An introduction to Post-Quantum Cryptography (PQC)

Jean-Christophe Deneuville

<jean-christophe.deneuville@enac.fr>

Fall 2019

TLS-SEC

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. Quantum safe alternatives

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. Quantum safe alternatives

What you've learnt so far (should have)

- What is cryptography

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)

What you’ve learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)
- (Collision Resistant) Hash Functions (and birthday paradox)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)
- (Collision Resistant) Hash Functions (and birthday paradox)
- Digital Signatures (RSA, ECDSA, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)
- (Collision Resistant) Hash Functions (and birthday paradox)
- Digital Signatures (RSA, ECDSA, ...)
- Security models

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)

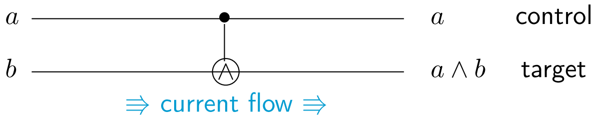
4 State-of-the-art quantum computers

5 Quantum safe alternatives

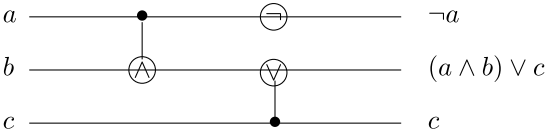
Classical Boolean Circuits

source: J. Royer

We view them as naming maps $\{0,1\}^n \to \{0,1\}^n$



Now consider



We can describe this by either of:

- $b \leftarrow a \wedge b; \quad a \leftarrow \neg a; \quad b \leftarrow b \vee c$
- $|a, b, c\rangle \;\mapsto\; |a, a \wedge b, c\rangle \;\mapsto\; |\neg a, a \wedge b, c\rangle \;\mapsto\; |\neg a, (a \wedge b) \vee c, c\rangle$

$|x, y, z\rangle$ = *state vector*

Classical computing

A classical computer (Turing machine) processes (through a language) classical boolean circuits.

The quantity of information is measured through Shannon's entropy, data can eventually be compressed, and there exist efficient algorithms for error correction.

Some circuits are computable *i.e.* the machine eventually halts (*e.g.* primality problem), some others aren't (*e.g.* the halting problem).

Current security

Current security vs. classical computing power (2019)

Current security vs. classical computing power (2019)

1 standard machine: 64 bits architecture
2^6

Current security vs. classical computing power (2019)

1 standard machine: 8 cores

$2^6 \times 2^4$

Current security vs. classical computing power (2019)

1 standard machine: 4 GHz

$2^6 \times 2^4 \times 2^2 \times 10^9$

Current security vs. classical computing power (2019)

1 standard machine: running 1 month

$2^6 \times 2^4 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30$

Current security vs. classical computing power (2019)

NSA $\geq$ 10 000 standard machines?

$$2^6 \times 2^4 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^5$$

Current security vs. classical computing power (2019)

NSA $\geq$ 10 000 standard machines?

$2^6 \times 2^4 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^5 \approx 2^{80}$ elementary operations

Current security vs. classical computing power (2019)

NSA $\geq$ 10 000 standard machines? (without possible GPU, ASICS, ...)

$2^6 \times 2^4 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^5 \approx 2^{80}$ elementary operations

Current security vs. classical computing power (2019)

NSA $\geq$ 10 000 standard machines? (without possible GPU, ASICS, ...)

$2^6 \times 2^4 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^5 \approx 2^{80}$ elementary operations

A concrete example

During 2018, there were 2^{89} SHA-256 hashes computed on the blockchain BitCoin...

Current security vs. classical computing power (2019)

NSA $\geq$ 10 000 standard machines? (without possible GPU, ASICS, ...)

$2^6 \times 2^4 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^5 \approx 2^{80}$ elementary operations

A concrete example

During 2018, there were 2^{89} SHA-256 hashes computed on the blockchain BitCoin...

Security in 2019

Setting parameters so that best known attacks have complexity (at least) 2^{128}.

Current security vs. classical computing power (2019)

NSA $\geq$ 10 000 standard machines? (without possible GPU, ASICS, ...)

$2^6 \times 2^4 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^5 \approx 2^{80}$ elementary operations

A concrete example

During 2018, there were 2^{89} SHA-256 hashes computed on the blockchain BitCoin...

Security in 2019

Setting parameters so that best known attacks have complexity (at least) 2^{128}.

Classical best known attacks:

- Symmetric primitives: brute-force
- Asymmetric primitives: GNFS, sub-exponential complexity

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

- Superposition: while a bit can be either in a state 0 or 1, a quantum bit (*qubit*) can be in any *superposition* of states $|0\rangle$ and $|1\rangle$.

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

- Superposition: while a bit can be either in a state 0 or 1, a quantum bit (*qubit*) can be in any *superposition* of states $|0\rangle$ and $|1\rangle$.
- Entanglement: the capability of two qubits to be *correlated*. If Alice and Bob both get one of two entangled qubits, and if Alice measures a $|0\rangle$ at some point, then necessarily Bob must measure the same, as $|00\rangle$ is the only state where Alice's qubit is a $|0\rangle$.

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

- Superposition: while a bit can be either in a state 0 or 1, a quantum bit (*qubit*) can be in any *superposition* of states $|0\rangle$ and $|1\rangle$.
- Entanglement: the capability of two qubits to be *correlated*. If Alice and Bob both get one of two entangled qubits, and if Alice measures a $|0\rangle$ at some point, then necessarily Bob must measure the same, as $|00\rangle$ is the only state where Alice's qubit is a $|0\rangle$.

Qubits can be "implemented" using the spin of an electron, or the polarization of a photon, ...

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 Quantum safe alternatives

Shor's algorithm

SIAM J. COMPUT.
Vol. 26, No. 5, pp. 1484–1509, October 1997
© 1997 Society for Industrial and Applied Mathematics
009

POLYNOMIAL-TIME ALGORITHMS FOR PRIME FACTORIZATION AND DISCRETE LOGARITHMS ON A QUANTUM COMPUTER*

PETER W. SHOR†

Abstract. A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

1 Pick $g \in \mathbb{Z}_N$ at random;

Shor’s algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

Pick $g \in \mathbb{Z}_N$ at random;

if $\gcd(g, N) \neq 1$ **then**

then **return** $(p = \gcd(g, N), q = N/p)$

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

```
Pick g ∈ Z_N at random;
if gcd(g, N) ≠ 1 then
    then return (p = gcd(g, N), q = N/p)
Find r such that g^r ≡ 1[N];
```

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

```
1 Pick g ∈ Z_N at random;
2 if gcd(g, N) ≠ 1 then
3     then return (p = gcd(g, N), q = N/p)
4 Find r such that g^r ≡ 1[N];
5 if r ≡ 0[2] then
6     return gcd(g^{r/2} ± 1, N)
7 else
8     go to 1
```

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$g^r \equiv 1[N] \quad \Leftrightarrow \quad \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1$$

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$\begin{aligned} g^r \equiv 1[N] \quad & \Leftrightarrow \quad \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1 \\ & \Leftrightarrow \quad g^r - 1 = kN \end{aligned}$$

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$\begin{aligned} g^r \equiv 1[N] &\Leftrightarrow \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1 \\ &\Leftrightarrow g^r - 1 = kN \\ \textbf{(assuming } r \textbf{ is even)} \quad &\Leftrightarrow \left(g^{r/2} - 1\right)\left(g^{r/2} + 1\right) = kN \end{aligned}$$

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$
\begin{aligned}
g^r \equiv 1[N] \quad &\Leftrightarrow \quad \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1 \\
&\Leftrightarrow \quad g^r - 1 = kN \\
\textbf{(assuming } r \textbf{ is even)} \quad &\Leftrightarrow \quad \left(g^{r/2} - 1\right)\left(g^{r/2} + 1\right) = kN
\end{aligned}
$$

Meaning that there is a non-negligible probability that $g^{r/2} \pm 1$ shares non trivial factors with N.

