mathbf{Sc}+\mathbf{y})-\mathbf{ASc}}, \mu) = \mathbf{c}$ $\rightarrow$ it is a lattice vector
- $\|\mathbf{z}\| \leq \eta\sigma\sqrt{m}$ $\rightarrow$ it has reasonnable norm

Sets of parameters

100 bits of security

n	512	512	512	512	512
m	8,786	8,139	3,253	1,024	1,024
k	80	512	512	512	512
$\log_2(q)$	27	25	33	18	26
d	1	1	31	1	31
M (retries)	2.72	2.72	2.72	7.4	7.4
$\approx$ sign size	163,000	142,300	73,000	14,500	19,500
$\approx$ pk size	2^{20}	$2^{22.5}$	2^{23}	$2^{19.5}$	$2^{21.5}$
$\approx$ sk size	2^{20}	$2^{22.5}$	2^{23}	$2^{22.1}$	$2^{22.7}$

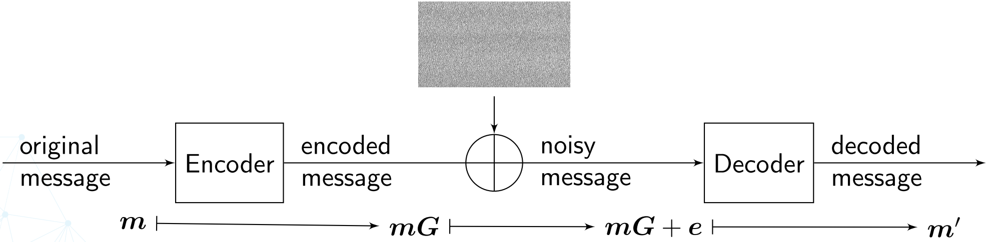
More recent proposals achieve better security, parameters and performances (along with other nice features).

Outline

6 Post-quantum cryptography
- Lattice-based cryptography
- Code-based cryptography
- Hash-based cryptography

Coding theory

Coding theory is the science of (efficiently) adding redundancy to information in order to detect/correct errors that could occur during transmission.



original message

Encoder

encoded message

noisy message

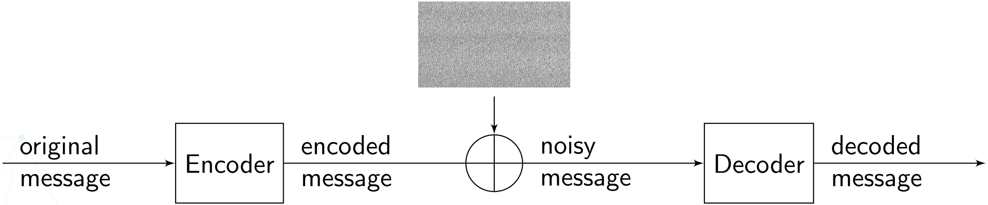
Decoder

decoded message

$$m \longmapsto mG \longmapsto mG + e \longmapsto m'$$

Coding theory

Coding theory is the science of (efficiently) adding redundancy to information in order to detect/correct errors that could occur during transmission.



original message

Encoder

encoded message

noisy message

Decoder

decoded message

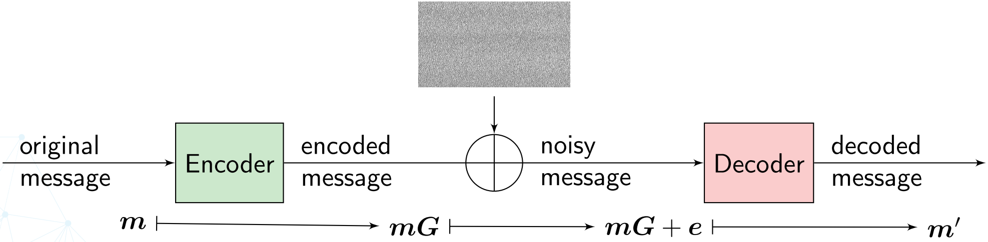
$$m \longmapsto mG \longmapsto mG + e \longmapsto m'$$

Preliminary remarks:

- Hopefully, we have $m' = m$

Coding theory

Coding theory is the science of (efficiently) adding redundancy to information in order to detect/correct errors that could occur during transmission.



$$m \longmapsto mG \longmapsto mG + e \longmapsto m'$$

Preliminary remarks:

- Hopefully, we have $m' = m$
- For code-based PKC, most of the time, public encoder / private decoder.

Definitions

Linear code

A *linear code* of dimension k and length n over $\mathbb{F}_q$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

A linear code $\mathcal{C}[n, k]$ is fully determined by one of the following matrices:

Definitions

Linear code

A *linear code* of dimension k and length n over $\mathbb{F}_q$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

A linear code $\mathcal{C}[n, k]$ is fully determined by one of the following matrices:

Generator matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$

$\mathcal{C} = \left\{ \mathbf{xG}, \text{ for } \mathbf{x} \in \mathbb{F}_q^k \right\}$

Definitions

Linear code

A *linear code* of dimension k and length n over $\mathbb{F}_q$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

A linear code $\mathcal{C}[n,k]$ is fully determined by one of the following matrices:

Generator matrix $\boldsymbol{G} \in \mathbb{F}_q^{k\times n}$

$\mathcal{C} = \left\{\mathbf{xG}, \text{ for } \mathbf{x} \in \mathbb{F}_q^k\right\}$

Parity-check matrix $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times n}$

$\mathcal{C} = \left\{\boldsymbol{s} \in \mathbb{F}_q^n \text{ such that } \boldsymbol{H}\boldsymbol{s}^\top = \mathbf{0}\right\}$

Definitions

Linear code

A *linear code* of dimension k and length n over $\mathbb{F}_q$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

A linear code $\mathcal{C}[n,k]$ is fully determined by one of the following matrices:

Generator matrix $\boldsymbol{G} \in \mathbb{F}_q^{k \times n}$

$\mathcal{C} = \left\{ \mathbf{xG}, \text{ for } \mathbf{x} \in \mathbb{F}_q^k \right\}$

Parity-check matrix $\boldsymbol{H} \in \mathbb{F}_q^{(n-k) \times n}$

$\mathcal{C} = \left\{ \boldsymbol{s} \in \mathbb{F}_q^n \text{ such that } \boldsymbol{H}\boldsymbol{s}^\top = \mathbf{0} \right\}$

