Post-Quantum

Cryptography Conference

Lattice-based Cryptography

Léo Ducas

Researcher at Centrum Wiskunde & Informatica (CWI) and Professor at Leiden University

Lattice-based Cryptography

Léo Ducas

Centrum Wiskunde & Informatica, Amsterdam
Leiden University, Mathematical Institute

PKI-Consortium, Post-Quantum Cryptography Conference,
Nov 2023

Standardisation of Lattice-based Cryptosystems

Lattice in Post-Quantum Cryptography Standardisation Process:

NIST
National Institute of Standards and Technology

Key Exchange:

- Kyber

Signature:

- Dilithium
- Falcon

ISO

Key Exchange:

- Kyber
- Frodo

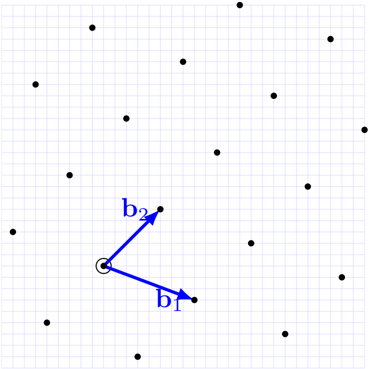
Signature:

- later ??

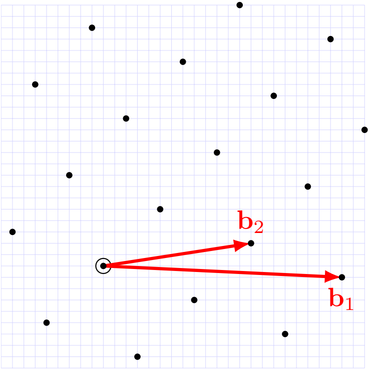
- Lattice
- Public Key Encryption with Lattices
- Digital Signatures with Lattices
- Current State of Cryptanalysis

Lattices and their Bases

Lattices are (infinite) regular grids of point in (euclidean) space.
They can be finitely described thanks to their bases.
Example in Dimension 2:



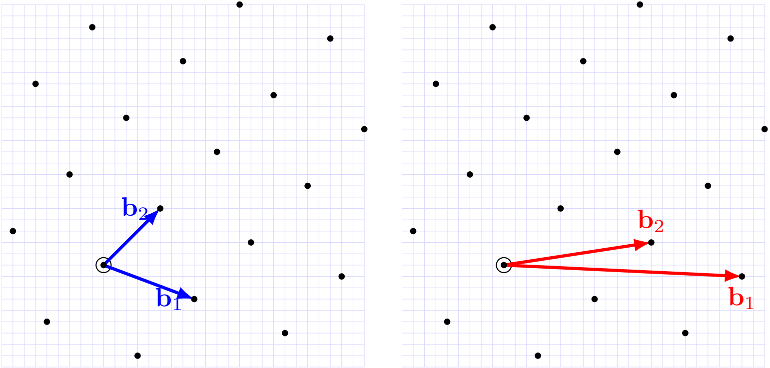
Good Basis $\mathbf{G}$ of L



Bad Basis $\mathbf{B}$ of L

Lattices and their Bases

Lattices are (infinite) regular grids of point in (euclidean) space.
They can be finitely described thanks to their bases.
Example in Dimension 2:

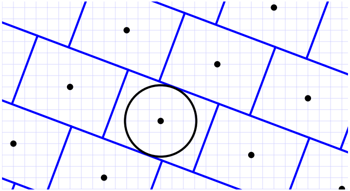


Good Basis $\mathbf{G}$ of L Bad Basis $\mathbf{B}$ of L

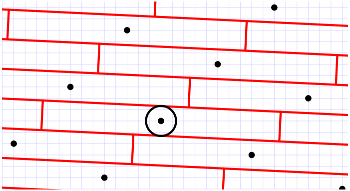
G $\rightarrow$ **B** : easy (randomization);
B $\rightarrow$ **G** : hard (LLL, BKZ, Lattice Sieve...).

Using Lattices in Cryptography

Bases allow to 'tile' the space, and perform error correction.