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1
- step 1. $g \leftarrow 127$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1
- step 1. $g \leftarrow 127$
- step 2. $r \leftarrow 17388$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q **(source: minutephysics)**

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1
- step 1. $g \leftarrow 127$
- step 2. $r \leftarrow 17388$
- step 3. let us denote $g_p = g^{17388/2} + 1$ and $g_q = g^{17388/2} - 1$
  we have that $\gcd(g_p, N) = 829 =: p$ and $\gcd(g_q, N) = 379 =: q$
  and indeed, $p \cdot q = 829 \times 379 = 314191 = N$

Shor's algorithm: how it works

- Second question: Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?

Shor's algorithm: how it works

- Second question: Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?

"Find r such that $g^r \equiv 1[N]$;"

Shor's algorithm: how it works

- Second question: Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

Shor's algorithm: how it works

- Second question: Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

Shor's algorithm: how it works

- Second question: Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

- Classically $\mathcal{O}(N)$

Shor's algorithm: how it works

- Second question: Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

- Classically $\mathcal{O}(N)$
- Quantumly $\mathcal{O}\left(\log(N)^3\right)$

Shor's algorithm: how it works

- Second question: Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

- Classically $\mathcal{O}(N)$
- Quantumly $\mathcal{O}\left(\log(N)^3\right)$. That's an **exponential** speedup!

Quantum period finding

How does it work? Why is it much much faster quantumly?

Fourier Transform is THE tool to analyse frequencies. Fortunately, it has a quantum equivalent: QFT.

Quantum computing allows to provide QFT a superposition of every possible states (assuming enough qubits).

Consequences of Shor's algorithm on PKC

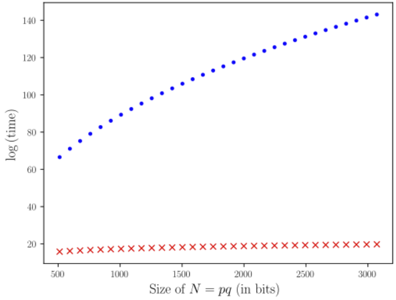
- Factoring becomes polynomial-time

Consequences of Shor's algorithm on PKC

- Factoring becomes polynomial-time
  $\mathcal{O}\left((\log N)^2 (\log\log N) (\log\log\log N)\right)$

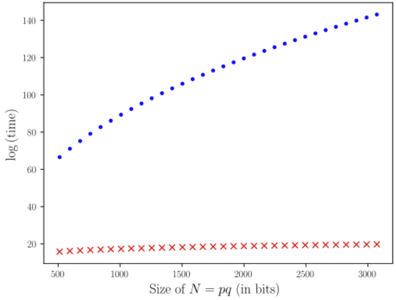
Consequences of Shor's algorithm on PKC

- Factoring becomes polynomial-time
  $\mathcal{O}\left((\log N)^2 (\log\log N)(\log\log\log N)\right)$



Consequences of Shor's algorithm on PKC

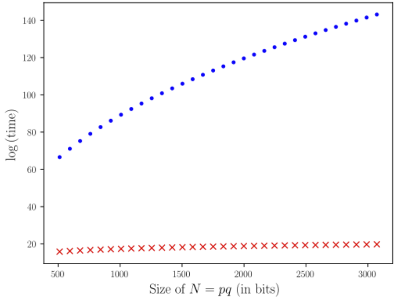
- Factoring becomes polynomial-time $\mathcal{O}\left((\log N)^2 (\log\log N)(\log\log\log N)\right)$
- Discrete logarithm becomes almost polynomial-time



Consequences of Shor's algorithm on PKC

- Factoring becomes polynomial-time $\mathcal{O}\left((\log N)^2 (\log\log N)(\log\log\log N)\right)$
- Discrete logarithm becomes almost polynomial-time

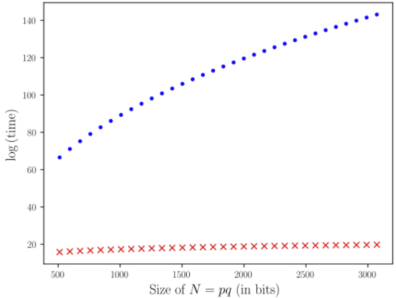
No more RSA, DSA, ECDSA, ElGamal, ...



Consequences of Shor's algorithm on PKC

- Factoring becomes polynomial-time
  $\mathcal{O}\left((\log N)^2 (\log \log N) (\log \log \log N)\right)$
- Discrete logarithm becomes almost polynomial-time

No more RSA, DSA, ECDSA, ElGamal, ...



In other words, **security as we know it collapses**...

Grover's algorithm

A fast quantum mechanical algorithm for database search

Lov K. Grover
3C-404A, Bell Labs
600 Mountain Avenue
Murray Hill NJ 07974
lkgrover@bell-labs.com

Summary

Imagine a phone directory containing N names arranged in completely random order. In order to find someone's phone number with a probability of $\frac{1}{2}$, any classical algorithm (whether deterministic or probabilistic) will need to look at a minimum of $\frac{N}{2}$ names. Quantum mechanical systems can be in a superposition of states and simultaneously examine multiple names. By properly adjusting the phases of various operations, successful computations reinforce each other while others interfere randomly. As a result, the desired phone number can be obtained in only $O(\sqrt{N})$ steps. The algorithm is within a small constant factor of the fastest possible quantum mechanical algorithm.

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}(\sqrt{n})$ queries instead of $\mathcal{O}(n)$.

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}(\sqrt{n})$ queries instead of $\mathcal{O}(n)$.

Consequence over symmetric crypto:

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}\left(\sqrt{n}\right)$ queries instead of $\mathcal{O}\left(n\right)$.

Consequence over symmetric crypto:

$\rightarrow$ The length of the secret key must be **doubled** to preserve the same level of security

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}\left(\sqrt{n}\right)$ queries instead of $\mathcal{O}\left(n\right)$.

Consequence over symmetric crypto:

$\rightarrow$ The length of the secret key must be **doubled** to preserve the same level of security

Consequence over hash functions:

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}\left(\sqrt{n}\right)$ queries instead of $\mathcal{O}\left(n\right)$.