The Hamming weight of a word $\boldsymbol{u}$ is the number of its non-zero coordinates:

$$wt\left(\boldsymbol{u}\right) = \#\left\{i \in \{0, \ldots, n-1\} \text{ such that } \boldsymbol{u}_i \neq 0\right\}$$

$$\text{example : } wt\left((0,1,0,0,1,0,1,0)\right) = 3$$

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
    - De détecter l'erreur $\Rightarrow$ Renvoi
    - De corriger l'erreur

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
  - De détecter l'erreur ⇒ Renvoi
  - De corriger l'erreur

Exemple basique : code à 3-répétition

- Alice souhaite envoyer $1 \cdot 0 \cdot 1$
- Elle envoie $111 \cdot 000 \cdot 111$ à Bob
- Bob reçoit $101 \cdot 001 \cdot 111$
- Il interprète correctement en $1 \cdot 0 \cdot 1$

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
    - De détecter l'erreur ⇒ Renvoi
    - De corriger l'erreur

Métrique de Hamming

$\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$, disons $\mathbb{F}_5^7$

$$\mathbf{u} = \begin{array}{|c|c|c|c|c|c|c|}\hline 3 & 3 & 2 & 4 & 4 & 5 & 2 \\ \hline\end{array}$$

$$\mathbf{v} = \begin{array}{|c|c|c|c|c|c|c|}\hline 5 & 3 & 1 & 2 & 4 & 5 & 5 \\ \hline\end{array}$$

Exemple basique : code à 3-répétition

- Alice souhaite envoyer $1 \cdot 0 \cdot 1$
- Elle envoie $111 \cdot 000 \cdot 111$ à Bob
- Bob reçoit $101 \cdot 001 \cdot 111$
- Il interprète correctement en $1 \cdot 0 \cdot 1$

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
    - De détecter l'erreur ⇒ Renvoi
    - De corriger l'erreur

Métrique de Hamming

$\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$, disons $\mathbb{F}_5^7$

$\mathbf{u} =$	3	3	2	4	4	5	2
$\mathbf{v} =$	5	3	1	2	4	5	5

Exemple basique : code à 3-répétition

- Alice souhaite envoyer $1 \cdot 0 \cdot 1$
- Elle envoie $111 \cdot 000 \cdot 111$ à Bob
- Bob reçoit $101 \cdot 001 \cdot 111$
- Il interprète correctement en $1 \cdot 0 \cdot 1$

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
    - De détecter l'erreur ⇒ Renvoi
    - De corriger l'erreur

Métrique de Hamming

$\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$, disons $\mathbb{F}_5^7 \rightarrow d_H(\mathbf{u}, \mathbf{v}) = 4$

$\mathbf{u} =$	3	3	2	4	4	5	2
$\mathbf{v} =$	5	3	1	2	4	5	5

Exemple basique : code à 3-répétition

- Alice souhaite envoyer $1 \cdot 0 \cdot 1$
- Elle envoie $111 \cdot 000 \cdot 111$ à Bob
- Bob reçoit $101 \cdot 001 \cdot 111$
- Il interprète correctement en $1 \cdot 0 \cdot 1$

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
  - De détecter l'erreur ⇒ Renvoi
  - De corriger l'erreur

Métrique de Hamming

$\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$, disons $\mathbb{F}_5^7 \to d_H(\mathbf{u}, \mathbf{v}) = 4$

$\mathbf{u} =$	3	3	2	4	4	5	2
$\mathbf{v} =$	5	3	1	2	4	5	5

Exemple basique : code à 3-répétition

- Alice souhaite envoyer $1 \cdot 0 \cdot 1$
- Elle envoie $111 \cdot 000 \cdot 111$ à Bob
- Bob reçoit $101 \cdot 001 \cdot 111$
- Il interprète correctement en $1 \cdot 0 \cdot 1$

- Métrique bien étudiée

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
    - De détecter l'erreur ⇒ Renvoi
    - De corriger l'erreur

Métrique de Hamming

$\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$, disons $\mathbb{F}_5^7 \rightarrow d_H(\mathbf{u}, \mathbf{v}) = 4$

$\mathbf{u} =$	3	3	2	4	4	5	2
$\mathbf{v} =$	5	3	1	2	4	5	5

Exemple basique : code à 3-répétition

- Alice souhaite envoyer $1 \cdot 0 \cdot 1$
- Elle envoie $111 \cdot 000 \cdot 111$ à Bob
- Bob reçoit $101 \cdot 001 \cdot 111$
- Il interprète correctement en $1 \cdot 0 \cdot 1$

- Métrique bien étudiée
- Nombreuses familles avec différentes propriétés

Codes Correcteurs

Théorie des Codes

- Ajout de redondance à l'information
- En cas d'erreur(s), permet soit :
  - De détecter l'erreur ⇒ Renvoi
  - De corriger l'erreur

Métrique de Hamming

$\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$, disons $\mathbb{F}_5^7 \to d_H(\mathbf{u}, \mathbf{v}) = 4$

$\mathbf{u} =$	3	3	2	4	4	5	2
$\mathbf{v} =$	5	3	1	2	4	5	5

Exemple basique : code à 3-répétition

- Alice souhaite envoyer $1 \cdot 0 \cdot 1$
- Elle envoie $111 \cdot 000 \cdot 111$ à Bob
- Bob reçoit $101 \cdot 001 \cdot 111$
- Il interprète correctement en $1 \cdot 0 \cdot 1$

- Métrique bien étudiée
- Nombreuses familles avec différentes propriétés
- Attaques plus directes qu'en métrique rang

Code-based cryptography (CBC)

Que sont les codes correcteurs ?

Code-based cryptography (CBC)

Que sont les codes correcteurs ?

Des façons de rajouter de la redondance à l'information utile, afin d'être capable de détecter — voire corriger — d'éventuelles erreurs lors de la transmission.

Code-based cryptography (CBC)

Que sont les codes correcteurs ?

Des façons de rajouter de la redondance à l'information utile, afin d'être capable de détecter — voire corriger — d'éventuelles erreurs lors de la transmission.

Exemple : le code à répétition

Message à envoyer		1			0			1	
Encodage	1	1	1	0	0	0	1	1	1
Message reçu	0	1	1	0	1	0	1	1	0
Message décodé		1			0			1	

Code-based cryptography (CBC)

Que sont les codes correcteurs ?

Des façons de rajouter de la redondance à l'information utile, afin d'être capable de détecter — voire corriger — d'éventuelles erreurs lors de la transmission.