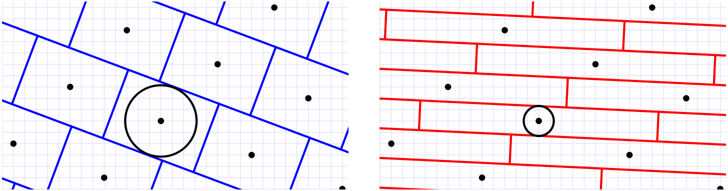
Decoding radius with $\mathbf{G}^\star$



Decoding radius with $\mathbf{B}^\star$

Using Lattices in Cryptography

Bases allow to 'tile' the space, and perform error correction.



Decoding radius with $\mathbf{G}^\star$ Decoding radius with $\mathbf{B}^\star$

As dimension grows > 2, the error tolerance gap between **G** and **B** grows exponentially.

Lattice-Based Asymetric Cryptography

- secret key = good basis **G**
- public key = bad basis **B**

Public Key Encryption with Lattices

Public Key Encryption with Lattices

Encryption Procedure

- View the message as a lattice point $m \in L$ (can do with **B**)
- Choose a random small error vector e (e.g. binary)
- Return ciphertext $c = m + e$

Public Key Encryption with Lattices

Encryption Procedure

- View the message as a lattice point $m \in L$ (can do with **B**)
- Choose a random small error vector e (e.g. binary)
- Return ciphertext $c = m + e$

Decryption Procedure

- Tile to recover the center m of the tile (should do with **G**)
- Return decrypted message m

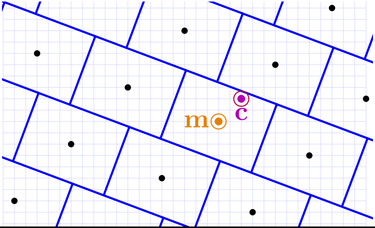
Public Key Encryption with Lattices

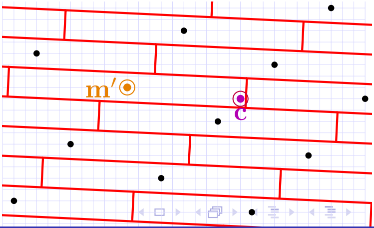
Encryption Procedure

- View the message as a lattice point $m \in L$ (can do with **B**)
- Choose a random small error vector e (e.g. binary)
- Return ciphertext $c = m + e$

Decryption Procedure

- Tile to recover the center m of the tile (should do with **G**)
- Return decrypted message m





Lattice-based Encryption is as simple as Tetris

It might be hard to get intuition for lattice in dimension > 2...

Cryptris:

A serious game to understand how it works, and why it is secure.

Developed with **Inria** (FR), translated to EN and NL at **CWI**

`https://cryptris.nl/`

Simple to Implement

- Encryption involve a Matrix-Vector product
- Tiling is a more involved, but Decryption can be simplified
- We can choose q-ary lattices, to make all computation mod q

Structured Lattices

- Use circulant blocks in the matrix to improve compactness

$$\begin{bmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{n-2} & & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix}$$

- Speed benefits as well thanks to Fast Fourier Transform

Fast, but a bit Large

- Computation speed is **not** an issue
  worst operation is a **fraction of milli-second** on `x86-Haswell`
- Key and ciphertext sizes are larger than pre-quantum
  but nothing is particularly huge

Kyber-512

Sizes (bytes)	Cycles (ref)	Cycles (avx2)
sk: 2400	gen: 199k	gen: 52k
pk: 1184	enc: 235k	enc: 68k
ct: 1088	dec: 274k	dec: 53k

Fast, but a bit Large

- Computation speed is **not** an issue
  worst operation is a **fraction of milli-second** on `x86-Haswell`
- Key and ciphertext sizes are larger than pre-quantum
  but nothing is particularly huge

Kyber-512

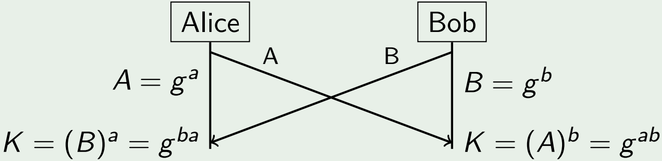
Sizes (bytes)	Cycles (ref)	Cycles (avx2)
sk: 2400	gen: 199k	gen: 52k
pk: 1184	enc: 235k	enc: 68k
ct: 1088	dec: 274k	dec: 53k

It's so fast that PRNG is the bottleneck $\approx 80\%$

- SHAKE (SHA-3 Hash with Extended Output)
- Hardware acceleration expected in future CPUs