Consequence over symmetric crypto:

→ The length of the secret key must be **doubled** to preserve the same level of security

Consequence over hash functions:

→ More tricky (depending on the model, the size of the quantum computer, . . .), at least $+33\%$ to preserve the security level

Outline

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

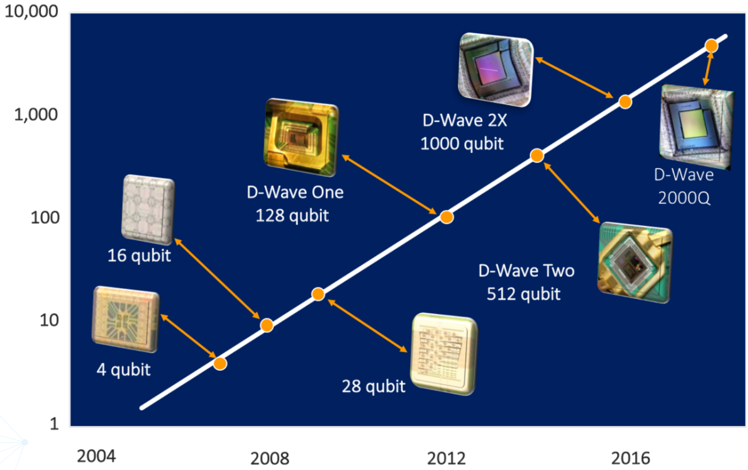
2 Classical vs Quantum computing

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

4 State-of-the-art quantum computers

5 Quantum safe alternatives

How far are we from a large-scale quantum computer?



A quantum analog to Moore's law: the number of qubits (y-axe) approximately doubles every year (x-axe). (Source: D-Wave)

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel
- fault-tolerance remains an open problem

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel
- fault-tolerance remains an open problem
- still far from what is required to factor 2048 bits moduli

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel
- fault-tolerance remains an open problem
- still far from what is required to factor 2048 bits moduli

In 2019, the largest quantum computer features 72 qubits (Google).

Hot news!

Hot news!

TECH • QUANTUM COMPUTING

Google Claims 'Quantum Supremacy,' Marking a Major Milestone in Computing

By Robert Hackett September 20, 2019

Hot news!

TECH • QUANTUM COMPUTING

Google Claims 'Quantum Supremacy,' Marking Major Milestone in Computing

By Robert Hackett September 20, 2019

Google's supposed milestone achievement became public last month when a preprint scientific paper accidentally leaked on the website of NASA, a collaborator, as *Fortune* reported at the time. Google has said nothing about the potentially historic experiment since then, lending credence to whispers that its researchers are bound to silence under the terms of a news embargo by a major science journal, unable to disclose more information until a certain date which is presumed to be imminent.

Hot news!

Google's supposed milesto…

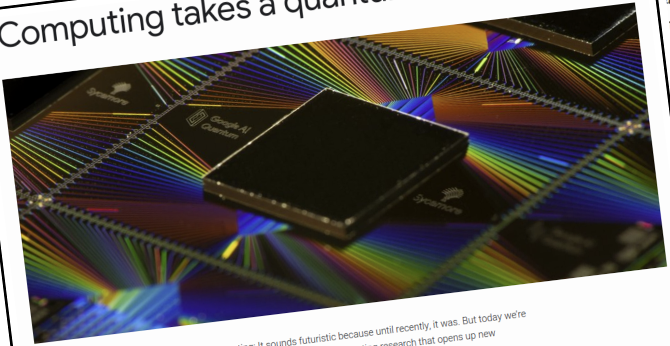
preprint…

…ecame public last month when a … on the website of NASA, a collaborator, … aid nothing about the potentially … nce to whispers that its researchers are … bargo by a major science journal, … ain date which is presumed to be

…rking

Google Cla… N…

Computing takes a quantum leap forward



Hartmut Neven
Engineering Director, Google AI Quantum Team

Published Oct 23, 2019

Quantum computing: It sounds futuristic because until recently, it was. But today we're marking a major milestone in quantum computing research that opens up new possibilities for this technology.

Unlike classical computing, which runs everything from your cell phone to a supercomputer, quantum computing is based on the properties of quantum mechanics. As a result, quantum computers could potentially solve problems that would be too difficult or even impossible for classical computers—like designing better batteries,

What is quantum supremacy?

What is quantum supremacy?

Quantum supremacy refers to the moment where a functional quantum computer can effectively solve a problem that is not solvable (within decent time frame, *e.g.* 100 years) with any (super) computer.

Wha

Quantum supre ntum computer can
effectively solve frame, *e.g.* 100 years) with
any (super) cor

Why is Google's quantum supremacy experiment impressive?

Asked 13 days ago Active 11 days ago Viewed 12k times

129

21

In the Nature paper published by Google, they say,

> To demonstrate quantum supremacy, we compare our quantum processor against state-of-the-art classical computers in the task of sampling the output of a pseudo-random quantum circuit. Random circuits are a suitable choice for benchmarking because they do not possess structure and therefore allow for limited guarantees of computational hardness. We design the circuits to entangle a set of quantum bits (qubits) by repeated application of single-qubit and two-qubit logical operations. Sampling the quantum circuit's output produces a set of bitstrings, for example {0000101, 1011100, …}. Owing to quantum interference, the probability distribution of the bitstrings resembles a speckled intensity pattern produced by light interference in laser scatter, such that some bitstrings are much more likely to occur than others. Classically computing this probability distribution becomes exponentially more difficult as the number of qubits (width) and number of gate cycles (depth) grow.

So, from what I can tell, they configure their qubits into a pseudo-randomly generated circuit, which, when run, puts the qubits into a state vector that represents a probability distribution over 2^{53} possible states of the qubits, but that distribution is intractable to calculate, or even estimate via sampling using a classical computer simulation. But they sample it by "looking" at the state of the qubits after running the circuit many times.

Isn't this just an example of creating a system whose output is intractable to calculate, and then "calculating" it by simply observing the output of the system?

It sounds similar to saying:

> If I spill this pudding cup on the floor, the exact pattern it will form is very chaotic, and intractable for any supercomputer to calculate. But I just invented a new special type of computer: this pudding cup. And I'm going to do the calculation by spilling it on the floor and observing the result. I have achieved pudding supremacy.

What i… ?

Quantum suprem… …antum computer can effectively solve a… …e frame, *e.g.* 100 years) with any (super) comp…

Why is Google's quantum supremacy experiment impressive?

Asked 13 days ago Active 11 days ago Viewed 12k times

129

21

In the Nature paper published by Google, they say,

> To demonstrate quantum supremacy, we compare our quantum processor against state-of-the-art classical computers in the task of sampling the output of a pseudo-random quantum circuit. Random circuits are a suitable choice for benchmarking because they do not possess structure and therefore allow for limited guarantees of computational hardness. We design the circuits to entangle a set of quantum bits (qubits) by repeated application of single-qubit and two-qubit logical operations. Sampling the quantum circuit's output produces a set of bitstrings, for example {0000101, 1011100, …}. Owing to quantum interference, the probability distribution of the bitstrings resembles a speckled intensity pattern produced by light interference in laser scatter, such that some bitstrings are much more likely to occur than others. Classically computing this probability distribution becomes exponentially more difficult as the number of qubits (width) and number of gate cycles (depth) grow.

So, from what I can tell, they configure their qubits into a pseudo-randomly generated circuit, which, when run, puts the qubits into a state vector that represents a probability distribution over 2^{53} possible states of the qubits, but that distribution is intractable to calculate, or even estimate via sampling using a classical computer simulation. But they sample it by "looking" at the state of the qubits after running the circuit many times.

Isn't this just an example of creating a system whose output is intractable to calculate, and then "calculating" it by simply observing the output of the system?

It sounds similar to saying:

> If I spill this pudding cup on the floor, the exact pattern it will form is very chaotic, and intractable for any supercomputer to calculate. But I just invented a new special type of computer: this pudding cup. And I'm going to do the calculation by spilling it on the floor and observing the result. I have achieved pudding supremacy.

What is quantum supremacy?