Exemple : le code à répétition

Message à envoyer		1			0			1	
Encodage	1	1	1	0	0	0	1	1	1
Message reçu	0	1	1	0	1	0	1	1	0
Message décodé		1			0			1	

Ce code est particulièrement mauvais (bien qu'utile pédagogiquement parlant) :

- dimension : $k = 1$
- longueur : $n = 3$
- distance minimale : $d = 3$
- capacité de détection : $d - 1 = 2$ erreurs
- capacité de correction : $\left\lfloor \frac{d-1}{2} \right\rfloor = 1$ erreur
- rendement $\frac{k}{n} = \frac{1}{3}$.

Code-based cryptography (CBC)

Un code $\mathcal{C}$ est entièrement définit par sa matrice génératrice $\mathbf{G}$:

$$\mathcal{C} = \left\{ \mathbf{x}\mathbf{G}, \text{ pour } \mathbf{x} \in \mathbb{F}_2^k \right\}$$

Code-based cryptography (CBC)

Un code $\mathcal{C}$ est entièrement définit par sa matrice génératrice $\mathbf{G}$:

$$\mathcal{C} = \left\{\mathbf{x}\mathbf{G}, \text{ pour } \mathbf{x} \in \mathbb{F}_2^k\right\}$$

Ou de manière équivalente, par une matrice de parité $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$:

$$\mathcal{C} = \left\{\mathbf{s} \in \mathbb{F}_2^n \text{ tels que } \mathbf{H}\mathbf{s}^\top = \mathbf{0}\right\}$$

Code-based cryptography (CBC)

Un code $\mathcal{C}$ est entièrement définit par sa matrice génératrice $\mathbf{G}$:

$$\mathcal{C} = \left\{\mathbf{xG}, \text{ pour } \mathbf{x} \in \mathbb{F}_2^k\right\}$$

Ou de manière équivalente, par une matrice de parité $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$:

$$\mathcal{C} = \left\{\mathbf{s} \in \mathbb{F}_2^n \text{ tels que } \mathbf{Hs}^\top = \mathbf{0}\right\}$$

Le poids de Hamming d'un mot (un vecteur) est défini comme l'ensemble de ses coordonnées non-nulles :

$$wt\left(\mathbf{x}\right) = \#\left\{i \in \{0, \ldots, n-1\} \text{ tels que } \mathbf{x}_i \neq 0\right\}$$

$$\text{exemple : } wt\left((0, 1, 0, 0, 1, 0, 1, 0)\right) = ?$$

Code-based cryptography (CBC)

Un code $\mathcal{C}$ est entièrement définit par sa matrice génératrice $\mathbf{G}$:

$$\mathcal{C} = \left\{\mathbf{xG}, \text{ pour } \mathbf{x} \in \mathbb{F}_2^k\right\}$$

Ou de manière équivalente, par une matrice de parité $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$:

$$\mathcal{C} = \left\{\mathbf{s} \in \mathbb{F}_2^n \text{ tels que } \mathbf{H}\mathbf{s}^\top = \mathbf{0}\right\}$$

Le poids de Hamming d'un mot (un vecteur) est défini comme l'ensemble de ses coordonnées non-nulles :

$$wt\left(\mathbf{x}\right) = \#\left\{i \in \{0, \ldots, n-1\} \text{ tels que } \mathbf{x}_i \neq 0\right\}$$

$$\text{exemple : } wt\left((0,1,0,0,1,0,1,0)\right) = 3$$

Code-based cryptography (CBC)

Problème du décodage de syndrome.

Code-based cryptography (CBC)

Problème du décodage de syndrome.

Problème

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^{n}$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$.

Code-based cryptography (CBC)

Problème du décodage de syndrome.

Problème

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^n$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$.

Ce problème est-il difficile ?

Code-based cryptography (CBC)

Problème du décodage de syndrome.

Problème

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^n$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$.

Ce problème est-il difficile ? **non !**

Code-based cryptography (CBC)

Problème du décodage de syndrome.

Problème

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^n$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$.

Ce problème est-il difficile ? non !
Il suffit de réaliser un pivot de Gauss sur la matrice $\mathbf{H}$. C'est purement un problème d'algèbre linéaire...

Code-based cryptography (CBC)

Problème du décodage de syndrome.

Problème

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^n$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$.

Ce problème est-il difficile ? non !
Il suffit de réaliser un pivot de Gauss sur la matrice $\mathbf{H}$. C'est purement un problème d'algèbre linéaire...

Problème modifié

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^n$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$**, et $\mathbf{x}$ de poids relativement faible**.

Code-based cryptography (CBC)

Problème du décodage de syndrome.

Problème

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^n$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$.

Ce problème est-il difficile ? non !
Il suffit de réaliser un pivot de Gauss sur la matrice $\mathbf{H}$. C'est purement un problème d'algèbre linéaire...

Problème modifié

Soit $\mathbf{s} \in \mathbb{F}_2^{n-k}$ et $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$. Trouver $\mathbf{x} \in \mathbb{F}_2^n$ tel que $\mathbf{H}\mathbf{x}^\top = \mathbf{s}$**, et $\mathbf{x}$ de poids relativement faible**.

Le problème devient *NP*-difficile [?].

(Traduction: il devient cryptographiquement intéressant)

Code-based cryptography (CBC)

Cryptosystème de McEliece [**?**]

Code-based cryptography (CBC)

Cryptosystème de McEliece [**?**]

Soit $\mathbf{G} \in \mathbb{F}_2^{k \times n}$ la matrice génératrice d'un code (de Goppa binaire) $\mathcal{C}$ pouvant corriger jusqu'à t erreurs à l'aide de l'algorithme de décodage $\mathcal{D}_{\mathbf{G}}$.

Code-based cryptography (CBC)

Cryptosystème de McEliece [?]
Soit $\mathbf{G} \in \mathbb{F}_2^{k\times n}$ la matrice génératrice d'un code (de Goppa binaire) $\mathcal{C}$ pouvant corriger jusqu'à t erreurs à l'aide de l'algorithme de décodage $\mathcal{D}_{\mathbf{G}}$.

matrice inversible $\mathbf{S} \in \mathbb{F}_2^{k\times k}$

matrice permutation $\mathbf{P} \in \mathbb{F}_2^{n\times n}$

message $\mathbf{m} \in \mathbb{F}_2^{k}$

$$\xrightarrow{\tilde{\mathbf{G}}=\mathbf{SGP},n,k,t}$$

$\mathbf{e} \in \mathbb{F}_2^{n}$ tel que $wt\left(\mathbf{e}\right) \leq t$

$$\mathbf{c} = \mathbf{m}\tilde{\mathbf{G}} + \mathbf{e}$$

$$\xleftarrow{\mathbf{c}}$$

$$\tilde{\mathbf{c}} = \mathcal{D}_{\mathbf{G}}\left(\mathbf{cP}^{-1}\right) = \mathcal{D}_{\mathbf{G}}\left(\mathbf{mSG} + \mathbf{eP}^{-1}\right)$$
$$\mathbf{m} = \tilde{\mathbf{c}}\mathbf{S}^{-1}$$

CBC : un exemple

Soit $\mathcal{C}$ le code (de Hamming) admettant pour matrice de parité $\mathbf{H}$:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Soit $\mathbf{s} = (1, 0, 0, 0, 1, 1, 1)$ le mot reçu. Quel était le message envoyé ?

