

POST-QUANTUM CRYPTOGRAPHY

LECTURE 1: INTRODUCTION TO POST-QUANTUM CRYPTOGRAPHY

Florida Atlantic University

Fall 2025

1 A Brief History of Encryption

- Merkle's Puzzles
- Diffie-Hellman
- RSA
- Elliptic Curves
- Symmetric Key Schemes
- Quantum Computers

2 Lattices and Modules

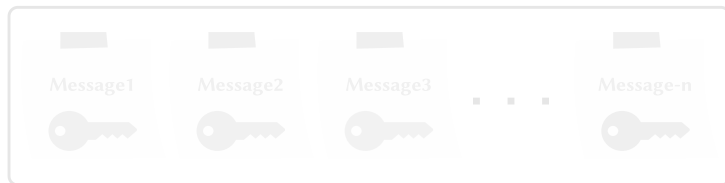
- Lattices
- Modules
- Module-Lattices
- Lattices' Hard Problems
- Arithmetic Operations in the Ring of Polynomials

How to exchange keys?

A BRIEF HISTORY OF ENCRYPTION

MERKLE'S PUZZLES

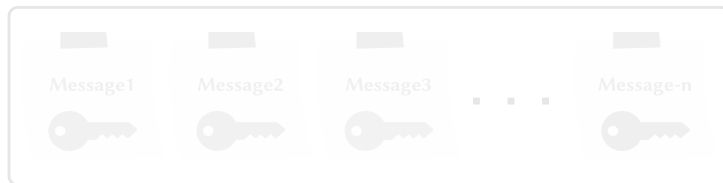
- ✧ First attempt to exchange the secret keys securely (1974).
- ✧ Lots of weakly encrypted of "This is message X . This is the symmetrical key Y ."
- ✧ Alice brute-forces one random pair and finds its message and key.
- ✧ Alice tells Bob the brute-forced message **publicly**. Bob realizes the key.
- ✧ Bob and Alice will have the same exact key to encrypt data with a symmetric key scheme.
- ✧ **Hard problem:** A large number of pair $\rightarrow$ a one-way game!



Merkle's (message, key) pairs

MERKLE'S PUZZLES

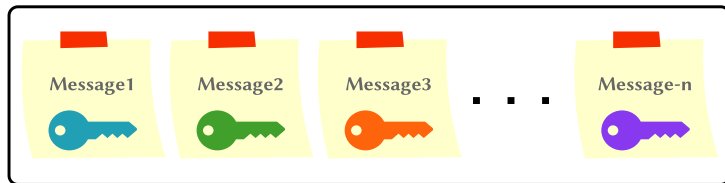
- ✧ First attempt to exchange the secret keys securely (1974).
- ✧ Lots of weakly encrypted of "This is message X . This is the symmetrical key Y ."
- ✧ Alice brute-forces one random pair and finds its message and key.
- ✧ Alice tells Bob the brute-forced message **publicly**. Bob realizes the key.
- ✧ Bob and Alice will have the same exact key to encrypt data with a symmetric key scheme.
- ✧ **Hard problem:** A large number of pair $\rightarrow$ a one-way game!



Merkle's (message, key) pairs

MERKLE'S PUZZLES

- ✧ First attempt to exchange the secret keys securely (1974).
- ✧ Lots of weakly encrypted of "This is message X . This is the symmetrical key Y ."
- ✧ Alice brute-forces one random pair and finds its message and key.
- ✧ Alice tells Bob the brute-forced message **publicly**. Bob realizes the key.
- ✧ Bob and Alice will have the same exact key to encrypt data with a symmetric key scheme.
- ✧ **Hard problem:** A large number of pair $\rightarrow$ a one-way game!



Merkle's (message, key) pairs

DIFFIE-HELLMAN

✧ Computations in $\mathbb{Z}_p^*$, a cyclic multiplicative group of integers modulo a prime number p .

Alice

Alice's secret key: a ,
Alice's public key: $g^a \pmod{p}$.



Bob

Bob's secret key: b ,
Bob's public key: $g^b \pmod{p}$.

✧ Both can compute the shared key $(g^b)^a = (g^a)^b = g^{ab} \pmod{p}$.

✧ Only Alice and Bob can find the shared key.

Elements in Diffie-Hellman

▷ $\mathbb{Z}_p^* = \{1, 2, \dots, p-1\}$.

▷ $g, a, b, g^a, g^b, g^{ab} \in \mathbb{Z}_p^*$.

DIFFIE-HELLMAN

✧ Computations in $\mathbb{Z}_p^*$, a cyclic multiplicative group of integers modulo a prime number p .

Alice

Alice's secret key: a ,
Alice's public key: $g^a \pmod{p}$.



Bob

Bob's secret key: b ,
Bob's public key: $g^b \pmod{p}$.

✧ Both can compute the shared key $(g^b)^a = (g^a)^b = g^{ab} \pmod{p}$.

✧ Only Alice and Bob can find the shared key.

Elements in Diffie-Hellman

- ▷ $\mathbb{Z}_p^* = \{1, 2, \dots, p-1\}$.
- ▷ $g, a, b, g^a, g^b, g^{ab} \in \mathbb{Z}_p^*$.

