

Lattice-Related Hard Problems

Dr. Essam Ghadafi

CyBOK © Crown Copyright, The National Cyber Security Centre 2025, licensed under the Open Government Licence <http://www.nationalarchives.gov.uk/doc/open-government-licence/>

CyBOK MAPPING

The lecture maps to the following CyBOK Knowledge Areas:

- Systems Security → Cryptography
- Infrastructure Security → Applied Cryptography

OUTLINE

- Worst-case vs Average-case Problems
 - Definition and key differences
 - Examples in cryptography
- SVP and its Variants (Shortest Vector Problem)
 - Problem description, Hardness and Variants
- CVP and its Variants (Closest Vector Problem)
 - Problem description, Hardness and Variants
- SIS (Short Integer Solution) Problem
 - Problem description, Hardness and Variants
- LWE (Learning With Errors) Problem
 - Problem description, Hardness and Variants

WORST-CASE VS. AVERAGE-CASE PROBLEMS

- **Average-Case Problems:** The attacker must succeed in solving some random instances of the problem

Example: Average-Case Factoring Problem: Given a composite integer $N = p \times q$ from a distribution D_N over products of two large primes, find p and q
- **Worst-Case Problems:** The attacker must succeed in solving all instances of the problem

Example: Worst-Case Factoring Problem: Given a composite integer $N = p \times q$, where p and q are large primes, find p and q

Some problems are hard in the worst-case but easy on average. Basing security on worst-case hardness provides stronger guarantees

SHORTEST VECTOR PROBLEM (SVP)

💡 **Intuition:** Which point in the point grid is closest to the origin point (centre of the grid)?

The Shortest Vector Problem (SVP):

- Given a basis $\mathbf{B}$ for a lattice $L(\mathbf{B})$, find the shortest non-zero vector $\mathbf{v}$ in the lattice

Mathematically:

$$\|\mathbf{v}\| = \min_{\mathbf{w} \in L(\mathbf{B}) \setminus \{\mathbf{0}\}} \|\mathbf{w}\|$$

equivalently

$$\|\mathbf{v}\| = \lambda_1(L(\mathbf{B}))$$

Remember the 1st successive minimum λ_1 denotes the shortest non-zero vector in the lattice

SHORTEST VECTOR PROBLEM (SVP)

How hard is the problem?

- Hard because finding the shortest vector is NP-hard
- Easy to solve when the basis vectors $\{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n\}$ are *orthogonal*, i.e. $\mathbf{b}_i^T \mathbf{b}_j = 0$ for all $i, j \in \{1, \dots, n\}$ where $i \neq j$. In this case the shortest vector is the shortest base vector $\mathbf{b}_i$

$$\|\mathbf{v}_{\text{SVP}}\| = \min_{1 \leq i \leq n} \|\mathbf{b}_i\|$$

APPROXIMATE SVP (γ -SVP)

A related variant to SVP

The goal is to find a vector $\mathbf{v}$ that is at most γ times the length of the shortest vector where $\gamma \geq 1$

- Given a basis $\mathbf{B}$ for a lattice $L(\mathbf{B})$, find a vector $\mathbf{v} \in L(\mathbf{B}) \setminus \{\mathbf{0}\}$ where $\|\mathbf{v}\| \leq \gamma \cdot \lambda_1(L(\mathbf{B}))$

Remember the 1st successive minimum λ_1 denotes the shortest non-zero vector in the lattice

- The larger γ , the easier the problem. 1-SVP is SVP

SVP VARIANTS

- **Short Independent Vectors Problem (SIVP)** denoted by (SIVP_γ) :
Input: Basis $\mathbf{B} \in \mathbb{Z}_q^{m \times n}$ for $L(\mathbf{B})$
Task: Find n linearly independent vectors $\mathbf{v}_1, \dots, \mathbf{v}_n \in L(\mathbf{B})$ such that $\|\mathbf{v}_i\| \leq \gamma \lambda_n(L(\mathbf{B}))$ for all $i = 1, \dots, n$
- **GAP SVP (GapSVP_γ):**
Input: Basis $\mathbf{B} \in \mathbb{Z}_q^{m \times n}$ for $L(\mathbf{B})$
Task: Decide whether $\lambda_1(L(\mathbf{B})) \leq 1$ or $\lambda_1(L(\mathbf{B})) > \gamma$