CBC : un exemple

Soit $\mathcal{C}$ le code (de Hamming) admettant pour matrice de parité $\mathbf{H}$:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Soit $\mathbf{s} = (1, 0, 0, 0, 1, 1, 1)$ le mot reçu. Quel était le message envoyé ? Décodons

CBC : un exemple

Soit $\mathcal{C}$ le code (de Hamming) admettant pour matrice de parité $\mathbf{H}$:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Soit $\mathbf{s} = (1, 0, 0, 0, 1, 1, 1)$ le mot reçu. Quel était le message envoyé ? Décodons

$$\mathbf{v} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$
$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

CBC : un exemple

Soit $\mathcal{C}$ le code (de Hamming) admettant pour matrice de parité $\mathbf{H}$:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Soit $\mathbf{s} = (1,0,0,0,1,1,1)$ le mot reçu. Quel était le message envoyé ? Décodons

$$\mathbf{v} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \mathbf{s}$$

CBC : un exemple

Soit $\mathcal{C}$ le code (de Hamming) admettant pour matrice de parité $\mathbf{H}$:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Soit $\mathbf{s} = (1,0,0,0,1,1,1)$ le mot reçu. Quel était le message envoyé ? Décodons

$$\mathbf{v} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \mathbf{s}$$

CBC : un exemple

Soit $\mathcal{C}$ le code (de Hamming) admettant pour matrice de parité $\mathbf{H}$:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Soit $\mathbf{s} = (1, 0, 0, 0, 1, 1, 1)$ le mot reçu. Quel était le message envoyé ? Décodons

$$\mathbf{v} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \quad \mathbf{e} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \mathbf{s}$$

CBC : un exemple

Soit $\mathcal{C}$ le code (de Hamming) admettant pour matrice de parité $\mathbf{H}$:

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Soit $\mathbf{s} = (1, 0, 0, 0, 1, 1, 1)$ le mot reçu. Quel était le message envoyé ? Décodons

$$\mathbf{v} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \quad \mathbf{e} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \mathbf{s}$$

$$\mathbf{m} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Quasi-Cyclic Moderate Density Parity-Check Codes

KeyGen

Sample $\mathbf{h}_0, \mathbf{h}_1 \leftarrow \mathbb{F}_2^r$ of small weight w, $\mathbf{h}_0$ inversible. Compute $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

$$\mathbf{H}_{\text{secret}} = \left(\begin{array}{c|c} \underset{\circlearrowright}{\mathbf{h}_0} & \underset{\circlearrowright}{\mathbf{h}_1} \end{array} \right)$$

$$\mathbf{H}_{\text{pub}} = \left(\begin{array}{c|c} \underset{\circlearrowright}{(1, 0, \ldots, 0)} & \underset{\circlearrowright}{\mathbf{h}} \end{array} \right)$$

Quasi-Cyclic Moderate Density Parity-Check Codes

KeyGen

Sample $\mathbf{h}_0, \mathbf{h}_1 \leftarrow \mathbb{F}_2^r$ of small weight w, $\mathbf{h}_0$ inversible. Compute $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

$$\mathbf{H}_{\text{secret}} = \left(\begin{array}{c|c} \mathbf{h}_0 & \mathbf{h}_1 \\ \circlearrowright & \circlearrowright \end{array} \right)$$

$$\mathbf{H}_{\text{pub}} = \left(\begin{array}{c|c} (1, 0, \ldots, 0) & \mathbf{h} \\ \circlearrowright & \circlearrowright \end{array} \right)$$

Encryption

As for McEliece, $\mathbf{e}$ of weight t,

$$\mathbf{c} = \mathbf{mG} + \mathbf{e}.$$

Quasi-Cyclic Moderate Density Parity-Check Codes

KeyGen

Sample $\mathbf{h}_0, \mathbf{h}_1 \leftarrow \mathbb{F}_2^r$ of small weight w, $\mathbf{h}_0$ inversible. Compute $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

$$\mathbf{H}_{\text{secret}} = \left(\begin{array}{c|c} \mathbf{h}_0 & \mathbf{h}_1 \\ \circlearrowright & \circlearrowright \end{array}\right)$$

$$\mathbf{H}_{\text{pub}} = \left(\begin{array}{c|c} (1, 0, \ldots, 0) & \mathbf{h} \\ \circlearrowright & \circlearrowright \end{array}\right)$$

Encryption

As for McEliece, $\mathbf{e}$ of weight t,

$$\mathbf{c} = \mathbf{m}\mathbf{G} + \mathbf{e}.$$

Decryption

Use an iterative decoder (e.g. the BitFlipping algorithm) to recover message $\mathbf{m}$.

Quasi-Cyclic Moderate Density Parity-Check Codes

KeyGen

Sample $\mathbf{h}_0, \mathbf{h}_1 \leftarrow \mathbb{F}_2^r$ of small weight w, $\mathbf{h}_0$ inversible. Compute $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

$$\mathbf{H}_{\text{secret}} = \left(\begin{array}{c|c} \underset{\circlearrowright}{\mathbf{h}_0} & \underset{\circlearrowright}{\mathbf{h}_1} \end{array} \right)$$

$$\mathbf{H}_{\text{pub}} = \left(\begin{array}{c|c} \underset{\circlearrowright}{(1, 0, \ldots, 0)} & \underset{\circlearrowright}{\mathbf{h}} \end{array} \right)$$

Encryption

As for McEliece, $\mathbf{e}$ of weight t,

$$\mathbf{c} = \mathbf{m}\mathbf{G} + \mathbf{e}.$$

Decryption

Use an iterative decoder (e.g. the BitFlipping algorithm) to recover message $\mathbf{m}$.

Suggested parameters: $r = 9857, n = 2r, w = 142, t = 134$ for 128 bits.

