Quantum Computing and Cryptography

Damien, Théo, Matthieu

February 12, 2025

Outline

Outline

1. Intro
2. RSA
3. Quantum computing
   - Introduction to the quantum world
   - Quantum algorithms
4. Post-Quantum cryptography
   - Intro to PQ cryptography
   - Lattice cryptography
     - What is a lattice ?
   - Limits of PQ cryptography
5. Conclusion

Introduction

What is Cryptography?

- Science of secret $\kappa\rho\upsilon\pi\tau o\varsigma$
- Two complementary parts: cryptography and cryptanalysis

Introduction

What is Cryptography?

- Science of secret $\kappa\rho\upsilon\pi\tau o\varsigma$
- Two complementary parts: cryptography and cryptanalysis

Historically

- Cryptography was about hiding the content of a message
- Cryptanalysis want to get this message

Introduction

What is Cryptography?

- Science of secret $\kappa\rho\upsilon\pi\tau o\varsigma$
- Two complementary parts: cryptography and cryptanalysis

Historically

- Cryptography was about hiding the content of a message
- Cryptanalysis want to get this message

Nowadays

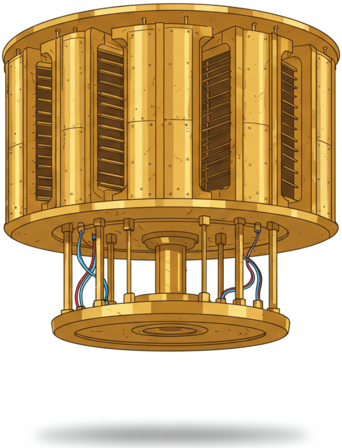
- Cryptography: creating protocols to protect a communication
- Cryptanalysis: Measuring the security level of those protocols

Outline

RSA

Outline

Quantum computing



Outline

Qubits

Classical bit

$b \in \{0, 1\}$

Qubits

Classical bit

$b \in \{0, 1\}$

- 0

Qubits

Classical bit

$b \in \{0, 1\}$

- 0
- 1

Qubits

Classical bit

$b \in \{0, 1\}$

- 0
- 1

Quantum bit

$|\psi\rangle \in \mathbb{C}^2$

Qubits

Classical bit

$b \in \{0, 1\}$

- 0
- 1

Quantum bit

$|\psi\rangle \in \mathbb{C}^2$

- $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Qubits

Classical bit

$b \in \{0, 1\}$

- 0
- 1

Quantum bit

$|\psi\rangle \in \mathbb{C}^2$

- $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$
- $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Qubits

Classical bit

$b \in \{0, 1\}$

- 0
- 1

Quantum bit

$|\psi\rangle \in \mathbb{C}^2$

- $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$
- $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
- $|+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Qubits

Classical bit

$b \in \{0, 1\}$

- 0
- 1

Quantum bit

$|\psi\rangle \in \mathbb{C}^2$

- $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$
- $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
- $|+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$
- $|-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

Measurements

Why measuring ?

We cannot read superposition. When we look at a qubit, it collapses to a classical bit.

Measurements

Why measuring ?

We cannot read superposition. When we look at a qubit, it collapses to a classical bit.

What do we get ?

We measure 0 or 1 with a probability that depends on the state of the qubit.

Why measuring ?

We cannot read superposition. When we look at a qubit, it collapses to a classical bit.

What do we get ?

We measure 0 or 1 with a probability that depends on the state of the qubit.

- $|0\rangle \rightarrow 0$ (100%)

Measurements

Why measuring ?

We cannot read superposition. When we look at a qubit, it collapses to a classical bit.

What do we get ?

We measure 0 or 1 with a probability that depends on the state of the qubit.

- $|0\rangle \rightarrow 0$ (100%)
- $|1\rangle \rightarrow 1$ (100%)

Measurements

Why measuring ?

We cannot read superposition. When we look at a qubit, it collapses to a classical bit.

What do we get ?

We measure 0 or 1 with a probability that depends on the state of the qubit.

- $|0\rangle \rightarrow 0$ (100%)
- $|1\rangle \rightarrow 1$ (100%)
- $|+\rangle \rightarrow 0$ (50%), 1 (50%)

Measurements

Why measuring ?

We cannot read superposition. When we look at a qubit, it collapses to a classical bit.

What do we get ?

We measure 0 or 1 with a probability that depends on the state of the qubit.