DIFFIE-HELLMAN PROBLEM

Diffie-Hellman Problem

Find $g^{ab} \pmod{p}$, given g , $g^a \pmod{p}$ and $g^b \pmod{p}$.

- ✧ Most efficient way: Discrete Logarithm.
- ✧ Anyone can compute a from g^a : $a = \log_g g^a$ (g and g^a are publicly known).
- ✧ But no known algorithm exists for computing a from $g^a \pmod{p}$ (Discrete Log Problem (DLP)), **on classical computers**.

- ✧ Uses $\mathbb{Z}_N^*$, a multiplicative group of integers modulo a composite number $N = p \times q$.
- ✧ Different approach:

Alice computes:

- ▷ $N = p \times q$.
- ▷ $\phi(N) = (p - 1)(q - 1)$.
- ▷ e : a random number coprime with $\phi(N)$.
- ▷ $d = e^{-1} \pmod{\phi(N)}$.

(e, N)
→

Everyone can encrypt:

$$c = m^e \pmod{N}.$$

- ✧ Alice tells everyone her public key. Everyone can encrypt anything (*including the shared key*) with Alice's public key, and Alice is the only person who can decrypt it:

Alice decrypts:

$$m = c^d \pmod{N} = m^{e \cdot d^{-1}} \pmod{N} = m.$$

RSA PROBLEM

RSA Problem

Find m , given N , e and $m^e \pmod{N}$.

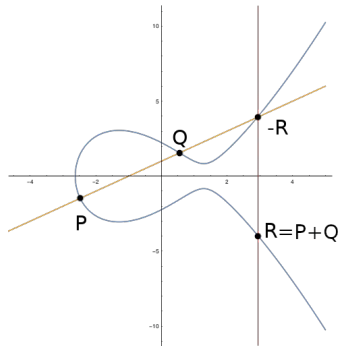
- ✧ Anyone can compute m from m^e : $m = \sqrt[e]{m^e}$.
- ✧ But no known algorithm exists for finding m from $m^e \pmod{N}$ $\longrightarrow$ Best known way: to find e^{-1} .
- ✧ To find e^{-1} , we need $\phi(n)$, or p and q $\longrightarrow$ Finding p and q : factoring N .
- ✧ There is no known algorithm to factor a super large number (Integer Factoring Problem (IFP)), **on classical computers**.

ELLIPTIC CURVES

- ✧ Using finite fields of a prime number, but every number is a point on an elliptic curve:

$$y^2 = x^3 + ax + b$$

- ✧ Computations in $E(\mathbb{F}_p)$, an abelian group of points on the elliptic curve over a finite field of a prime number p .



Point Addition on an Elliptic Curve

ELLIPTIC CURVES

✧ Similar to Diffie-Hellman, but point multiplication is used instead of number exponentiation.

Alice

Alice's secret key: a ,
Alice's public key: $[a]G \in E(\mathbb{F}_p)$.



Bob

Bob's secret key: b ,
Bob's public key: $[b]G \in E(\mathbb{F}_p)$.

✧ Alice and Bob can compute $[a] \cdot [b]G = [b] \cdot [a]G = [ab]G \in E(\mathbb{F}_p)$.

ECDH Elements

▷ Example: $E(\mathbb{F}_{17}) = \{O, (0, 6), (0, 11), (5, 1), (5, 16), \dots\}$.

▷ $a, b, G, [a]G, [b]G, [ab]G \in E(\mathbb{F}_p)$.

0

ELLIPTIC CURVES PROBLEM

Elliptic Curves Problem

Find a , given $[a]G$, G and $E(\mathbb{F}_p)$.

- ✧ No known algorithm exists for computing a from $[a]G$ (Elliptic Curve Discrete Logarithm Problem (ECDLP)), **on classical computers**.

Now, both parties have the same key to encrypt data with a symmetric key scheme.

SYMMETRIC KEY SCHEMES

- ✧ Symmetric key schemes use the same key for encryption and decryption.
- ✧ **AES**, DES, 3DES, ChaCha20, Salsa20, RC5, etc.
- ✧ Three security levels for AES:
 - ▷ **Low**: 128-bit key (AES-128).
 - ▷ **Medium**: 192-bit key (AES-192).
 - ▷ **High**: 256-bit key (AES-256).

Security Strength

The minimum security strength is 112 bits.

SYMMETRIC KEY SCHEMES

- ✧ Symmetric key schemes use the same key for encryption and decryption.
- ✧ **AES**, DES, 3DES, ChaCha20, Salsa20, RC5, etc.
- ✧ Three security levels for AES:
 - ▷ **Low**: 128-bit key (AES-128).
 - ▷ **Medium**: 192-bit key (AES-192).
 - ▷ **High**: 256-bit key (AES-256).

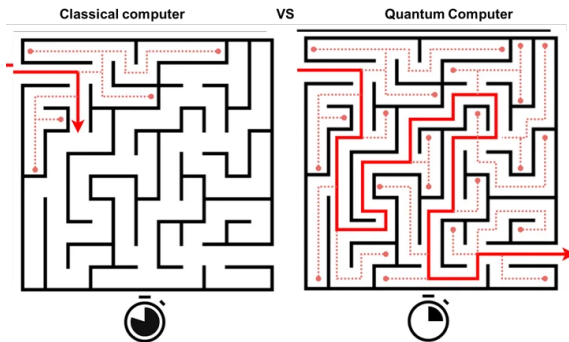
Security Strength

The minimum security strength is 112 bits.