Quasi-Cyclic Moderate Density Parity-Check Codes

KeyGen

Sample $\mathbf{h}_0, \mathbf{h}_1 \leftarrow \mathbb{F}_2^r$ of small weight w, $\mathbf{h}_0$ inversible. Compute $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

$$\mathbf{H}_{\text{secret}} = \left(\begin{array}{c|c} \mathbf{h}_0 & \mathbf{h}_1 \\ \circlearrowright & \circlearrowright \end{array} \right)$$

$$\mathbf{H}_{\text{pub}} = \left(\begin{array}{c|c} (1, 0, \ldots, 0) & \mathbf{h} \\ \circlearrowright & \circlearrowright \end{array} \right)$$

Encryption

As for McEliece, $\mathbf{e}$ of weight t,

$$\mathbf{c} = \mathbf{m}\mathbf{G} + \mathbf{e}.$$

Decryption

Use an iterative decoder (e.g. the BitFlipping algorithm) to recover message $\mathbf{m}$.

Suggested parameters: $r = 9857, n = 2r, w = 142, t = 134$ for 128 bits. Resulting sizes?

Code-based cryptography (CBC)

Avantage

Code-based cryptography (CBC)

Avantage

- Très efficace (algèbre linéaire)

Code-based cryptography (CBC)

Avantage

- Très efficace (algèbre linéaire)
- Arithmétique simple (modulo 2 vs modulo $2^{\sim 1024}$ pour RSA)

Code-based cryptography (CBC)

Avantage

- Très efficace (algèbre linéaire)
- Arithmétique simple (modulo 2 vs modulo $2^{\sim 1024}$ pour RSA)
- Hautement parallélisable

Code-based cryptography (CBC)

Avantage

- Très efficace (algèbre linéaire)
- Arithmétique simple (modulo 2 vs modulo $2^{\sim 1024}$ pour RSA)
- Hautement parallélisable

Inconvénients

- Taille de clés conséquente... (quasi-cyclique ?)

Code-based cryptography (CBC)

Avantage

- Très efficace (algèbre linéaire)
- Arithmétique simple (modulo 2 vs modulo $2^{\sim 1024}$ pour RSA)
- Hautement parallélisable

Inconvénients

- Taille de clés conséquente... (quasi-cyclique ?)
- Hypothèse d'indistingabilité de la famille de codes utilisée (plus technique)

Code-based cryptography (CBC)

Avantage

- Très efficace (algèbre linéaire)
- Arithmétique simple (modulo 2 vs modulo $2^{\sim 1024}$ pour RSA)
- Hautement parallélisable

Inconvénients

- Taille de clés conséquente... (quasi-cyclique ?)
- Hypothèse d'indistingabilité de la famille de codes utilisée (plus technique)

Chiffrement OK. Existe-t-il un algo de signature aussi simple ?

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w

Clé publique $\mathbf{H}$ et le syndrome de la clé secrète $\mathbf{s} = \mathbf{H}\mathbf{x}^\top$

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w
Clé publique $\mathbf{H}$ et le syndrome de la clé secrète $\mathbf{s} = \mathbf{H}\mathbf{x}^\top$

message m

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w
Clé publique $\mathbf{H}$ et le syndrome de la clé secrète $\mathbf{s} = \mathbf{H}\mathbf{x}^\top$

message m
$\mathbf{y}$ de poids faible

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w
Clé publique $\mathbf{H}$ et le syndrome de la clé secrète $\mathbf{s} = \mathbf{H}\mathbf{x}^{\top}$

message m
$\mathbf{y}$ de poids faible
$\mathbf{c} = \mathcal{H}\left(\mathbf{H}\mathbf{y}^{\top}, \mathbf{m}\right)$ de poids faible

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w
Clé publique $\mathbf{H}$ et le syndrome de la clé secrète $\mathbf{s} = \mathbf{H}\mathbf{x}^\top$

message m

$\mathbf{y}$ de poids faible

$\mathbf{c} = \mathcal{H}\left(\mathbf{H}\mathbf{y}^\top, \mathbf{m}\right)$ de poids faible

$$\mathbf{z} = \mathbf{x} \cdot \mathbf{c} + \mathbf{y}$$

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w

Clé publique $\mathbf{H}$ et le syndrome de la clé secrète $\mathbf{s} = \mathbf{H}\mathbf{x}^\top$

message m

$\mathbf{y}$ de poids faible

$\mathbf{c} = \mathcal{H}\left(\mathbf{H}\mathbf{y}^\top, \mathbf{m}\right)$ de poids faible

$$\mathbf{z} = \mathbf{x} \cdot \mathbf{c} + \mathbf{y} \qquad \xrightarrow{\mathbf{z},\mathbf{c}}$$

Code-based cryptography (CBC) : Exemple de signature efficace

Clé secrète $\mathbf{x}$ de poids faible w
Clé publique $\mathbf{H}$ et le syndrome de la clé secrète $\mathbf{s} = \mathbf{H}\mathbf{x}^\top$

message m
$\mathbf{y}$ de poids faible
$\mathbf{c} = \mathcal{H}\left(\mathbf{H}\mathbf{y}^\top, \mathbf{m}\right)$ de poids faible

$$\mathbf{z} = \mathbf{x} \cdot \mathbf{c} + \mathbf{y} \quad \xrightarrow{\mathbf{z},\mathbf{c}} \quad \text{Verif ?}$$

Code-based cryptography (CBC) : Exemple de signature efficace

Verif :

Code-based cryptography (CBC) : Exemple de signature efficace

Verif :

- $wt\left(\mathbf{z}\right) \leq \tilde{w}$ pas trop grand

Code-based cryptography (CBC) : Exemple de signature efficace

Verif :

- $wt\left(\mathbf{z}\right) \leq \tilde{w}$ pas trop grand
- Vérifier que $\mathcal{H}\left(\mathbf{H}\mathbf{z}^{\top} - \mathbf{s}\cdot\mathbf{c}, \mathbf{m}\right) == \mathbf{c}$

Code-based cryptography (CBC) : Exemple de signature efficace

Verif :

- $wt\left(\mathbf{z}\right) \leq \tilde{w}$ pas trop grand
- Vérifier que $\mathcal{H}\left(\mathbf{H}\mathbf{z}^{\top} - \mathbf{s} \cdot \mathbf{c}, \mathbf{m}\right) == \mathbf{c}$

En théorie, ça fonctionne. Mais en pratique...