- $|0\rangle \rightarrow 0$ (100%)
- $|1\rangle \rightarrow 1$ (100%)
- $|+\rangle \rightarrow 0$ (50%), 1 (50%)
- $|-\rangle \rightarrow 0$ (50%), 1 (50%)

NOT gate

X gate

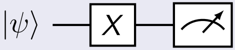
- $X\left|0\right\rangle \rightarrow \left|1\right\rangle$
- $X\left|1\right\rangle \rightarrow \left|0\right\rangle$

NOT gate

X gate

- $X\,|0\rangle \rightarrow |1\rangle$
- $X\,|1\rangle \rightarrow |0\rangle$

Circuit representation



Hadamard gate

H gate

- $H |0\rangle \rightarrow |+\rangle$
- $H |1\rangle \rightarrow |-\rangle$

Hadamard gate

H gate

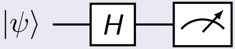
- $H\left|0\right\rangle \rightarrow \left|+\right\rangle$
- $H\left|1\right\rangle \rightarrow \left|-\right\rangle$
- $H\left|+\right\rangle \rightarrow \left|0\right\rangle$
- $H\left|-\right\rangle \rightarrow \left|1\right\rangle$

Hadamard gate

H gate

- $H|0\rangle \rightarrow |+\rangle$
- $H|1\rangle \rightarrow |-\rangle$
- $H|+\rangle \rightarrow |0\rangle$
- $H|-\rangle \rightarrow |1\rangle$

Circuit representation



Outline

Bernstein-Vazirani Problem

Problem Definition

Given an oracle for a function f:

$$f : \{0,1\}^n \to \{0,1\}$$

$$f(\mathrm{x}) = \mathrm{x} \cdot \mathrm{s}$$

where s is a secret bit string. Find s with the fewest oracle calls. ($\cdot$ is the bitwise dot product, XOR sum).

Classical Algorithm - Example

Example (n=2)

To find s $= s_0 s_1$:

Requires 2 queries.

Classical Algorithm - Example

Example (n=2)

To find s $= s_0 s_1$:

- Query $f(10) = 1 \cdot s_0 + 0 \cdot s_1 = s_0$

Requires 2 queries.

Classical Algorithm - Example

Example (n=2)

To find s $= s_0 s_1$:

- Query $f(10) = 1 \cdot s_0 + 0 \cdot s_1 = s_0$
- Query $f(01) = 0 \cdot s_0 + 1 \cdot s_1 = s_1$

Requires 2 queries.

Classical Algorithm - General Case

Classical complexity: $\mathcal{O}(n)$

We need to isolate each bit of s by querying with inputs that have a single '1'. This requires n queries for an n-bit string.

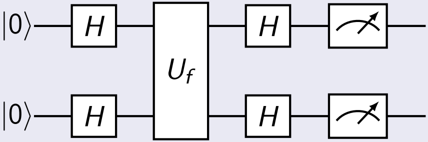
Quantum Algorithm - Overview

Quantum complexity: $\mathcal{O}(1)$

The quantum algorithm can find s with just **one** query. It uses superposition to query all possible inputs simultaneously.

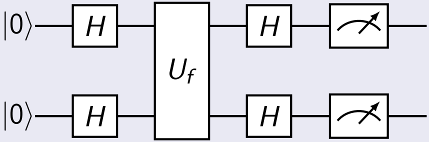
Quantum Algorithm - Circuit

Circuit Diagram



Quantum Algorithm - Circuit

Circuit Diagram



Explanation

Quantum Algorithm - Circuit

Circuit Diagram

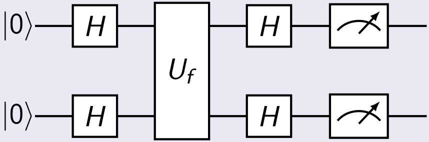
$$
\begin{array}{l}
|0\rangle - \boxed{H} - \boxed{} - \boxed{H} - \boxed{\nearrow} - \\
\qquad\qquad\qquad\;\; U_f \\
|0\rangle - \boxed{H} - \boxed{} - \boxed{H} - \boxed{\nearrow} -
\end{array}
$$

Explanation

- H : Hadamard gates on all n input qubits (creates superposition).

Quantum Algorithm - Circuit

Circuit Diagram

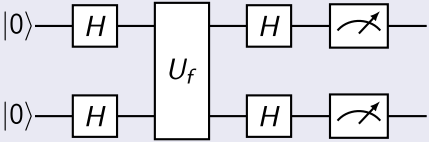


Explanation

- H : Hadamard gates on all n input qubits (creates superposition).
- U_f : The quantum oracle.

Quantum Algorithm - Circuit

Circuit Diagram



Explanation

- H : Hadamard gates on all n input qubits (creates superposition).
- U_f : The quantum oracle.
- Final Hadamards and measurement reveal s.

Shor's Algorithm

Complexity Gain

Classical factoring is very slow (roughly $\mathcal{O}(e^{\sqrt[3]{n}})$). Shor's algorithm is much faster (polynomial, $\mathcal{O}(n^3)$).

Shor's Algorithm

Complexity Gain

Classical factoring is very slow (roughly $\mathcal{O}(e^{\sqrt[3]{n}})$). Shor's algorithm is much faster (polynomial, $\mathcal{O}(n^3)$).