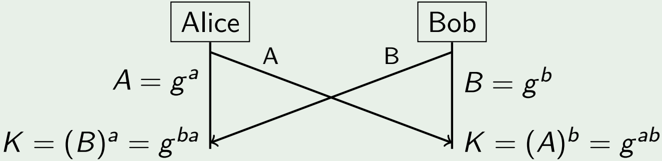
A Migration Challenge: Interactivity in Key-Exchange

DH & ECDH are non-interactive It doesn't matter who speaks first

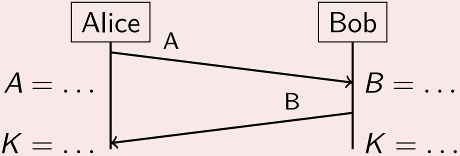


A Migration Challenge: Interactivity in Key-Exchange

DH & ECDH are **non-interactive** It doesn't matter who speaks first



Kyber is **interactive**



- the migration may require more than drop-in replacement
- the rest is only a matter of performances

Digital Signatures with Lattices

And why they are a bit more painful

A Naive Approach

RSA "Hash-then-Sign" Signatures

- Signature : Set sig := RSA-decrypt(Hash(message))
- Encryption : Check RSA-encrypt(sig) = Hash(message)

Could we just do the same with lattices ?

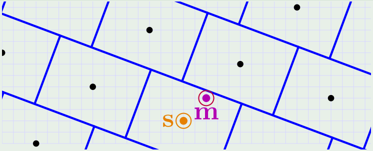
A Naive Approach

RSA "Hash-then-Sign" Signatures

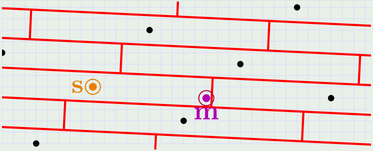
- Signature : Set sig := RSA-decrypt(Hash(message))
- Encryption : Check RSA-encrypt(sig) = Hash(message)

Could we just do the same with lattices ?

Yes !



Correct signature (close)



Incorrect signature (far)

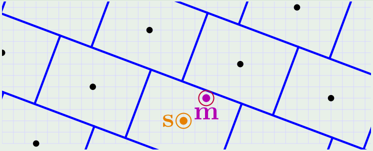
A Naive Approach

RSA "Hash-then-Sign" Signatures

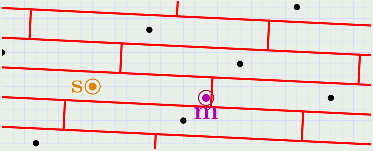
- Signature : Set sig := RSA-decrypt(Hash(message))
- Encryption : Check RSA-encrypt(sig) = Hash(message)

Could we just do the same with lattices ?

Yes !



Correct signature (close)



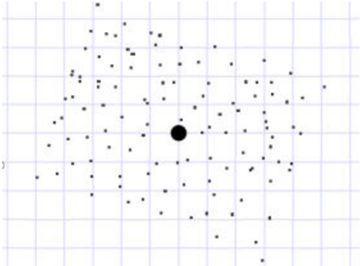
Incorrect signature (far)

but ...

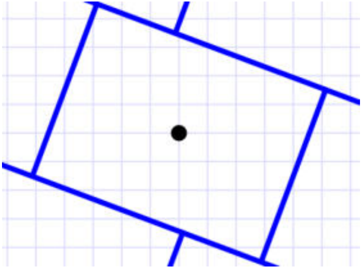
It's only secure if you don't use it much...

A Learning attack

The distribution of signatures leaks the secret key !