Code-based cryptography (CBC) : Exemple de signature efficace

Verif :

- $wt(\mathbf{z}) \leq \tilde{w}$ pas trop grand
- Vérifier que $\mathcal{H}\left(\mathbf{H}\mathbf{z}^{\top} - \mathbf{s} \cdot \mathbf{c}, \mathbf{m}\right) == \mathbf{c}$

En théorie, ça fonctionne. Mais en pratique...

Le problème peut s'écrire sous forme d'un décodage de syndrome :

Code-based cryptography (CBC) : Exemple de signature efficace

Verif :

- $wt(\mathbf{z}) \leq \tilde{w}$ pas trop grand
- Vérifier que $\mathcal{H}\left(\mathbf{Hz}^{\top} - \mathbf{s} \cdot \mathbf{c}, \mathbf{m}\right) == \mathbf{c}$

En théorie, ça fonctionne. Mais en pratique...

Le problème peut s'écrire sous forme d'un décodage de syndrome :

$$\mathbf{z} = \left(\begin{array}{cccc|cccc} 1 & 0 & \dots & 0 & c_0 & c_1 & \dots & c_{n-1} \\ 0 & 1 & \dots & 0 & c_{n-1} & c_0 & \dots & c_{n-2} \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & 1 & c_1 & c_2 & \dots & c_0 \end{array}\right) \cdot \begin{pmatrix} \mathbf{y} \\ \mathbf{x} \end{pmatrix}$$

Code-based cryptography (CBC) : Exemple de signature efficace

Syndrome connu, matrice de parité creuse connue (LDPC)

Code-based cryptography (CBC) : Exemple de signature efficace

Syndrome connu, matrice de parité creuse connue (LDPC)
⇒ décodage classique facile et efficace

Code-based cryptography (CBC) : Exemple de signature efficace

Syndrome connu, matrice de parité creuse connue (LDPC)

⇒ décodage classique facile et efficace

⇒ cryptanalyse

Code-based cryptography (CBC) : Exemple de signature efficace

Syndrome connu, matrice de parité creuse connue (LDPC)

⇒ décodage classique facile et efficace

⇒ cryptanalyse

Claimed security	Persichetti's OTS parameters				xBF parameters		Verification	Cryptanalysis
	n	w_1	w_2	δ	τ	N	t_{verify} (ms)	t_{break} (ms)
80	4801	90	100	10	7	5	22.569	165.459
	3072	85	85	7	5	5	14.271	68.858
128	9857	150	200	12	9	10	99.492	453.680
	6272	125	125	10	7	10	42.957	288.442

Code-based cryptography (CBC) : Exemple de signature efficace

Syndrome connu, matrice de parité creuse connue (LDPC)
⇒ décodage classique facile et efficace
⇒ cryptanalyse

Claimed security	Persichetti's OTS parameters				xBF parameters		Verification	Cryptanalysis
	n	w_1	w_2	δ	τ	N	t_{verify} (ms)	t_{break} (ms)
80	4801	90	100	10	7	5	22.569	165.459
	3072	85	85	7	5	5	14.271	68.858
128	9857	150	200	12	9	10	99.492	453.680
	6272	125	125	10	7	10	42.957	288.442

D'autres schémas de signature (plus complexes à exposer) existent, et ne souffrent pas de ce type de problème:

- WAVE [?]: `https://eprint.iacr.org/2018/996`
- DURANDAL [?]: `https://eprint.iacr.org/2018/1192` **(métrique rang)**

Rank-based cryptography

D'autres métriques existent en codes correcteurs d'erreur. Par exemple, la métrique rang.

Rank-based cryptography

D'autres métriques existent en codes correcteurs d'erreur. Par exemple, la métrique rang. Soit $\mathbb{F}_q$ le corps fini à q éléments, et $\mathbb{F}_{q^m}$ une extension de degré m sur $\mathbb{F}_q$.

Rank-based cryptography

D'autres métriques existent en codes correcteurs d'erreur. Par exemple, la métrique rang.
Soit $\mathbb{F}_q$ le corps fini à q éléments, et $\mathbb{F}_{q^m}$ une extension de degré m sur $\mathbb{F}_q$.
Soit $(\mathbf{b}_0, \ldots, \mathbf{b}_{m-1})$ une base de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$.

Rank-based cryptography

D'autres métriques existent en codes correcteurs d'erreur. Par exemple, la métrique rang.
Soit $\mathbb{F}_q$ le corps fini à q éléments, et $\mathbb{F}_{q^m}$ une extension de degré m sur $\mathbb{F}_q$.
Soit $(\mathbf{b}_0, \ldots, \mathbf{b}_{m-1})$ une base de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$.

$$\mathbf{v} = \begin{pmatrix} v_0 & v_1 & \ldots & v_{n-1} \end{pmatrix} \in \mathbb{F}_{q^m}^n$$

$$\mathbf{V} = \begin{pmatrix} v_{0,0} & v_{1,0} & \ldots & v_{n-1,0} \\ v_{0,0} & v_{1,0} & \ldots & v_{n-1,0} \\ \vdots & & \ddots & \vdots \\ v_{0,m-1} & v_{1,m-1} & \ldots & v_{n-1,m-1} \end{pmatrix} \in \mathbb{F}_q^{m \times n} \begin{pmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_{m-1} \end{pmatrix}$$

Rank-based cryptography

D'autres métriques existent en codes correcteurs d'erreur. Par exemple, la métrique rang.
Soit $\mathbb{F}_q$ le corps fini à q éléments, et $\mathbb{F}_{q^m}$ une extension de degré m sur $\mathbb{F}_q$.
Soit $(\mathbf{b}_0, \dots, \mathbf{b}_{m-1})$ une base de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$.

$$\mathbf{v} = \begin{pmatrix} v_0 & v_1 & \dots & v_{n-1} \end{pmatrix} \in \mathbb{F}_{q^m}^n$$

$$\mathbf{V} = \begin{pmatrix} v_{0,0} & v_{1,0} & \dots & v_{n-1,0} \\ v_{0,0} & v_{1,0} & \dots & v_{n-1,0} \\ \vdots & & \ddots & \vdots \\ v_{0,m-1} & v_{1,m-1} & \dots & v_{n-1,m-1} \end{pmatrix} \in \mathbb{F}_q^{m\times n} \begin{pmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_{m-1} \end{pmatrix}$$

Le poids rang du vecteur $\mathbf{v}$ est défini comme le rang de la matrice $\mathbf{V}$