**But all of them will be broken
by quantum computers!**

QUANTUM COMPUTERS

- ✧ Quantum computers use quantum bits (qubits) instead of classical bits.
- ✧ Qubits can be: 0 , 1 , or a *superposition* of 0 and 1 .
- ✧ Quantum computers can perform operations on multiple qubits at once.
- ✧ [Watch this video to learn more about quantum computers.](#)



Algorithms that can be executed on quantum computers

❖ **Shor's Algorithm:** Solves the IFP, DLP, and ECDLP: $\mathcal{O}((\log X)^3)$, $X \in \{N, p\}$.

▷ RSA, DH, ECDH, ECDSA, etc.

❖ **Grover's Algorithm:** Quadratically speeds up brute-force search $\rightarrow$ Convert 2^n to $2^{n/2}$.

▷ AES, SHA-2, etc.

❖ Grover's Algorithm can break AES-128 in 2^{64} operations $\rightarrow$ Feasible.

▷ **Solution:** Only use AES-256.

❖ But Shor's Algorithm can break RSA, DH, and ECC in polynomial time.

▷ **Not a solution:** Increasing the number of bits is not the solution!

▷ **Solution:** Use a new structure and a new hard problem $\rightarrow$ **Post-Quantum Cryptography.**

LATTICES AND MODULES

FAMILIAR STRUCTURES

Groups

$\mathcal{G} = (\mathbb{Z}, +)$ is a group under addition.

Rings

$\mathcal{R} = (\mathbb{Z}, +, \times)$ is a ring under addition and multiplication.

Fields

$\mathcal{F} = (\mathbb{Q}, +, \times)$ is a field under addition and multiplication.

OTHER STRUCTURES

Vector Spaces

$\mathcal{V} = \mathbb{R}^k$ is a vector space over the field $\mathbb{R}$.

Example: $\mathbb{R}^3 = \{(x, y, z) \mid x, y, z \in \mathbb{R}\}$.

Lattices

$\mathcal{L} = \mathbb{Z}^k$ is a lattice, a discrete structure.

Example: $\mathbb{Z}^3 = \{(x, y, z) \mid x, y, z \in \mathbb{Z}\}$.

Modules

$\mathcal{M} = \mathbb{Z}_n^k$ is a module, $\mathbb{Z}_n$ is a ring.

Example: $\mathcal{M} = \{(x, y, z) \mid x, y, z \in \mathbb{Z}_8\}$.

OTHER STRUCTURES

Vector Spaces

$\mathcal{V} = \mathbb{R}^k$ is a vector space over the field $\mathbb{R}$.

Example: $\mathbb{R}^3 = \{(x, y, z) \mid x, y, z \in \mathbb{R}\}$.

Lattices

$\mathcal{L} = \mathbb{Z}^k$ is a lattice, a discrete structure.

Example: $\mathbb{Z}^3 = \{(x, y, z) \mid x, y, z \in \mathbb{Z}\}$.

Modules

$\mathcal{M} = \mathbb{Z}_n^k$ is a module, $\mathbb{Z}_n$ is a ring.

Example: $\mathcal{M} = \{(x, y, z) \mid x, y, z \in \mathbb{Z}_8\}$.

OTHER STRUCTURES

Vector Spaces

$\mathcal{V} = \mathbb{R}^k$ is a vector space over the field $\mathbb{R}$.

Example: $\mathbb{R}^3 = \{(x, y, z) \mid x, y, z \in \mathbb{R}\}$.

Lattices

$\mathcal{L} = \mathbb{Z}^k$ is a lattice, a discrete structure.

Example: $\mathbb{Z}^3 = \{(x, y, z) \mid x, y, z \in \mathbb{Z}\}$.

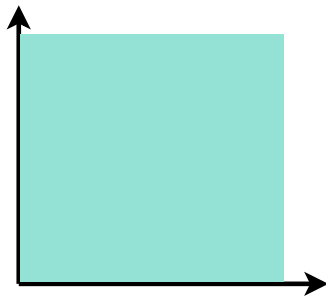
Modules

$\mathcal{M} = \mathbb{Z}_n^k$ is a module, $\mathbb{Z}_n$ is a ring.

Example: $\mathcal{M} = \{(x, y, z) \mid x, y, z \in \mathbb{Z}_8\}$.

VECTOR SPACES

- ✧ Vector spaces are continuous structures, defined over a field $\rightarrow$ We have division.
- ✧ Closure for addition: $\forall \mathbf{v}, \mathbf{w} \in \mathcal{V}, \mathbf{v} + \mathbf{w} \in \mathcal{V}$.
- ✧ Closure for scalar multiplication: $\forall \mathbf{v} \in \mathcal{V}, \forall \alpha \in \mathbb{R}, \alpha \mathbf{v} \in \mathcal{V}$.
- ✧ α is selected from the field $\mathbb{R}$.