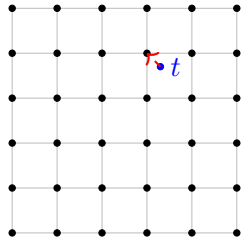
CLOSEST VECTOR PROBLEM (CVP)

💡 **Intuition:** From a location (which may not be a point) on the grid, find the nearest grid point

Closest Vector Problem (CVP)

- Given a basis $\mathbf{B}$ for a lattice $L(\mathbf{B})$ and a target point $\mathbf{t}$, find the closest lattice vector $\mathbf{v} \in L(\mathbf{B})$

Mathematically: $\min_{\mathbf{v} \in L(\mathbf{B})} \|\mathbf{v} - \mathbf{t}\|$



- Harder than SVP and is NP-hard
 - SVP is a special case where $\mathbf{t} = \mathbf{0}$. If one can solve CVP, one can solve SVP, i.e. $\text{SVP} \leq \text{CVP}$

APPROXIMATE CVP (γ -CVP)

The **Approximate CVP (γ -CVP)** is a variant of CVP

Instead of finding the closest vector $\mathbf{v} \in L(\mathbf{B})$ to $\mathbf{t}$, the task is instead to find $\mathbf{v} \in L(\mathbf{B})$ where the distance between $\mathbf{v}$ and $\mathbf{t}$ is at most γ times the distance between $\mathbf{t}$ and the closest vector in the lattice

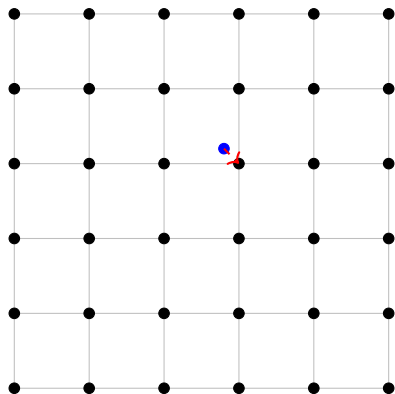
Mathematically:

$$\min_{\mathbf{v} \in L(\mathbf{B})} \|\mathbf{v} - \mathbf{t}\| \leq \gamma \cdot \min_{\mathbf{v}' \in L(\mathbf{B})} \|\mathbf{v}' - \mathbf{t}\|$$

- At least as hard as γ -SVP, i.e. $\gamma\text{-SVP} \leq \gamma\text{-CVP}$

BOUNDED DISTANCE DECODING (BDD)

The **Bounded Distance Decoding (BDD)** problem is a special case of CVP where the target $\mathbf{t}$ is close to a lattice point

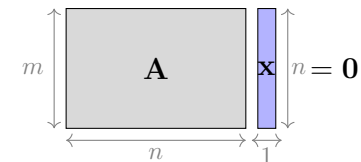


SHORT INTEGER SOLUTION (SIS)

The **SIS** problem introduced by Ajtai [1] is parametrised by parameters m, n, q, γ and is denoted by $\text{SIS}_{m,n,q,\gamma}$

- Given a random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, find a short non-zero vector $\mathbf{x} \in \mathbb{Z}^n$ such that $\|\mathbf{x}\| \leq \gamma$ and

$$\mathbf{A}\mathbf{x} \equiv \mathbf{0} \pmod{q}$$



- Generally, $\gamma < qn$, as otherwise, $\mathbf{s} = (q, 0, \dots, 0)$ is a valid solution
- The bigger m , the harder the problem
- The problem is trivial when $n < m$

SHORT INTEGER SOLUTION (SIS)