Quantum supremacy refers to the moment where a functional quantum computer can effectively solve a problem that is not solvable (within decent time frame, *e.g.* 100 years) with any (super) computer.

This result is a bit biased and overselled: it was obtain using a very specific (ad-hoc) problem that was purposely designed to behave much much better quantumly than classicaly...

What is quantum supremacy?

Quantum supremacy refers to the moment where a functional quantum computer can effectively solve a problem that is not solvable (within decent time frame, *e.g.* 100 years) with any (super) computer.

This result is a bit biased and oversellled: it was obtain using a very specific (ad-hoc) problem that was purposely designed to behave much much better quantumly than classicaly...

It however remains impressive, since no regular computer can do that efficiently. A bit weaker than supremacy is “quantum advantage”, where a quantum computer simply performs better than any computer.

Open challenges towards quantum computing

More work in required to embrace a large scale quantum computer:

Open challenges towards quantum computing

More work in required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing

Open challenges towards quantum computing

More work in required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories

Open challenges towards quantum computing

More work in required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- building architectures and interfaces between quantum computers and communication systems

Open challenges towards quantum computing

More work in required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- building architectures and interfaces between quantum computers and communication systems
- developing quantum programming languages, compilers and middle-ware stack

Open challenges towards quantum computing

More work in required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- building architectures and interfaces between quantum computers and communication systems
- developing quantum programming languages, compilers and middle-ware stack

Still, a Sword of Damocles hanging over our heads

Open challenges towards quantum computing

More work in required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- building architectures and interfaces between quantum computers and communication systems
- developing quantum programming languages, compilers and middle-ware stack

Still, a Sword of Damocles hanging over our heads, and **now** is the time for designing **quantum-safe** alternatives.

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 Quantum safe alternatives

Clarification

Quelles sont les alternatives à la cryptographie classique en présence d'un adversaire possédant un ordinateur quantique puissant ?

Clarification

Quelles sont les alternatives à la cryptographie classique en présence d'un adversaire possédant un ordinateur quantique puissant ?

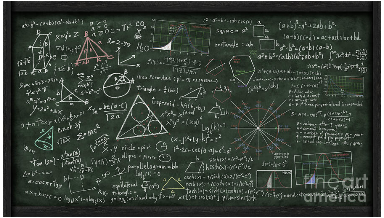
- Échange de clés quantiques (hors du cadre de cette présentation)

Clarification

Quelles sont les alternatives à la cryptographie classique en présence d'un adversaire possédant un ordinateur quantique puissant ?

- Échange de clés quantiques (hors du cadre de cette présentation)

- Cryptographie post-quantique



Cryptographie post-quantique

Quels sont les outils qui peuvent utiles à la conception de primitives post-quantiques ?

Cryptographie post-quantique

Quels sont les outils qui peuvent utiles à la conception de primitives post-quantiques ?

- Cryptographie fondée sur les réseaux euclidiens

Cryptographie post-quantique

Quels sont les outils qui peuvent utiles à la conception de primitives post-quantiques ?

- Cryptographie fondée sur les réseaux euclidiens
- Cryptographie fondée sur les codes correcteurs d'erreurs

Cryptographie post-quantique

Quels sont les outils qui peuvent utiles à la conception de primitives post-quantiques ?

- Cryptographie fondée sur les réseaux euclidiens
- Cryptographie fondée sur les codes correcteurs d'erreurs
- Cryptographie fondée sur les fonctions de hachage

Cryptographie post-quantique

Quels sont les outils qui peuvent utiles à la conception de primitives post-quantiques ?

- Cryptographie fondée sur les réseaux euclidiens
- Cryptographie fondée sur les codes correcteurs d'erreurs
- Cryptographie fondée sur les fonctions de hachage
- Cryptographie fondée sur les polynomes multivariés

Cryptographie post-quantique

Quels sont les outils qui peuvent utiles à la conception de primitives post-quantiques ?

- Cryptographie fondée sur les réseaux euclidiens
- Cryptographie fondée sur les codes correcteurs d'erreurs
- Cryptographie fondée sur les fonctions de hachage
- Cryptographie fondée sur les polynomes multivariés
- Cryptographie fondée sur les isogénies de courbes elliptiques

Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.

Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.
Cela inclus les primitives :

- d'échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.
Cela inclus les primitives :

- d'échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

En quelques chiffres (source NIST) :

Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.

Cela inclus les primitives :

- d'échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

En quelques chiffres (source NIST) :

- 82 soumissions reçues,

Processus de standardisation du NIST

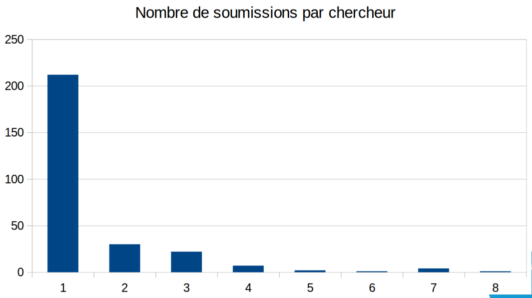
Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.

Cela inclus les primitives :

- d'échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

En quelques chiffres (source NIST) :

- 82 soumissions reçues,
- 278 chercheurs,



Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d’algorithmes cryptographiques post-quantiques.

Cela inclus les primitives :

- d’échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

En quelques chiffres (source NIST) :

- 82 soumissions reçues,
- 278 chercheurs,
- 25 pays (6 continents),

Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.

Cela inclus les primitives :

- d'échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

En quelques chiffres (source NIST) :

- 82 soumissions reçues,
- 278 chercheurs,
- 25 pays (6 continents),
- 69 qualifiées pour le premier tour,

Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.

Cela inclus les primitives :

- d'échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

En quelques chiffres (source NIST) :

- 82 soumissions reçues,
- 278 chercheurs,
- 25 pays (6 continents),
- 69 qualifiées pour le premier tour,
- Au 28/01 : 64 toujours en lice (dont 12 attaqués).

Processus de standardisation du NIST

Le NIST a lancé fin 2016 un appel à standardisation d'algorithmes cryptographiques post-quantiques.

Cela inclus les primitives :

- d'échange de clés (Key Encapsulation Mechanisms),
- de chiffrement,
- de signature.

En quelques chiffres (source NIST) :

- 82 soumissions reçues,
- 278 chercheurs,
- 25 pays (6 continents),
- 69 qualifiées pour le premier tour,
- Au 28/01 : 64 toujours en lice (dont 12 attaqués).
- Au 13/02 : 26 acceptées au 2nd tour.