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \dots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

$$\mathbf{v} = \begin{pmatrix} 1 + 4\alpha + 2\alpha^2 & 2 + 3\alpha & 3 + 2\alpha + 2\alpha^2 \end{pmatrix}$$

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

$$\mathbf{v} = \begin{pmatrix} 1 + 4\alpha + 2\alpha^2 & 2 + 3\alpha & 3 + 2\alpha + 2\alpha^2 \end{pmatrix}$$

$$\mathbf{V} = \begin{array}{c} \left(\begin{array}{ccc} 1 & 2 & 3 \\ 4 & 3 & 2 \\ 2 & 0 & 2 \end{array}\right) \end{array} \begin{array}{c} 1 \\ \alpha \\ \alpha^2 \end{array}$$

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \dots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

$$\mathbf{v} = \begin{pmatrix} 1 + 4\alpha + 2\alpha^2 & 2 + 3\alpha & 3 + 2\alpha + 2\alpha^2 \end{pmatrix}$$

$$\mathbf{V} = \begin{array}{c} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 3 & 2 \\ 2 & 0 & 2 \end{pmatrix} \end{array} \begin{array}{c} 1 \\ \alpha \\ \alpha^2 \end{array}$$

Définitions

Rang d'un vecteur $\mathbf{v} \in \mathbb{F}_{q^m}^n$:

→ rang de la matrice ainsi obtenue

(ne dépend pas de la base choisie)

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

$$\mathbf{v} = \begin{pmatrix} 1 + 4\alpha + 2\alpha^2 & 2 + 3\alpha & 3 + 2\alpha + 2\alpha^2 \end{pmatrix}$$

$$\mathbf{V} = \begin{array}{c} \left(\begin{array}{ccc} 1 & 2 & 3 \\ 0 & 0 & 0 \\ 2 & 0 & 2 \end{array}\right) \end{array} \begin{array}{l} 1 \\ \alpha \\ \alpha^2 \end{array}$$

Définitions

Rang d'un vecteur $\mathbf{v} \in \mathbb{F}_{q^m}^n$:

→ rang de la matrice ainsi obtenue

(ne dépend pas de la base choisie)

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

$$\mathbf{v} = \begin{pmatrix} 1 + 4\alpha + 2\alpha^2 & 2 + 3\alpha & 3 + 2\alpha + 2\alpha^2 \end{pmatrix}$$

$$\mathbf{V} = \begin{array}{c} \left(\begin{array}{ccc} 1 & 2 & 3 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{array} \right) \end{array} \begin{array}{c} 1 \\ \alpha \\ \alpha^2 \end{array}$$

Définitions

Rang d'un vecteur $\mathbf{v} \in \mathbb{F}_{q^m}^n$:

→ rang de la matrice ainsi obtenue

(ne dépend pas de la base choisie)

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

$$\mathbf{v} = \begin{pmatrix} 1 + 4\alpha + 2\alpha^2 & 2 + 3\alpha & 3 + 2\alpha + 2\alpha^2 \end{pmatrix}$$

$$\mathbf{V} = \begin{array}{c} \begin{pmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} \end{array} \begin{array}{c} 1 \\ \alpha \\ \alpha^2 \end{array}$$

→ $\text{rang}(\mathbf{v}) = 2$

Définitions

Rang d'un vecteur $\mathbf{v} \in \mathbb{F}_{q^m}^n$:

→ rang de la matrice ainsi obtenue

(ne dépend pas de la base choisie)

Métrique Rang

- Extension de corps fini $\mathbb{F}_{q^m}$, disons $\mathbb{F}_{5^3}$
- Base $(b_1, \ldots, b_m)$ de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$

→ disons $(1, \alpha, \alpha^2)$, α racine de $X^3 + X + 1$, $\mathbb{F}_{5^3} \cong \mathbb{F}_5/(X^3 + X + 1)$

- $\mathbf{v} \in \mathbb{F}_{q^m}^n$ s'écrit comme une matrice $\mathbf{V} \in \mathbb{F}_q^{m \times n}$ dans cette base

Exemple avec $q = 5$, $m = 3$, et $n = 3$:

$$\mathbf{v} = \begin{pmatrix} 1 + 4\alpha + 2\alpha^2 & 2 + 3\alpha & 3 + 2\alpha + 2\alpha^2 \end{pmatrix}$$

$$\mathbf{V} = \begin{array}{cc} \begin{pmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} & \begin{array}{c} 1 \\ \alpha \\ \alpha^2 \end{array} \end{array}$$

→ $\text{rang}(\mathbf{v}) = 2$

Définitions

Rang d'un vecteur $\mathbf{v} \in \mathbb{F}_{q^m}^n$:

→ rang de la matrice ainsi obtenue

(ne dépend pas de la base choisie)

Distance rang entre deux vecteurs $\mathbf{u}, \mathbf{v} \in \mathbb{F}_{q^m}^n$:

→ $d_R(\mathbf{u}, \mathbf{v}) = \text{rang}(\mathbf{U} - \mathbf{V})$

(symétrie, séparation, inégalité triangulaire)

Attaques sur ces Métriques

Idée : Compter le nombre de mots possibles de longueur n et de poids t

Attaques sur ces Métriques

Idée : Compter le nombre de mots possibles de longueur n et de poids t

Hamming : nombre d'ensembles à t éléments parmi les ensembles à n éléments : binôme de Newton $\binom{n}{t}$ $(\leq 2^n)$

Attaques sur ces Métriques

Idée : Compter le nombre de mots possibles de longueur n et de poids t

Hamming : nombre d'ensembles à t éléments parmi les ensembles à n éléments : binôme de Newton $\binom{n}{t}$ $(\leq 2^n)$

Rank : nombre de sous-espaces vectoriels de dimension t sur $\mathbb{F}_q$ dans un espace de dimension n sur $\mathbb{F}_{q^m}$: binôme de Gauss $\begin{bmatrix} n \\ t \end{bmatrix}_q (\sim q^{t(n-t)})$

Attaques sur ces Métriques

Idée : Compter le nombre de mots possibles de longueur n et de poids t

Hamming : nombre d'ensembles à t éléments parmi les ensembles à n éléments : binôme de Newton $\binom{n}{t}$ $(\leq 2^n)$

Rank : nombre de sous-espaces vectoriels de dimension t sur $\mathbb{F}_q$ dans un espace de dimension n sur $\mathbb{F}_{q^m}$: binôme de Gauss $\begin{bmatrix} n \\ t \end{bmatrix}_q (\sim q^{t(n-t)})$