Requirements

Complexity Gain

Classical factoring is very slow (roughly $\mathcal{O}(e^{\sqrt[3]{n}})$). Shor's algorithm is much faster (polynomial, $\mathcal{O}(n^3)$).

Requirements

- Requires a large number of high-quality (low-error) qubits (roughly $2n$ for an n-bit number).

Shor's Algorithm

Complexity Gain

Classical factoring is very slow (roughly $\mathcal{O}(e^{\sqrt[3]{n}})$). Shor's algorithm is much faster (polynomial, $\mathcal{O}(n^3)$).

Requirements

- Requires a large number of high-quality (low-error) qubits (roughly $2n$ for an n-bit number).
- We currently don't have quantum computers large and stable enough to break practical RSA encryption.

Outline

Outline

What is PQ cryptography

What it is:

- Based on (other) mathematical problems
- Considered unsolvable by a quantum computer

What is PQ cryptography

What it is:

- Based on (other) mathematical problems
- Considered unsolvable by a quantum computer
- **And** by classical computers

What is PQ cryptography

What it is:

- Based on (other) mathematical problems
- Considered unsolvable by a quantum computer
- **And** by classical computers

What it is not:

Cryptography **using** quantum technologies

- Many cases where it is unusable
- Considered unreliable

The problems

- Codes
- Hash functions
- Multivariates polynomials systems
- Isogenies
- Lattices

The problems

- Codes
- Hash functions
- Multivariates polynomials systems
- Isogenies
- Lattices

Why lattices ?

- Well spread
- Good results

Encryption/Key encapsulation	
Crystals-Kyber	Lattices
Signatures	
Crystals-Dilithium	Lattices
Falcon	Lattices
Sphincs+	Hash

Table: Results from the NIST

Outline

Some definitions

The unformal definition

A arrangement of points in space, following a regular pattern

Some definitions

The unformal definition

A arrangement of points in space, following a regular pattern

The (more) formal one

A discret subgroup of $\mathbb{R}^n$, with the euclidean distance

Some definitions

The unformal definition

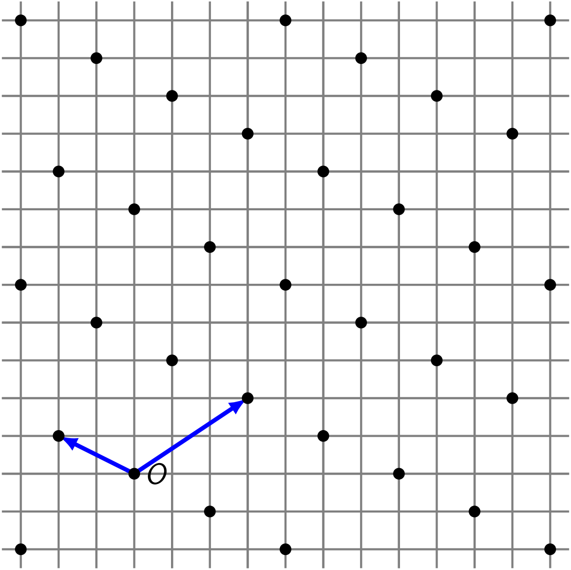
A arrangement of points in space, following a regular pattern

The (more) formal one

A discret subgroup of $\mathbb{R}^n$, with the euclidean distance

$\rightarrow$ We have vectors, dot/scalar product and matrices

Example



Learning with error problem

TODO

(Fully) homomorphic encryption

- Based on lattices (variant of LWE problem)

(Fully) homomorphic encryption

- Based on lattices (variant of LWE problem)
- We can evaluate a circuit (operations) on encrypted data
- Two operations : $+$ and $\cdot$, forms a ring

(Fully) homomorphic encryption

- Based on lattices (variant of LWE problem)
- We can evaluate a circuit (operations) on encrypted data
- Two operations : $+$ and $\cdot$, forms a ring
- We can evaluate (or compile) a function $f : P \times P \to P$ on encrypted data C_1 and C_2 :

$$eval(f, C_1, C_2) = Enc(f(Dec(C_1), Dec(C_2)))$$

(Fully) homomorphic encryption

- Based on lattices (variant of LWE problem)
- We can evaluate a circuit (operations) on encrypted data
- Two operations : $+$ and $\cdot$, forms a ring
- We can evaluate (or compile) a function $f : P \times P \to P$ on encrypted data C_1 and C_2 :

$$eval(f, C_1, C_2) = Enc(f(Dec(C_1), Dec(C_2)))$$

- Used to manipulate private data (e.g. Medical data, data science)

Outline

Sizes of the keys and data

TODO	

Not necessarly robust to classical computer

- Example : Supersingular isogenies Diffie-Hellman key exchange

Outline

Conclusion