SIS is used in some hash functions and digital signatures

The **Inhomogeneous Short Integer Solution (ISIS)** problem is similar to SIS, except the right-hand side is $\mathbf{y} \in \mathbb{Z}_q^n$ instead of $\mathbf{0}$

ISIS is parametrised by m, n, q, γ and denoted by $\text{ISIS}_{m,n,q,\gamma}$

- Given a random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ and a random vector $\mathbf{y} \in \mathbb{Z}_q^m$ find a short vector $\mathbf{x} \in \mathbb{Z}^n$ such that $\|\mathbf{x}\| \leq \gamma$ and

$$\mathbf{Ax} \equiv \mathbf{y} \pmod{q}$$

- ISIS is as hard as SIS
 - SIS requires solving $\mathbf{Ax} = \mathbf{0}$ vs. solving $\mathbf{Ax} - \mathbf{y} = \mathbf{0}$ in ISIS

SIS AS A LATTICE PROBLEM

SIS is a lattice problem as well

We use the matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ to define the lattice:

$$L(\mathbf{A}) = \{\mathbf{e} \in \mathbb{Z}^n : \mathbf{Ae} = \mathbf{0} \pmod{q}\}$$

SIS is then defined as finding a short vector $\mathbf{e}$ in this lattice

LWE – SETTING THE SCENE

Consider a system of m linear equations of the form:

$$\sum_{i=1}^n a_{1,i} s_i = b_1 \pmod{q}$$

$\vdots$

$$\sum_{i=1}^n a_{m,i} s_i = b_m \pmod{q}$$

where $a_{j,i} \in \mathbb{Z}_q$ are known coefficients, $b_j \in \mathbb{Z}_q$ are known results, and $s_i \in \mathbb{Z}_q$ are the unknowns we seek to solve for.

We can represent the system as

$$\mathbf{As} = \mathbf{b} \pmod{q}$$

This system of equations is straightforward to solve (in poly time), e.g., using Gaussian elimination

LEARNING WITH ERRORS (LWE)

LWE introduced by Regev [6] adds small noise to linear equations, making them hard to solve even for quantum computers

LWE and its variants have been used to construct various types of cryptosystems, including:

- Public-Key Encryption
- Key Exchange
- Identity-Based Encryption
- Zero-Knowledge Proofs
- ...

LEARNING WITH ERRORS (LWE)

Definition: Let m, n, q be positive integers, and let χ be a probability distribution over $\mathbb{Z}_q$

The (search) LWE problem denoted by $(\text{LWE}_{m,n,q,\chi})$ is:

- Randomly choose a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$
- Choose a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$
- Choose a random error/noise vector $\mathbf{e}$ according to χ^m
 - i.e., each coordinate e_i is independently drawn from χ

Input: $(\mathbf{A}, \mathbf{b})$ where $\mathbf{A}\mathbf{s} + \mathbf{e} \equiv \mathbf{b} \pmod{q}$ where $\mathbf{b} \in \mathbb{Z}_q^m$

Task: Recover $\mathbf{s}$

DECISIONAL LWE

The (decision) LWE problem denoted by $(\text{D-LWE}_{m,n,q,\chi})$ is:

- $\mathbf{A}$, $\mathbf{s}$, and $\mathbf{e}$ are all chosen as in the search LWE problem
- **Challenge Generation:**
 - **Case 0** (real): Compute $\mathbf{b}_0 = \mathbf{A}\mathbf{s} + \mathbf{e} \pmod{q}$
 - **Case 1** (fake): Choose $\mathbf{b}_1 \in \mathbb{Z}_q^m$ uniformly at random

Input: $(\mathbf{A}, \mathbf{b}_b)$ where $b \in \{0, 1\}$ is chosen uniformly at random

Task: Guess the case b

Interestingly, LWE can be reduced to D-LWE

- An algorithm solving D-LWE can also solve LWE. The reverse implication is trivial
 - This means equivalence of the two variants of LWE