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 Nominations 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

5 Quantum safe alternatives
- Lattice-based cryptography
- Hash-based cryptography
- Code-based cryptography

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

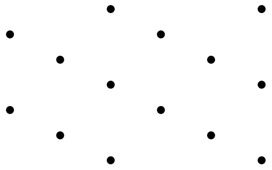
Lattice

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

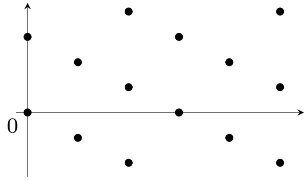
Lattice

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

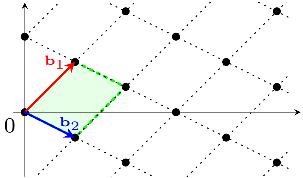
Lattice

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

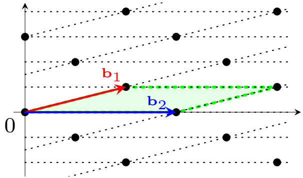
Lattice

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

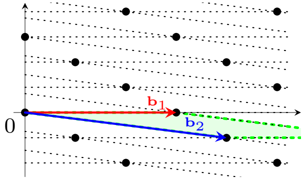
Lattice

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

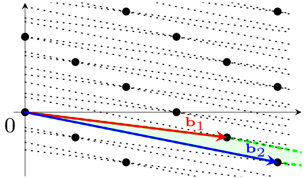
Lattice

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)**



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

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

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

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

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

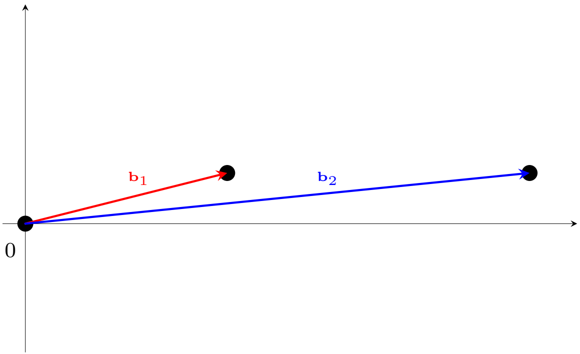
q-ary Lattices

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

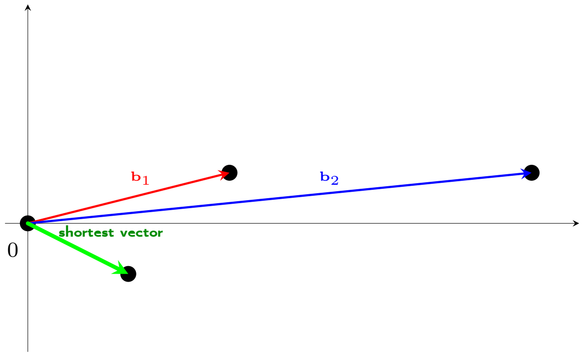
$$\Lambda_q(\mathbf{B}) = \{\mathbf{B} \cdot \mathbf{x} \mod q : \mathbf{x} \in \mathbb{Z}^n\}, \text{ and}$$

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

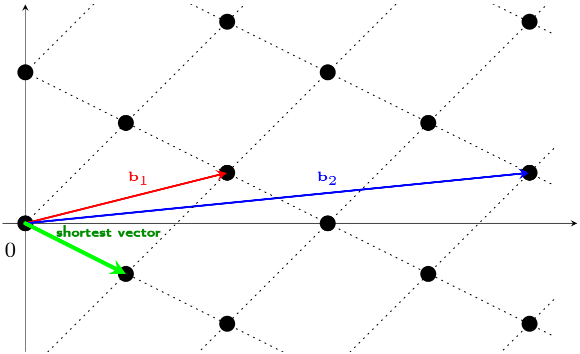
Hard problems: the Shortest Vector Problem



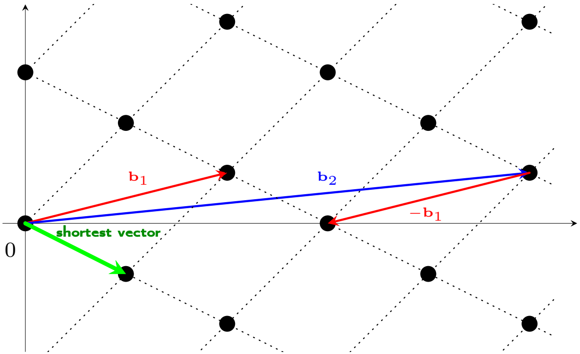
Hard problems: the Shortest Vector Problem



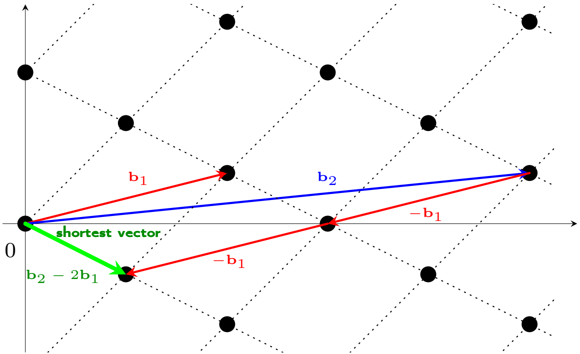
Hard problems: the Shortest Vector Problem



Hard problems: the Shortest Vector Problem

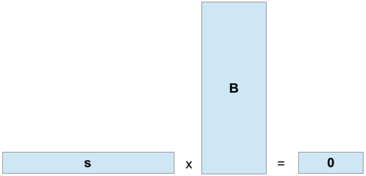


Hard problems: the Shortest Vector Problem



Hard problems: the Small Integer Solution

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

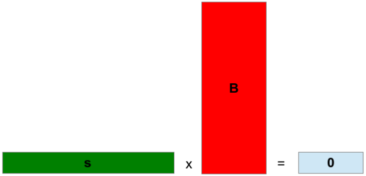


Relationship to Lattices

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

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 $\mathbf{s}^t\mathbf{B} = \mathbf{0} \mod q$

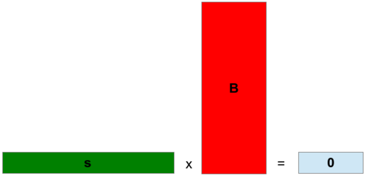


Relationship to Lattices

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

Hard problems: the Small Integer Solution

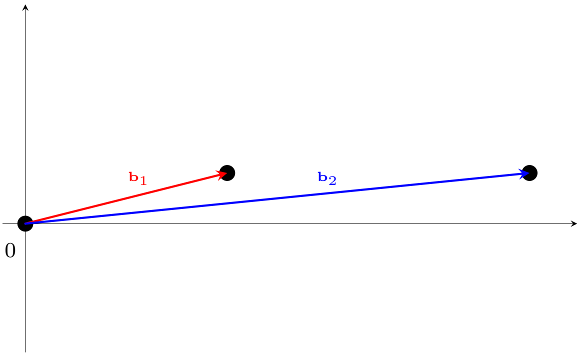
Given $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, find "small" $\mathbf{s} \in \mathbb{Z}^m$ such that $\mathbf{s}^t\mathbf{B} = \mathbf{0} \mod q$



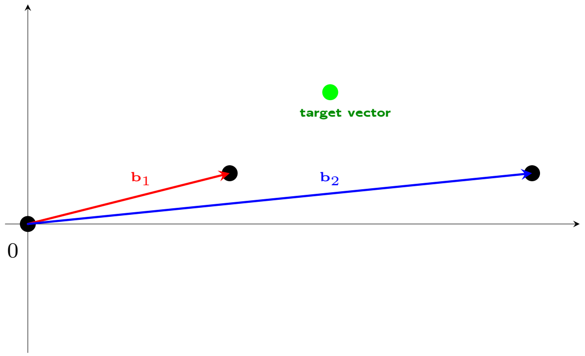
Relationship to Lattices

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

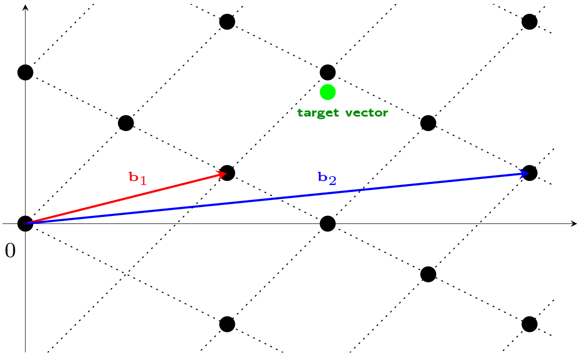
Hard problems: the Closest Vector Problem