$\mathbb{R}^2$ as a Vector Space

LATTICES

- ✧ Every lattice has n bases (linearly independent vectors, selected from a **vector space**).
- ✧ The lattice is the set of all **integer** linear combinations of the bases, which is a **discrete** structure.

$$\mathcal{L} = \left\{ \sum_{i=1}^n z_i \cdot \mathbf{b}_i \mid z_i \in \mathbb{Z}, \mathbf{b}_i \in \mathbb{R}^k \right\}.$$

If $\mathbf{b}_1 = (0.4, 0.8)$, $\mathbf{b}_2 = (1.2, 0.8)$ are the bases of a lattice, then the lattice is the set of all integer linear combinations of the bases:

$$\mathcal{L}_1 = \{ z_1 \cdot (0.4, 0.8) + z_2 \cdot (1.2, 0.8) \mid z_1, z_2 \in \mathbb{Z} \}$$

For example, $(1.6, 1.2) \in \mathcal{L}_1$,
because $4 \cdot (0.4, 0.8) - 2 \cdot (1.2, 0.8) = (0.4, 1.6)$.



LATTICES

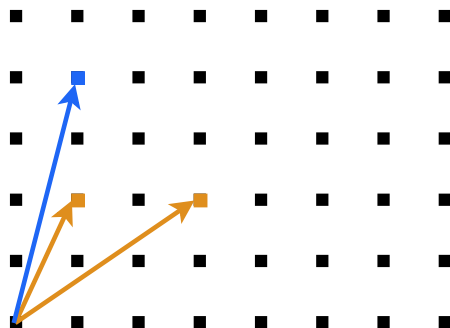
- ✧ Every lattice has n bases (linearly independent vectors, selected from a **vector space**).
- ✧ The lattice is the set of all **integer** linear combinations of the bases, which is a **discrete** structure.

$$\mathcal{L} = \left\{ \sum_{i=1}^n z_i \cdot \mathbf{b}_i \mid z_i \in \mathbb{Z}, \mathbf{b}_i \in \mathbb{R}^k \right\}.$$

If $\mathbf{b}_1 = (0.4, 0.8)$, $\mathbf{b}_2 = (1.2, 0.8)$ are the bases of a lattice, then the lattice is the set of all integer linear combinations of the bases:

$$\mathcal{L}_1 = \{ z_1 \cdot (0.4, 0.8) + z_2 \cdot (1.2, 0.8) \mid z_1, z_2 \in \mathbb{Z} \}$$

For example, $(1.6, 1.2) \in \mathcal{L}_1$,
because $4 \cdot (0.4, 0.8) - 2 \cdot (1.2, 0.8) = (0.4, 1.6)$.



- ✧ Modules are exactly the same as vector spaces, but the scalars are selected from a **ring** instead of a field.
- ✧ One the most used rings is the ring of polynomials.

$$\mathcal{R} = \mathbb{R}[x]/(x^n + 1)$$

- ✧ It means each element in the ring is a polynomial with n coefficients in $\mathbb{R}$:
- ✧ Example: $3.6 + 5.1x + 2.4x^2 - 5.7x^3 \in \mathbb{R}[x]/(x^4 + 1)$.
- ✧ Each polynomial can be represented by a vector of its coefficients:

$$3.6 + 5.1x + 2.4x^2 - 5.7x^3 \equiv (3.6, 5.1, 2.4, -5.7)$$

- ✧ In simpler words, a polynomial in the ring $\mathcal{R} = \mathbb{R}[x]/(x^n + 1)$ is a vector of length n with elements in $\mathbb{R}$.

$$(a_0, a_1, a_2, \dots, a_{n-1}) \in \mathbb{R}[x]/(x^n + 1), \quad a_i \in \mathbb{R}$$

- ✧ So, a module over a ring of polynomials can be defined as:

$$\mathcal{M} = \{(x, y, z) \mid x, y, z \in \mathbb{R}[x]/(x^4 + 1)\}$$

- ✧ For example:

$$\begin{bmatrix} (3.6, 5.1, 2.4, -5.7) \\ (1.4, -2.5, 3.6, 4.7) \\ (-5.8, 6.9, -7.0, 8.1) \end{bmatrix} \in \mathcal{M}$$

Module-Lattice

A module-lattice is a lattice where b_i s are selected from a ring, such as the ring of polynomials.

$$\mathcal{L} = \left\{ \sum_{i=1}^n s_i \cdot \mathbf{b}_i \mid s_i \in \mathcal{R}, \mathbf{b}_i \in \mathcal{R}^k \right\}$$

- ✧ A discrete module can be a module-lattice. If we define the ring over the integers, we have a module-lattice.

Ring of Polynomials $\mathcal{R}_q$

$\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ is a ring, where each element in the ring is a polynomial.

- ✧ $\mathbb{Z}_q$: Each coefficient of the polynomial should be less than the prime number q . All the arithmetic operations are done modulo q .
- ✧ $[x]/x^n + 1$: The irreducible polynomial of the ring is $x^n + 1$. The maximum degree of the polynomial should be $n - 1$. In other words, number of terms in the polynomial is n .

Module-Lattice

A module-lattice is a lattice where b_i s are selected from a ring, such as the ring of polynomials.

$$\mathcal{L} = \left\{ \sum_{i=1}^n s_i \cdot \mathbf{b}_i \mid s_i \in \mathcal{R}, \mathbf{b}_i \in \mathcal{R}^k \right\}$$

- ✧ A discrete module can be a module-lattice. If we define the ring over the integers, we have a module-lattice.

Ring of Polynomials $\mathcal{R}_q$

$\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ is a ring, where each element in the ring is a polynomial.