HARDNESS OF LWE

- If one can solve $\text{SIS}_{m,n,q,\gamma}$, one can solve $\text{LWE}_{m,n,q,\chi}$, i.e. $\text{LWE}_{m,n,q,\chi} \leq \text{SIS}_{m,n,q,\gamma}$

- If one can solve $\text{LWE}_{m,n,q,\chi}$, one can solve GapSVP_γ
 - e.g. [6, 2]: $\text{GapSVP}_\gamma, \text{SIVP}_\gamma \leq \text{LWE}_{m,n,q,\chi}$

Note: In [6] the reduction is *quantum*, while in [2] it is *classical* but only for *polynomial modulus* q , with some losses in dimension and noise

RING-LWE – SETTING THE SCENE

Ring-LWE (R-LWE) [4] is a LWE variant adapted to polynomial rings (instead of integers) to improve efficiency

Remember: A polynomial is of the form $\sum_{i=0}^d c_i x^i$, where:

- x is the indeterminate, d is the degree of the polynomial (i.e. the highest power of x), and $c_i \in \mathbb{Z}$ are the coefficients

Polynomials can be represented by their coefficient vector

$$\mathbf{c} = (c_0, c_1, \dots, c_d)$$

- **Polynomial Ring R :** Let $R = \mathbb{Z}[x]/f(x)$ for some monic polynomial $f(x)$ of degree $d = n$, (e.g. $f(x) = x^n + 1$)
 - Elements of R are polynomials of degree $< n$ with integer coefficients
 - ▶ R is a set of polynomials reduced modulo $f(x)$
- **Polynomial Ring R_q :** Let $R_q = \mathbb{Z}_q[x]/f(x)$
 - Elements of R_q are polynomials of degree $< n$ with coefficients in $\mathbb{Z}_q$

Definition: Given polynomial rings R and R_q , and an error distribution χ over small elements of ring R

The **(Search) Ring-LWE (R-LWE)** problem, denoted by $(R-LWE_{q,n,\chi})$, is defined as follows:

- Sample polynomials $a_1, \dots, a_n \in R_q$ independently and uniformly at random. Let $\mathbf{a} = (a_1, \dots, a_n)$
- Sample error polynomials $e_1, \dots, e_n$ independently from χ . Let $\mathbf{e} = (e_1, \dots, e_n)$

Input: $(\mathbf{a}, \mathbf{b} = s \mathbf{a} + \mathbf{e} \bmod qR)$, for some fixed polynomial $s \in R_q$

Task: Recover s

Note: $\bmod qR$ means reducing polynomial coefficients modulo q , where qR is the ideal in R generated by q

The **Decisional Ring-LWE (D-R-LWE)** problem denoted by $(D-R-LWE_{q,n,\chi})$ can be defined similarly where the task is to distinguish $(\mathbf{a}, \mathbf{b})$, where

$$\mathbf{b} = s \mathbf{a} + \mathbf{e} \bmod qR$$

from a uniform tuple $(\mathbf{a}, \mathbf{b}) \in R_q^n \times R_q^n$

Lyubashevsky, et al. [4, 5] gave the following results

$$\text{Ideal-SIVP}_\gamma(\text{worst case}) \leq_{\text{quantum polytime}} D-R-LWE_{q,n,\chi}$$

- If you can solve D-R-LWE, there is a **quantum algorithm** that can solve $\text{Ideal-SIVP}_\gamma(\text{worst case})$

$$R-LWE_{q,n,\chi} \leq D-R-LWE_{q,n,\chi}$$

- If you can solve Decisional R-LWE, there is a **classical algorithm** that can solve Search R-LWE

RING-LWE vs. LWE

- R-LWE yield more efficient constructions
 - Multiplication in the polynomial ring is more efficient, e.g. using FFT-like techniques
 - Smaller element representations (Elements from R_q vs. elements from $\mathbb{Z}$)
 - ▶ Storage cost in standard LWE: $O(m \cdot n)$ integers (typically $O(n^2)$ since $m = O(n)$)
 - ▶ Storage cost in Ring-LWE: $O(n)$ integers