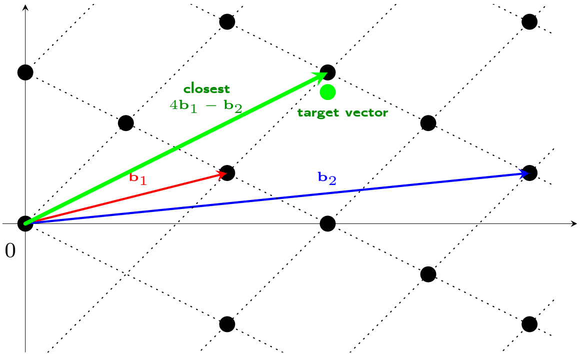
Hard problems: the Closest Vector Problem



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

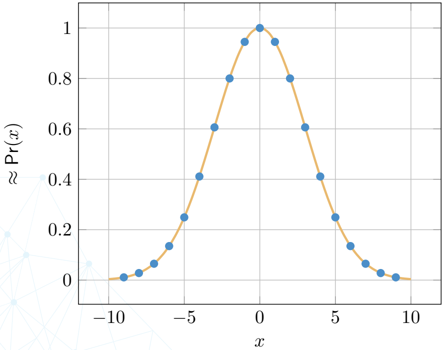
$$\begin{pmatrix} \\ \\ \mathbf{c} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} \leftarrow & n & \rightarrow \\ & & \\ & \mathbf{A} & \\ & & \\ & & \end{pmatrix} \cdot \begin{pmatrix} \mathbf{s} \end{pmatrix} + \begin{pmatrix} \\ \\ \mathbf{e} \\ \\ \\ \end{pmatrix}$$

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 $\Lambda_q(\mathbf{B})$.

Parameters



- Parameters are:
  - dimension n,
  - modulus q (e.g. $q \approx n^2$),
  - noise size α (e.g. $\alpha q \approx \sqrt{n}$),
  - number of samples m.
- Elements of $\mathbf{A}, \mathbf{s}, \mathbf{e}, \mathbf{c}$ are in $\mathbb{Z}_q$.
- $\mathbf{e}$ is sampled from χ_α, a discrete Gaussian with width

$$\sigma = \frac{\alpha q}{\sqrt{2\pi}}.$$

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.

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) $\rightarrow$ pk
- easy given a "good" basis (of short and almost orthogonal vectors) $\rightarrow$ sk

Lattice problems

Idea behind lattice-based cryptography: these problems are

- hard given a "bad" basis (constituted of long and almost parallel vectors) $\to$ pk
- easy given a "good" basis (of short and almost orthogonal vectors) $\to$ sk

All these problems do not seem hard in dimension 2...

Lattice problems

Idea behind lattice-based cryptography: these problems are

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

All these problems do not seem hard in dimension 2...

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

Best known attacks: lattice reduction

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

Best known attacks: 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!

Best known attacks: 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!

$\rightarrow$ right, but the resulting set of vectors no longer spans the same lattice.

Best known attacks: 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!

$\rightarrow$ 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:

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. $\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=0}^{i-1} \frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \mathbf{b}_j^*$

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. $\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=0}^{i-1} \frac{\langle \mathbf{b}_j^*, \mathbf{b}_i \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle} \mathbf{b}_j^*$

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

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

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

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

Richard Lindner* Chris Peikert†

November 30, 2010

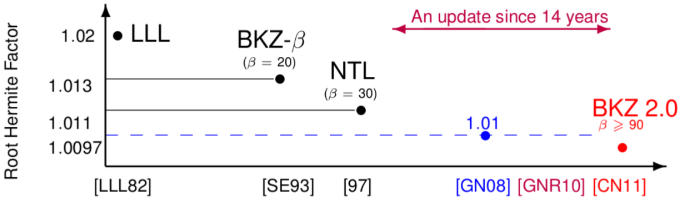
Abstract

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



NTRUSign: lattice-based signature

NTRUSign: lattice-based signature

History

- Originally NSS [HPS01]
- NTRUSign [HPSW02]

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]
- NTRUSign [HPSW02]

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:

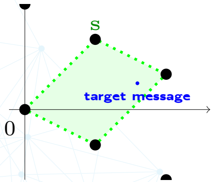
- Define $\mathbf{m} = \mathcal{H}(\mu)$
- Solve CVP with target $(\mathbf{0}, \mathbf{m})$ and good basis $\mathbf{S}$

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 $\mathbf{S}$

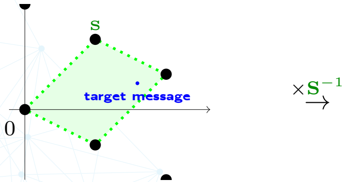


NTRUSign

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 $\mathbf{S}$

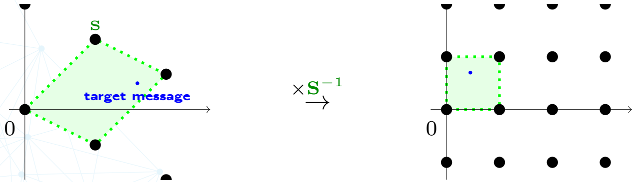


NTRUSign

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 $\mathbf{S}$

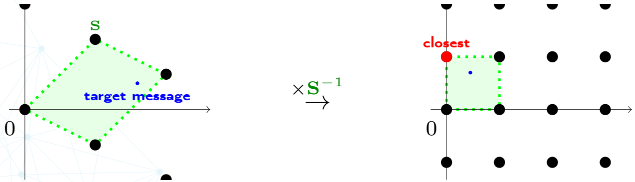


NTRUSign

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 $\mathbf{S}$

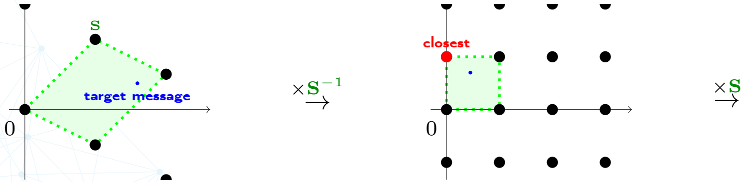


NTRUSign

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 $\mathbf{S}$

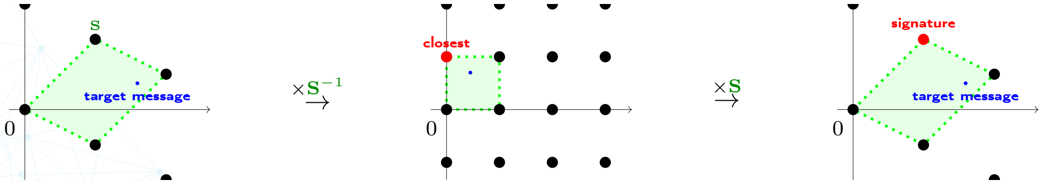


NTRUSign

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

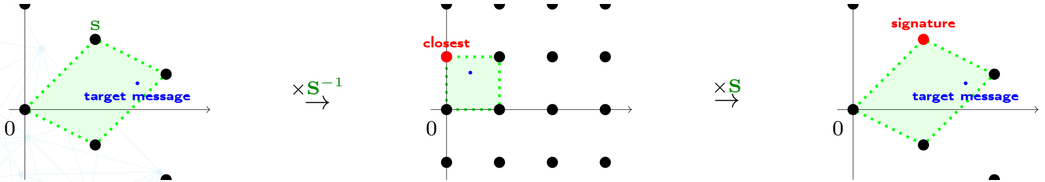
Given $\mu \in \{0,1\}^*$ to sign:

- 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

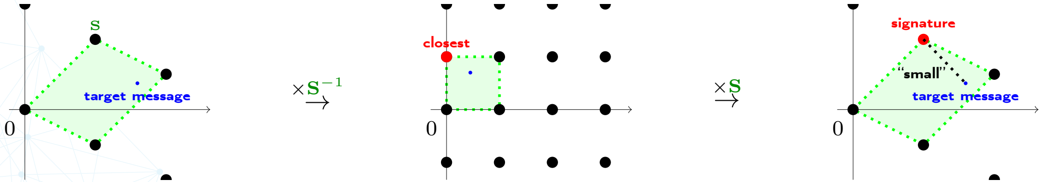
Given $\mu \in \{0,1\}^*$ to sign:

- 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

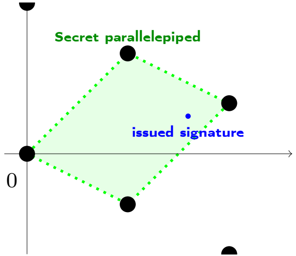
NTRUSign runs faster !

But...

Problem : Not Zero-Knowledge

Key-recovery attacks

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

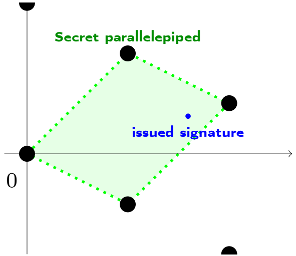


Number of signature issued : 1

Problem : Not Zero-Knowledge

Key-recovery attacks

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

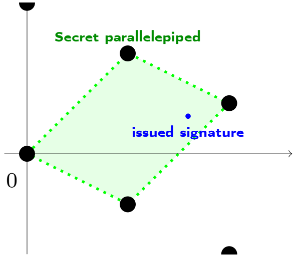


Number of signature issued : 1

Problem : Not Zero-Knowledge

Key-recovery attacks

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

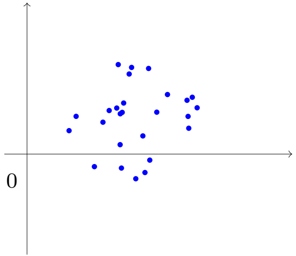


Number of signature issued : 1

Problem : Not Zero-Knowledge

Key-recovery attacks

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

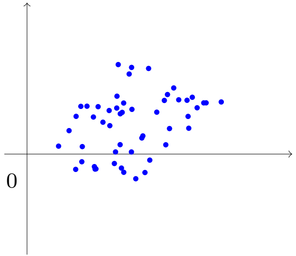


Number of signatures issued : 25

Problem : Not Zero-Knowledge

Key-recovery attacks

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

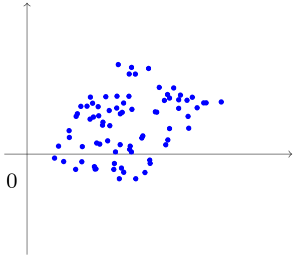


Number of signatures issued : 50

Problem : Not Zero-Knowledge

Key-recovery attacks

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

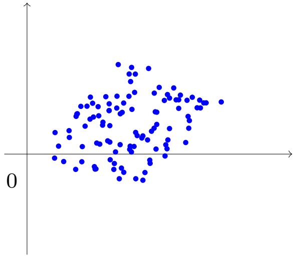


Number of signatures issued : 75

Problem : Not Zero-Knowledge

Key-recovery attacks

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

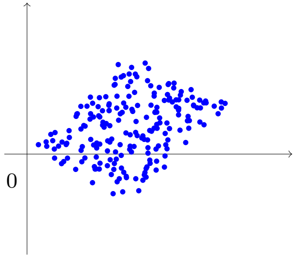


Number of signatures issued : 100

Problem : Not Zero-Knowledge

Key-recovery attacks

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

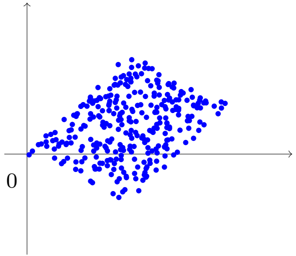


Number of signatures issued : 200

Problem : Not Zero-Knowledge

Key-recovery attacks

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

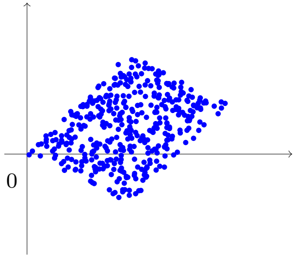


Number of signatures issued : 300

Problem : Not Zero-Knowledge

Key-recovery attacks

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

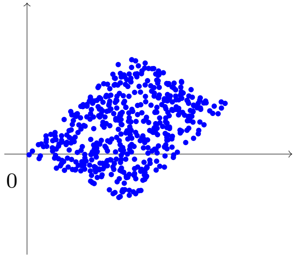


Number of signatures issued : 400

Problem : Not Zero-Knowledge

Key-recovery attacks

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

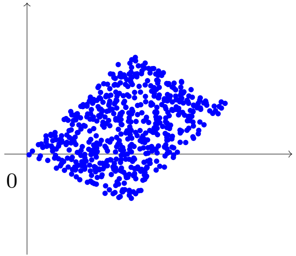


Number of signatures issued : 500

Problem : Not Zero-Knowledge

Key-recovery attacks

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

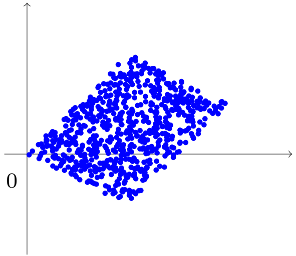


Number of signatures issued : 600

Problem : Not Zero-Knowledge

Key-recovery attacks

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

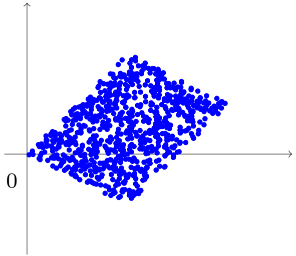


Number of signatures issued : 700

Problem : Not Zero-Knowledge

Key-recovery attacks

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

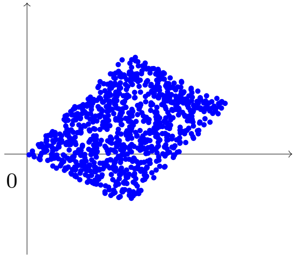


Number of signatures issued : 800

Problem : Not Zero-Knowledge

Key-recovery attacks

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

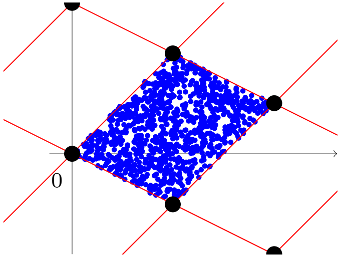


Number of signatures issued : 900

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 : $\mathbf{S} \xleftarrow{\$} \{-d, \ldots, 0, \ldots, d\}^{m \times k}$
- Public key : $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ 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]