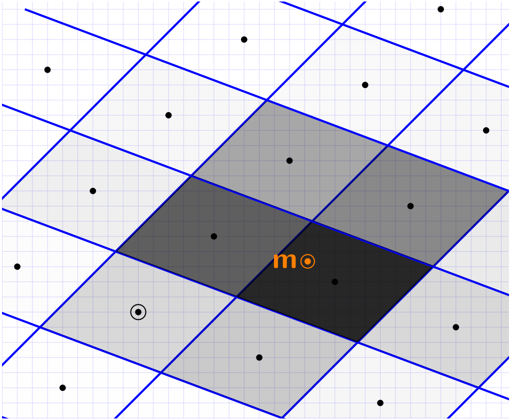
$\Rightarrow$



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

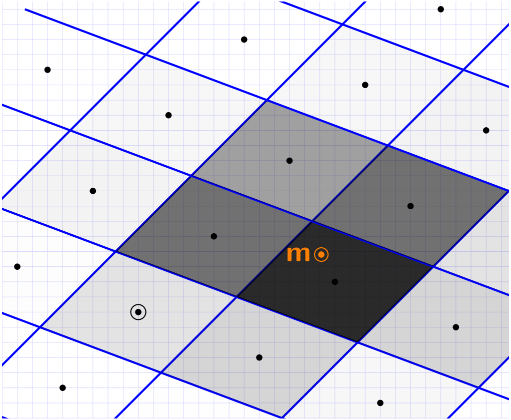
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

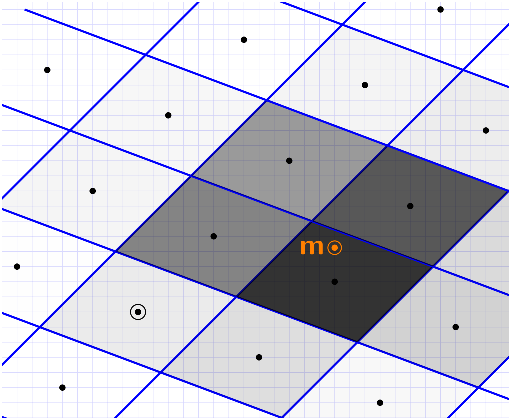
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

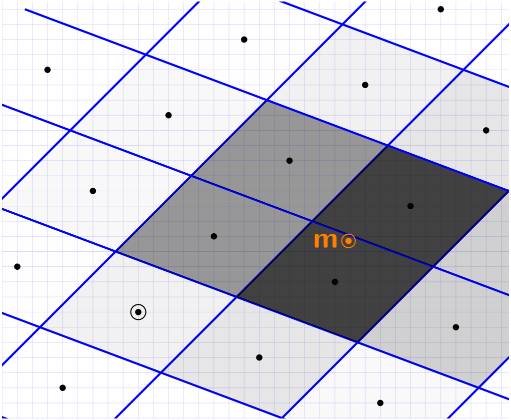
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

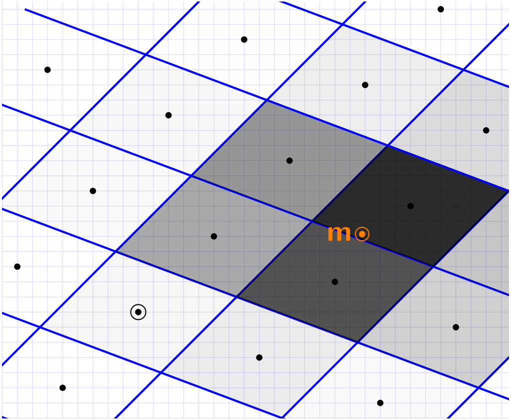
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

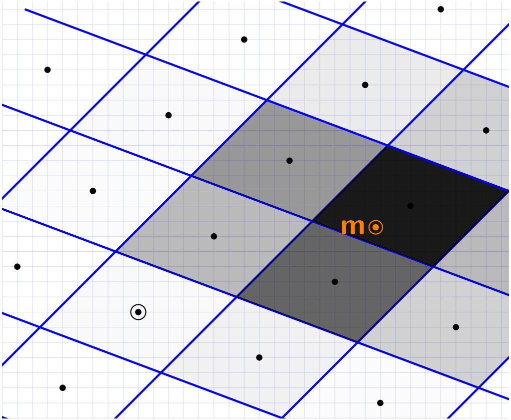
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

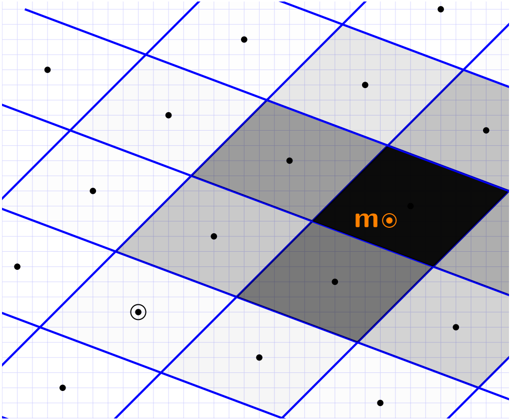
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