- ✧ $\mathbb{Z}_q$: Each coefficient of the polynomial should be less than the prime number q . All the arithmetic operations are done modulo q .
- ✧ $[x]/x^n + 1$: The irreducible polynomial of the ring is $x^n + 1$. The maximum degree of the polynomial should be $n - 1$. In other words, number of terms in the polynomial is n .

✧ Example: $\mathcal{R}_7 = \mathbb{Z}_7[x]/(x^4 + 1)$ is a ring, where $q = 7$ and $n = 4$.

✧ For example, $2 + 4x + x^2 + 5x^3 \in \mathbb{Z}_7[x]/(x^4 + 1)$.

✧ Showing by vectors: $(2, 4, 1, 5) \in \mathbb{Z}_7[x]/(x^4 + 1)$.

✧ Basically, a module-lattice over the ring $\mathcal{R}_q^k$ is a matrix of polynomials, where k is the dimension of the matrix:

$$\begin{bmatrix} 14 + 35x + 12x^2 + 23x^3 \\ 19 + 10x + 8x^2 + 6x^3 \\ 25 + 16x + 14x^2 + 18x^3 \end{bmatrix} \in \mathbb{Z}_{37}^3[x]/(x^4 + 1)$$

✧ In other words, a vector of vectors of length n with elements in $\mathbb{Z}_q$:

$$\begin{bmatrix} (14, 35, 12, 23) \\ (19, 10, 8, 6) \\ (25, 16, 14, 18) \end{bmatrix} \in \mathbb{Z}_{37}^3/(x^4 + 1)$$

- ✧ Example: $\mathcal{R}_7 = \mathbb{Z}_7[x]/(x^4 + 1)$ is a ring, where $q = 7$ and $n = 4$.
- ✧ For example, $2 + 4x + x^2 + 5x^3 \in \mathbb{Z}_7[x]/(x^4 + 1)$.
- ✧ Showing by vectors: $(2, 4, 1, 5) \in \mathbb{Z}_7[x]/(x^4 + 1)$.
- ✧ Basically, a module-lattice over the ring $\mathcal{R}_q^k$ is a matrix of polynomials, where k is the dimension of the matrix:

$$\begin{bmatrix} 14 + 35x + 12x^2 + 23x^3 \\ 19 + 10x + 8x^2 + 6x^3 \\ 25 + 16x + 14x^2 + 18x^3 \end{bmatrix} \in \mathbb{Z}_{37}^3[x]/(x^4 + 1)$$

- ✧ In other words, a vector of vectors of length n with elements in $\mathbb{Z}_q$:

$$\begin{bmatrix} (14, 35, 12, 23) \\ (19, 10, 8, 6) \\ (25, 16, 14, 18) \end{bmatrix} \in \mathbb{Z}_{37}^3/(x^4 + 1)$$

LATTICES' HARD PROBLEMS

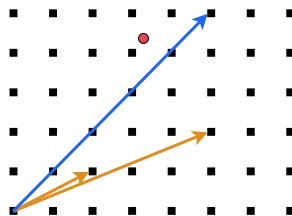
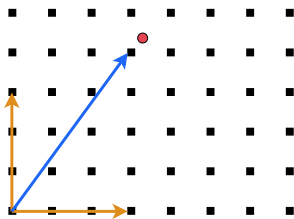
- ✧ There are some very hard mathematical problems in lattices:
 - ▷ Shortest Vector Problem (SVP)
 - ▷ Closest Vector Problem (CVP), even harder than SVP
- ✧ They are not always hard: Best-case, Average-case, Worst-case hardness.
- ✧ To build a cryptosystem, we need a problem that is as hard as *the worst-case* of CVP or SVP.
- ✧ Mathematically proven to be NP-hard, *not solvable in polynomial time*, **unlike IFP and DLP**.



Closest Vector Problem, left: best-case, right: worst-case

LATTICES' HARD PROBLEMS

- ✧ There are some very hard mathematical problems in lattices:
 - ▷ Shortest Vector Problem (SVP)
 - ▷ Closest Vector Problem (CVP), even harder than SVP
- ✧ They are not always hard: Best-case, Average-case, Worst-case hardness.
- ✧ To build a cryptosystem, we need a problem that is as hard as *the worst-case* of CVP or SVP.
- ✧ Mathematically proven to be NP-hard, *not solvable in polynomial time*, **unlike IFP and DLP**.



Closest Vector Problem, left: best-case, right: worst-case

BUILDING THE HARD PROBLEM

- ✧ CVP and SVP cannot be used to build a cryptosystem, because they need a very large lattice for maximum security.