En résumé: les attaques en **métrique Rang** ont une complexité **quadratiquement** exponentielle $2^{\mathcal{O}(n^2)}$, contre **simplement** exponentielle $2^{\mathcal{O}(n)}$ pour la **métrique de Hamming**

Rank-based cryptography

Avantages

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
  - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
  - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
  - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
    - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
    - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)
- Les tailles de clés sont donc plus petites

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
  - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
  - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)
- Les tailles de clés sont donc plus petites

Inconvénients

- Les opérations sont plus complexes

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
    - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
    - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)
- Les tailles de clés sont donc plus petites

Inconvénients

- Les opérations sont plus complexes
    - Arithmétique dans des extensions de corps finis

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
  - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
  - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)
- Les tailles de clés sont donc plus petites

Inconvénients

- Les opérations sont plus complexes
  - Arithmétique dans des extensions de corps finis
- La métrique est moins intuitive

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
  - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
  - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)
- Les tailles de clés sont donc plus petites

Inconvénients

- Les opérations sont plus complexes
  - Arithmétique dans des extensions de corps finis
- La métrique est moins intuitive
  - Moins de gens s'y intéressent, les schémas sont moins étudiés/éprouvés

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
    - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
    - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)
- Les tailles de clés sont donc plus petites

Inconvénients

- Les opérations sont plus complexes
    - Arithmétique dans des extensions de corps finis
- La métrique est moins intuitive
    - Moins de gens s'y intéressent, les schémas sont moins étudiés/éprouvés
- Des attaques structurelles sont plus facilement exploitables

Rank-based cryptography

Avantages

- Complexité du problème de décodage par syndrome plus élevé
  - Quadratiquement exponentiel au lieu de simplement exponentiel pour la métrique de Hamming
  - Ceci est dû au plus grand nombre de mots ayant le même support (espace vectoriel)
- Les tailles de clés sont donc plus petites

Inconvénients

- Les opérations sont plus complexes
  - Arithmétique dans des extensions de corps finis
- La métrique est moins intuitive
  - Moins de gens s'y intéressent, les schémas sont moins étudiés/éprouvés
- Des attaques structurelles sont plus facilement exploitables
  - Bases de Gröbner

Outline

6 Post-quantum cryptography
- Lattice-based cryptography
- Code-based cryptography
- Hash-based cryptography

Outline

1 What you've learnt so far (should have)

2 Classical vs Quantum computing

3 Two noticeable quantum algorithms (and their impact over cryptography)

4 State-of-the-art quantum computers

5 Possible alternatives

6 Post-quantum cryptography

7 Conclusion

Course conclusion

- Cryptography has reached a stable phase, where:

Course conclusion

- Cryptography has reached a stable phase, where:
    - Symmetric primitives are fast and secure (sufficiently attacked to be considered as such)

Course conclusion

- Cryptography has reached a stable phase, where:
  - Symmetric primitives are fast and secure (sufficiently attacked to be considered as such)
  - Asymmetric primitives have been sufficiently improved to be largely deployed

Course conclusion

- Cryptography has reached a stable phase, where:
  - Symmetric primitives are fast and secure (sufficiently attacked to be considered as such)
  - Asymmetric primitives have been sufficiently improved to be largely deployed
  - Hybrid encryption allows to benefit from SE efficiency while avoiding its disadvantages

Course conclusion

- Cryptography has reached a stable phase, where:
    - Symmetric primitives are fast and secure (sufficiently attacked to be considered as such)
    - Asymmetric primitives have been sufficiently improved to be largely deployed
    - Hybrid encryption allows to benefit from SE efficiency while avoiding its disadvantages
- There is a real quantum threat for actual cryptography.

Course conclusion

- Cryptography has reached a stable phase, where:
  - Symmetric primitives are fast and secure (sufficiently attacked to be considered as such)
  - Asymmetric primitives have been sufficiently improved to be largely deployed
  - Hybrid encryption allows to benefit from SE efficiency while avoiding its disadvantages
- There is a real quantum threat for actual cryptography.
- Post-quantum alternatives exist and are being developed/standardized (involve classical computers, not quantum).

Course conclusion

- Cryptography has reached a stable phase, where:
    - Symmetric primitives are fast and secure (sufficiently attacked to be considered as such)
    - Asymmetric primitives have been sufficiently improved to be largely deployed
    - Hybrid encryption allows to benefit from SE efficiency while avoiding its disadvantages
- There is a real quantum threat for actual cryptography.
- Post-quantum alternatives exist and are being developed/standardized (involve classical computers, not quantum).
- Most serious candidates are lattices, error correcting codes and hash functions.

Course conclusion

- Cryptography has reached a stable phase, where:
  - Symmetric primitives are fast and secure (sufficiently attacked to be considered as such)
  - Asymmetric primitives have been sufficiently improved to be largely deployed
  - Hybrid encryption allows to benefit from SE efficiency while avoiding its disadvantages
- There is a real quantum threat for actual cryptography.
- Post-quantum alternatives exist and are being developed/standardized (involve classical computers, not quantum).
- Most serious candidates are lattices, error correcting codes and hash functions.
- Keys for symmetric algorithms need to be doubled.

Upgoing developments for PQC

- NIST PQC standards should be ready by 2024.

Upgoing developments for PQC

- NIST PQC standards should be ready by 2024.
- Integration and deployment will probably take another 5-10 years

Upgoing developments for PQC

- NIST PQC standards should be ready by 2024.
- Integration and deployment will probably take another 5-10 years
- Most robust proposals will be standardized sooner, probably featuring:

Upgoing developments for PQC

- NIST PQC standards should be ready by 2024.
- Integration and deployment will probably take another 5-10 years
- Most robust proposals will be standardized sooner, probably featuring:
    - Classic McEliece (code) encryption
    - 1 lattice encryption & signature
    - 1 multivariate signature?

Upgoing developments for PQC

- NIST PQC standards should be ready by 2024.
- Integration and deployment will probably take another 5-10 years
- Most robust proposals will be standardized sooner, probably featuring:
  - Classic McEliece (code) encryption
  - 1 lattice encryption & signature
  - 1 multivariate signature?
- More practical candidates will come after

Upgoing developments for PQC

- NIST PQC standards should be ready by 2024.
- Integration and deployment will probably take another 5-10 years
- Most robust proposals will be standardized sooner, probably featuring:
  - Classic McEliece (code) encryption
  - 1 lattice encryption & signature
  - 1 multivariate signature?
- More practical candidates will come after

Part of your future job might consist in integrating/improving these schemes!