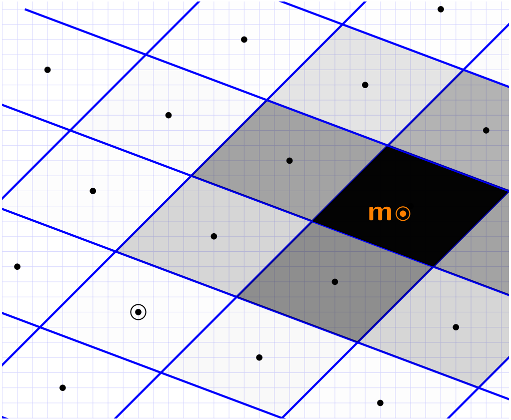
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

Idea: Hide the tile by randomized rounding



Implementation Details

- Linear algebra mod q (as for Encryption)
- Linear algebra over the real numbers
- Sampling from very specific distribution

Requires Floating Point Arithmetic

Something never done in crypto before !

- Numerical precision issues
- Determinism issues
- Timing side-channel issues

FALCON signature scheme

Despite the above technical difficulties, "Hash-then-Sign" Lattice Signatures are essentially as performant as encryption **on standard CPUs**.

Falcon-512

Sizes (bytes)	Cycles (ref)
	gen: 8.6ms
pk: 897	sign: 160μs
sig: 666	verif: 35μs

Alternative Lattice Signatures

Dilithium

- Based on a different paradigm (Fiat-Shamir with Aborts)
- No Floating Points, but an annoying "restart"
- Bigger Signatures and Public Keys

Hawk: new on-ramp candidate at NIST standardization

- Same "hash-then-sign" paradigm
- but extra orthogonal lattice structure
- No Floating Points
- Very new assumption, lacks cryptanalytic maturity
- Might be too similar to Falcon for the NIST to standardize it

Current State of Cryptanalysis

Reading Through the Public Noise

A Converging State of The Art

The cost of lattice attack is driven by the cost of solving SVP.

- The best asymptotic algorithm stabilized in 2015 at:

$$T = 2^{.292n+o(n)} \quad M = 2^{.207n+o(n)}$$

- Further improvements in practice (e.g. dimensions for free)
- Current practical records solves SVP in dimension ≈ 180, using tensor cores GPUs. Needs to reach 400.
- Precise modeling is painstaking, many things are often simplified

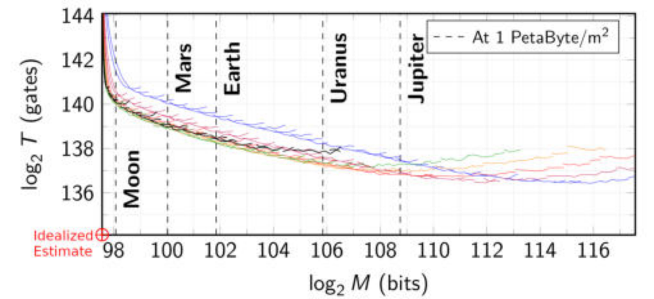
Cryptanalysis Always Gets Better ?

Publication Bias: Ignored overheads keep being ignored, even after they are pointed at and quantified.

Cryptanalysis Always Gets Better ?

Publication Bias: Ignored overheads keep being ignored, even after they are pointed at and quantified.

Estimating the Hidden Overheads in the BDGL Lattice Sieving Algorithm [D. , PQ Crypto 2022]



Is Kyber-512 Bleeding Edge ?

There are claims that Kyber-512 is weaker than AES-128 by a few bits.

These claims ignore

- Documented Algorithmic Overheads [D., Q Crypto 2022]
- The feasibility of gathering so much memory
- The logisitic of routing RAM
- The speed-of-light bound for RAM access

Is Kyber-512 Bleeding Edge ?

There are claims that Kyber-512 is weaker than AES-128 by a few bits.

These claims ignore

- Documented Algorithmic Overheads [D., Q Crypto 2022]
- The feasibility of gathering so much memory
- The logisitic of routing RAM
- The speed-of-light bound for RAM access

The Real Question

- Yes, there are unknown and room for small improvements.
- But fine tuning saving a few more bits is not the concern.
- Significant new cryptanalytic ideas is!

Questions ?

Post-Quantum Cryptography Conference

PKI Consortium