Private Key

$$\mathbf{s} = \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{\ell-1} \end{bmatrix}, s_i \in \mathbb{Z}_q, \mathbf{s} \in \mathbb{Z}_q^\ell$$

Public Key

$$\mathbf{A} = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0\ell-1} \\ a_{10} & a_{11} & \dots & a_{1\ell-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k-1,0} & a_{k-1,1} & \dots & a_{k-1,\ell-1} \end{bmatrix}, a_{ij} \in \mathbb{Z}_q, \mathbf{A} \in \mathbb{Z}_q^{k \times \ell}.$$

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{s} = \begin{bmatrix} A_{00} \cdot s_0 + A_{01} \cdot s_1 + \dots + A_{0\ell-1} \cdot s_{\ell-1} \\ A_{10} \cdot s_0 + A_{11} \cdot s_1 + \dots + A_{1\ell-1} \cdot s_{\ell-1} \\ \vdots \\ A_{k-1,0} \cdot s_0 + A_{k-1,1} \cdot s_1 + \dots + A_{k-1,\ell-1} \cdot s_{\ell-1} \end{bmatrix}$$

- ✧ Given $\mathbf{A}$ and $\mathbf{t}$, and considering $k \geq \ell$, finding $\mathbf{s}$ is very easy!
- ▷ $\mathbf{s} \rightarrow \ell$ parameters (variables), $\mathbf{t} \rightarrow k$ parameters (equations) $\Rightarrow$ A square system of equations.

BUILDING THE HARD PROBLEM

- ✧ CVP and SVP cannot be used to build a cryptosystem, because they need a very large lattice for maximum security.

Private Key

$$\mathbf{s} = \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{\ell-1} \end{bmatrix}, s_i \in \mathbb{Z}_q, \mathbf{s} \in \mathbb{Z}_q^\ell$$

Public Key

$$\mathbf{A} = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0\ell-1} \\ a_{10} & a_{11} & \dots & a_{1\ell-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k-1,0} & a_{k-1,1} & \dots & a_{k-1,\ell-1} \end{bmatrix}, a_{ij} \in \mathbb{Z}_q, \mathbf{A} \in \mathbb{Z}_q^{k \times \ell}.$$

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{s} = \begin{bmatrix} A_{00} \cdot s_0 + A_{01} \cdot s_1 + \dots + A_{0\ell-1} \cdot s_{\ell-1} \\ A_{10} \cdot s_0 + A_{11} \cdot s_1 + \dots + A_{1\ell-1} \cdot s_{\ell-1} \\ \vdots \\ A_{k-1,0} \cdot s_0 + A_{k-1,1} \cdot s_1 + \dots + A_{k-1,\ell-1} \cdot s_{\ell-1} \end{bmatrix}$$

- ✧ Given $\mathbf{A}$ and $\mathbf{t}$, and considering $k \geq \ell$, finding $\mathbf{s}$ is very easy!
- ▷ $\mathbf{s} \rightarrow \ell$ parameters (variables), $\mathbf{t} \rightarrow k$ parameters (equations) $\Rightarrow$ A square system of equations.

LEARNING WITH ERRORS (LWE)

- ✧ But if we add a random error $\mathbf{e}$ to $\mathbf{t}$, $\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$, the number of equations is still k , but $\mathbf{e}$ adds k additional parameters. Now, the system has $k + \ell$ variables $\rightarrow$ an **underdetermined** system.

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} = \begin{bmatrix} A_{00} \cdot s_0 + A_{01} \cdot s_1 + \cdots + A_{0\ell-1} \cdot s_{\ell-1} + e_0 \\ A_{10} \cdot s_0 + A_{11} \cdot s_1 + \cdots + A_{1\ell-1} \cdot s_{\ell-1} + e_1 \\ \vdots \\ A_{k-1,0} \cdot s_0 + A_{k-1,1} \cdot s_1 + \cdots + A_{k-1,\ell-1} \cdot s_{\ell-1} + e_{k-1} \end{bmatrix}$$

Example

$$\mathbf{A} = \begin{bmatrix} 2 & 3 & 14 \\ 19 & 10 & 8 \\ 25 & 16 & 14 \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} 3 \\ 0 \\ 2 \end{bmatrix}, \quad \mathbf{e} = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}, \quad \mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} = \begin{bmatrix} 2 \cdot 18 + 3 \cdot 15 + 14 \cdot 14 + 1 \\ 19 \cdot 18 + 10 \cdot 15 + 8 \cdot 14 + 2 \\ 25 \cdot 18 + 16 \cdot 15 + 14 \cdot 14 + 3 \end{bmatrix}$$

- ✧ Learning the hidden structure (the secret polynomial $\mathbf{s}$) with errors $\mathbf{e}$ (LWE) is **not possible**!
- ✧ Downside: Again, a very high-dimensional lattice for maximum security $\rightarrow$ low performance.

LWE ON MODULES

- ✧ Solution: Substitute elements of the lattice with polynomials $\equiv$ using module-lattices.
- ✧ A vector of polynomials is a module-lattice ($\mathbf{A}$ has three elements in module-lattice, and $\mathbf{s}$ has one element in module-lattice):

$$\mathbf{A} = \begin{bmatrix} 2 + 3x + 14x^2 & 12 + 17x^2 & 18 + 2x + 12x^2 \\ 19 + 10x + 8x^2 & 6 + 9x + 19x^2 & 15 + 13x + 7x^2 \\ 25 + 16x + 14x^2 & 18 + 15x + 14x^2 & 13 + 24x + 15x^2 \end{bmatrix} \quad \mathbf{s} = \begin{bmatrix} 5 + 0x + 1x^2 \\ 1 + 4x + 6x^2 \\ 3 + 2x + 1x^2 \end{bmatrix}$$

✧ $\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$:

- ▷ $\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$ ($k \times \ell$ matrix of polynomials of length n),
 - ▷ $\mathbf{s} \in \mathcal{R}_q^\ell$ ($\ell \times 1$ vector of polynomials of length n),
 - ▷ $\mathbf{e} \in \mathcal{R}_q^k$ ($k \times 1$ vector of polynomials of length n),
 - ▷ $\Rightarrow \mathbf{t} \in \mathcal{R}_q^k$ ($k \times 1$ vector of polynomials of length n).
- ✧ Given $\mathbf{t}$ ($k \cdot n$ parameters or equations), $\mathbf{A}$, and the lattice ring $\rightarrow \mathbf{s}$ and $\mathbf{e}$ have $k \cdot n + \ell \cdot n$ parameters $\rightarrow$ Infeasible to solve $\rightarrow$ Module-Learning With Error (MLWE).