KeyGen

- Secret key : $\mathbf{S} \xleftarrow{\$} \{-d, \ldots, 0, \ldots, d\}^{m \times k}$
- Public key : $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ 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]

KeyGen

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

Sign

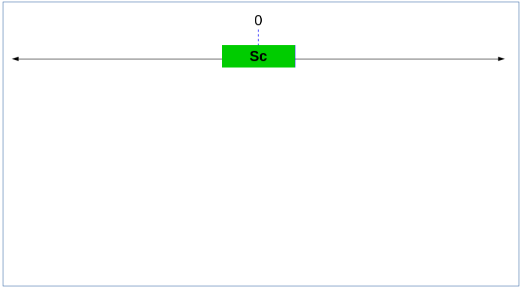
First stage [Finding pre-image]

- 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]

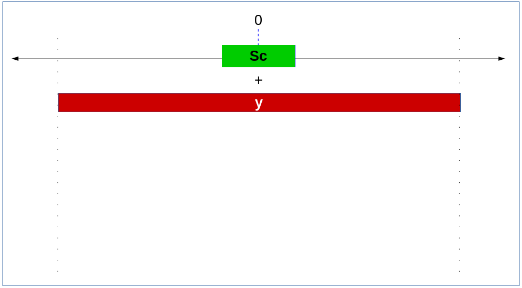


Verify

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

Secure lattice based signatures [Lyu12]

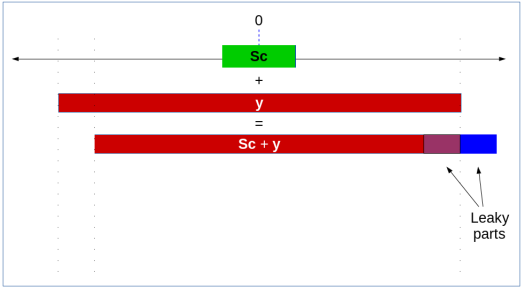


Verify

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

Secure lattice based signatures [Lyu12]

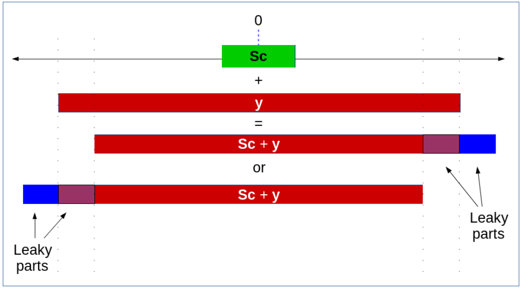


Verify

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

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

Secure lattice based signatures [Lyu12]

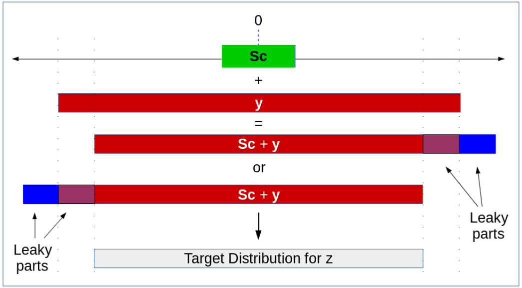


Verify

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

Secure lattice based signatures [Lyu12]

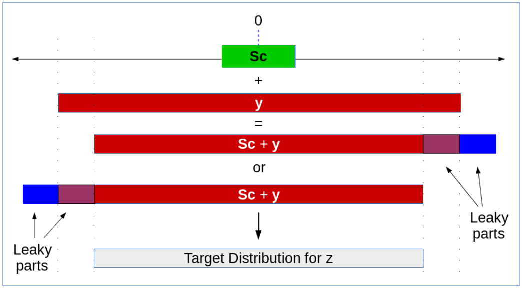


Verify

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

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
≈ sign size	163,000	142,300	73,000	14,500	19,500
≈ pk size	2^{20}	$2^{22.5}$	2^{23}	$2^{19.5}$	$2^{21.5}$
≈ 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

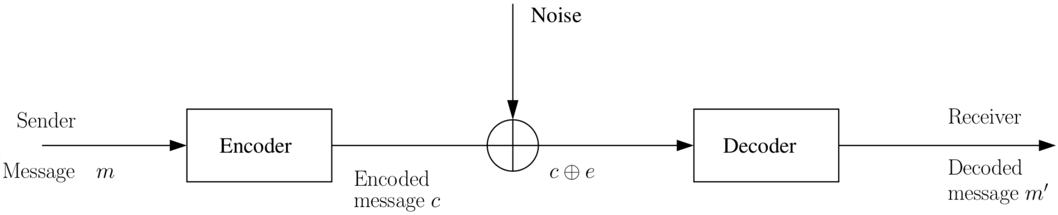
5 Quantum safe alternatives
- Lattice-based cryptography
- Hash-based cryptography
- Code-based cryptography

Outline

5 Quantum safe alternatives
- Lattice-based cryptography
- Hash-based cryptography
- Code-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.



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 $\Rightarrow$ 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 $\mathbb{F}_5^7$

$\mathbf{u} =$	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 $\mathbb{F}_5^7$

$\mathbf{u} =$	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 $\Rightarrow$ 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 $\mathbb{F}_5^7 \rightarrow d_H(\mathbf{u}, \mathbf{v}) = 4$

$\mathbf{u} =$	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

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 $\mathbf{G}$:

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

Code-based cryptography (CBC)

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

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

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

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

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 :

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

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

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 :

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

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

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 $wt(\mathbf{e}) \leq t$

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

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

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

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

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

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

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

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

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

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

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

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

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

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 $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

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

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

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,

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

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 $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

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

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

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 $\mathbf{h} = \mathbf{h}_1 \mathbf{h}_0^{-1}$.

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

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

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

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

$\xrightarrow{\mathbf{z},\mathbf{c}}$

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

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

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

Verif :

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

Verif :

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

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

Verif :

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

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

Verif :

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

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 $\mathcal{H}\left(\mathbf{Hz}^\top - \mathbf{s} \cdot \mathbf{c}\right) == \mathbf{c}$

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 $\mathcal{H}\left(\mathbf{Hz}^\top - \mathbf{s} \cdot \mathbf{c}\right) == \mathbf{c}$

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)
$\Rightarrow$ décodage classique facile et efficace

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

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

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

Syndrome connu, matrice de parité creuse connue (LDPC)
$\Rightarrow$ décodage classique facile et efficace
$\Rightarrow$ 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 $(\mathbf{b}_0, \ldots, \mathbf{b}_{m-1})$ une base de $\mathbb{F}_{q^m}$ sur $\mathbb{F}_q$.

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

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

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 $(b_1, \ldots, b_m)$ 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 $\mathbb{F}_q$

$\rightarrow$ 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 $\mathbb{F}_q$

$\rightarrow$ 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 $n = 3$:

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

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

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 $\mathbb{F}_q$

$\rightarrow$ 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 $n = 3$:

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

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

Définitions

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

$\rightarrow$ 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 $(b_1, \dots, b_m)$ 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 $n = 3$:

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

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

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 $n = 3$:

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

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

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 $\mathbb{F}_q$

$\rightarrow$ 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 $n = 3$:

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

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

$\rightarrow$ rang$(\mathbf{v}) = 2$

Définitions

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

$\rightarrow$ 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 $n = 3$:

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

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

→ $\text{rang}(\mathbf{v}) = 2$

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

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.