LWE ON MODULES

- ✧ Solution: Substitute elements of the lattice with polynomials $\equiv$ using module-lattices.
- ✧ A vector of polynomials is a module-lattice ($\mathbf{A}$ has three elements in module-lattice, and $\mathbf{s}$ has one element in module-lattice):

$$\mathbf{A} = \begin{bmatrix} 2 + 3x + 14x^2 & 12 + 17x^2 & 18 + 2x + 12x^2 \\ 19 + 10x + 8x^2 & 6 + 9x + 19x^2 & 15 + 13x + 7x^2 \\ 25 + 16x + 14x^2 & 18 + 15x + 14x^2 & 13 + 24x + 15x^2 \end{bmatrix} \quad \mathbf{s} = \begin{bmatrix} 5 + 0x + 1x^2 \\ 1 + 4x + 6x^2 \\ 3 + 2x + 1x^2 \end{bmatrix}$$

- ✧ $\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$:

- ▷ $\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$ ($k \times \ell$ matrix of polynomials of length n),
 - ▷ $\mathbf{s} \in \mathcal{R}_q^\ell$ ($\ell \times 1$ vector of polynomials of length n),
 - ▷ $\mathbf{e} \in \mathcal{R}_q^k$ ($k \times 1$ vector of polynomials of length n),
 - ▷ $\Rightarrow \mathbf{t} \in \mathcal{R}_q^k$ ($k \times 1$ vector of polynomials of length n).
- ✧ Given $\mathbf{t}$ ($k \cdot n$ parameters or equations), $\mathbf{A}$, and the lattice ring $\rightarrow \mathbf{s}$ and $\mathbf{e}$ have $k \cdot n + \ell \cdot n$ parameters $\rightarrow$ Infeasible to solve $\rightarrow$ Module-Learning With Error (**MLWE**).

✧ $\mathbf{A}$ is a $k \times \ell$ matrix, $\mathbf{s}$ is a $k \times 1$ vector, $\mathbf{e}$ is a $k \times 1$ vector $\Rightarrow \mathbf{t}$ is a $k \times 1$ vector.

✧ Assume $k = \ell = 3$:

$$\mathbf{A} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} \quad \mathbf{s} = \begin{bmatrix} s_0 \\ s_1 \\ s_2 \end{bmatrix} \quad \mathbf{e} = \begin{bmatrix} e_0 \\ e_1 \\ e_2 \end{bmatrix}$$

✧ So, t can be computed as follows:

$$\mathbf{t} = \begin{bmatrix} t_0 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} a_{00} \cdot s_0 + a_{01} \cdot s_1 + a_{02} \cdot s_2 + e_0 \\ a_{10} \cdot s_0 + a_{11} \cdot s_1 + a_{12} \cdot s_2 + e_1 \\ a_{20} \cdot s_0 + a_{21} \cdot s_1 + a_{22} \cdot s_2 + e_2 \end{bmatrix}$$

✧ Each a_{ij} , s_i , e_i , t_i , such that $i, j \in \{0, 1, 2\}$ are vectors of length n .

✧ MLWE: Considering $\mathbf{e}$, finding $\mathbf{s}$ is infeasible.

MLWE AND D-MLWE

- ✧ MLWE is at least as hard as LWE, but more efficient (needs lower-dimensional lattices).
- ✧ Smaller keys, faster computations, and efficient implementations without compromising security.
- ✧ It is proved that the average-case hardness of MLWE, i.e., when inputs $\mathbf{A}$, $\mathbf{s}$, and $\mathbf{e}$ are random, is **at least** as hard as the worst-case hardness of CVP:
 - ▷ Any solution to MLWE can be converted to a solution to CVP (reductions to worst-case CVP).
- ✧ One cannot even distinguish $\mathbf{t}$ (an instance of MLWE) from a randomly selected matrix of polynomials $\rightarrow$ Decisional-Module-Learning With Errors (**D-MLWE**).

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} = \begin{bmatrix} t_0 \\ t_1 \\ t_2 \end{bmatrix} \quad \text{vs} \quad \mathbf{t}' = \begin{bmatrix} (t'_{0,0}, t'_{0,1}, \dots, t'_{0,n-1}) \\ (t'_{1,0}, t'_{1,1}, \dots, t'_{1,n-1}) \\ (t'_{2,0}, t'_{2,1}, \dots, t'_{2,n-1}) \end{bmatrix}$$

MLWE AND D-MLWE

- ✧ MLWE is at least as hard as LWE, but more efficient (needs lower-dimensional lattices).
- ✧ Smaller keys, faster computations, and efficient implementations without compromising security.
- ✧ It is proved that the average-case hardness of MLWE, i.e., when inputs $\mathbf{A}$, $\mathbf{s}$, and $\mathbf{e}$ are random, is **at least** as hard as the worst-case hardness of CVP:
 - ▷ Any solution to MLWE can be converted to a solution to CVP (reductions to worst-case CVP).
- ✧ One cannot even distinguish $\mathbf{t}$ (an instance of MLWE) from a randomly selected matrix of polynomials $\rightarrow$ Decisional-Module-Learning With Errors (**D-MLWE**).

$$\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} = \begin{bmatrix} t_0 \\ t_1 \\ t_2 \end{bmatrix} \quad \text{vs} \quad \mathbf{t}' = \begin{bmatrix} (t'_{0,0}, t'_{0,1}, \dots, t'_{0,n-1}) \\ (t'_{1,0}, t'_{1,1}, \dots, t'_{1,n-1}) \\ (t'_{2,0}, t'_{2,1}, \dots, t'_{2,n-1}) \end{bmatrix}$$

ARITHMETIC OPERATIONS IN $\mathcal{R}_q$

✧ Let $a = (a_0, a_1, \dots, a_{n-1})$ and $b = (b_0, b_1, \dots, b_{n-1})$ be two polynomials in $\mathcal{R}_q$.

✧ Addition and Subtraction:

$$a + b = \begin{bmatrix} (a_0 + b_0) \pmod{q} \\ (a_1 + b_1) \pmod{q} \\ \vdots \\ (a_{n-1} + b_{n-1}) \pmod{q} \end{bmatrix}, \quad a - b = \begin{bmatrix} (a_0 - b_0) \pmod{q} \\ (a_1 - b_1) \pmod{q} \\ \vdots \\ (a_{n-1} - b_{n-1}) \pmod{q} \end{bmatrix}$$

✧ Multiplication:

$$a \cdot b = a \cdot b \pmod{\mathcal{R}_q}$$

✧ Multiplication is not pointwise, it is a convolution.

ADDITION AND SUBTRACTION IN $\mathcal{R}_q$

✧ $a = 2 + 8x + 4x^2 + x^3$ and $b = 10 + 2x + 7x^2 + 9x^3$ are two polynomials in $\mathcal{R}_q = \mathbb{Z}_{11}[x]/(x^4 + 1)$.

✧ Addition and Subtraction are performed element-wise:

$$a + b = \begin{bmatrix} 2 \\ 8 \\ 4 \\ 1 \end{bmatrix} + \begin{bmatrix} 10 \\ 2 \\ 7 \\ 9 \end{bmatrix} = \begin{bmatrix} (2 + 10) \pmod{11} \\ (8 + 2) \pmod{11} \\ (4 + 7) \pmod{11} \\ (1 + 9) \pmod{11} \end{bmatrix} = \begin{bmatrix} 1 \\ 10 \\ 0 \\ 10 \end{bmatrix} = 1 + 10x + 10x^3$$

$$a - b = \begin{bmatrix} 2 \\ 8 \\ 4 \\ 1 \end{bmatrix} - \begin{bmatrix} 10 \\ 2 \\ 7 \\ 9 \end{bmatrix} = \begin{bmatrix} (2 - 10) \pmod{11} \\ (8 - 2) \pmod{11} \\ (4 - 7) \pmod{11} \\ (1 - 9) \pmod{11} \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \\ 8 \\ 3 \end{bmatrix} = 3 + 6x + 8x^2 + 3x^3$$

MULTIPLICATION IN $\mathcal{R}_q$

✧ $a = 2 + 8x + 4x^2 + x^3$ and $b = 10 + 2x + 7x^2 + 9x^3$ are two polynomials in $\mathcal{R}_q = \mathbb{Z}_{11}[x]/(x^4 + 1)$.

✧ Multiplication:

$$\begin{aligned}a \cdot b &= (2 + 8x + 4x^2 + x^3) \cdot (10 + 2x + 7x^2 + 9x^3) \pmod{\mathcal{R}_q} \\&= 20 + 4x + 14x^2 + 18x^3 + 80x + 16x^2 + 56x^3 + 72x^4 + 40x^2 + 8x^3 + 28x^4 + 36x^5 + \\&\quad 10x^3 + 2x^4 + 7x^5 + 9x^6 \pmod{\mathcal{R}_q} \\&= 20 + 84x + 70x^2 + 92x^3 + 102x^4 + 43x^5 + 2x^6 \pmod{\mathcal{R}_q}.\end{aligned}$$

✧ Then, substitute $x^4 \equiv -1 \pmod{\mathcal{R}_q}$, $x^5 \equiv -x \pmod{\mathcal{R}_q}$, $x^6 \equiv -x^2 \pmod{\mathcal{R}_q}$:

$$\begin{aligned}20 + 84x + 70x^2 + 92x^3 - 102 - 43x - 2x^2 &\pmod{\mathcal{R}_q} \\&= -82 + 41x + 68x^2 + 92x^3 \pmod{\mathcal{R}_q} \\&= 6 + 8 + 2x^2 + 4x^3 \pmod{\mathcal{R}_q}\end{aligned}$$

✧ So, $a \cdot b = 6 + 8x + 2x^2 + 4x^3$. Very hard!

Thank you!