- Like LWE's relation to SIS, Ring-LWE retains worst-case hardness due to its close connection to Ring-SIS

Note: R-LWE is defined over **structured lattices** (ideal lattices with algebraic structure), whereas LWE works over **general, unstructured lattices**

- R-LWE assumes hardness for a special class of lattices

MODULE-LWE — INTUITION

💡 **Intuition:** M-LWE is a bridge between (unstructured) LWE and structured Ring-LWE

- LWE: Elements are simple vectors over $\mathbb{Z}_q^n$ (unstructured)
- Ring-LWE: Elements are polynomials from a ring R_q (highly structured)
- M-LWE: Elements are vectors of polynomials in R_q^k , more structured than LWE but less structured than Ring-LWE
 - Interpolates between LWE and Ring-LWE

Ring-LWE is a special case of Module-LWE where $k = 1$

MODULE-LWE – DEFINITION

Similarly to the setup for Ring-LWE, given

$$R = \mathbb{Z}[x]/f(x), \quad R_q = R/qR,$$

a module rank k , and an error distribution χ over small elements of R

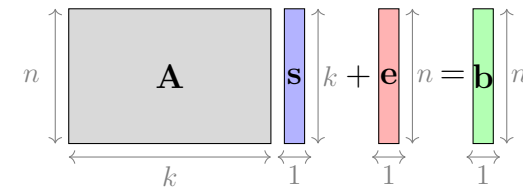
The (Search) **Module-LWE (M-LWE)** problem, denoted by $(\text{M-LWE}_{q,n,k,\chi})$, is:

- Sample elements $\mathbf{a}_1, \dots, \mathbf{a}_n \in R_q^k$ independently and uniformly at random. Let $\mathbf{A} = (\mathbf{a}_1, \dots, \mathbf{a}_n) \in R_q^{n \times k}$.
- Independently sample error elements $e_1, \dots, e_n \in R_q$ from χ . Let $\mathbf{e} = (e_1, \dots, e_n) \in R_q^n$.

Input: $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}) \in R_q^{n \times k} \times R_q^n$, for some secret vector of polynomials $\mathbf{s} \in R_q^k$

Task: Recover $\mathbf{s}$

MODULE-LWE – A CLOSER LOOK



Note: Some formulations explicitly sample $\mathbf{s}$ from a distribution χ_s (e.g. discrete Gaussian or binomial); others assume $\mathbf{s}$ is uniform in R_q^k

Comparison with Ring-LWE:

- Ring-LWE is a special case of M-LWE when $k = 1$

HARDNESS OF MODULE-LWE

Langlois & Stehlé [3] showed that

$$\text{Module-SIVP}_\gamma \leq_{\text{quantum polytime}} \text{M-LWE}_{q,n,k,\chi}$$

- If $\text{M-LWE}_{q,n,k,\chi}$ can be solved efficiently, then worst-case $\text{Module-SIVP}_\gamma$ can also be solved efficiently (by a quantum polynomial-time algorithm)

EFFICIENCY OF MODULE-LWE

Storage cost in Module-LWE is $O(n \cdot k)$

NIST-standard CRYSTALS-Kyber (key encapsulation mechanism) and CRYSTALS-Dilithium (signature) both rely on Module-LWE

Example:

In CRYSTALS-Kyber 512, the M-LWE parameters used are:
 $n = k = 2$, $f(x) = X^{256} + 1$, and $q = 3329$

DECISIONAL MODULE-LWE – DEFINITION

The setup is similar to search M-LWE

Decisional Module-LWE (D-M-LWE): The problem $(\text{D-M-LWE}_{q,n,k,\chi})$ is to **distinguish** between:

$$(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}),$$

for a secret $\mathbf{s} \in R_q^k$, an error vector $\mathbf{e} \in R_q^n$ with entries independently sampled from χ , and $\mathbf{A}$ is a uniformly random matrix over $R_q^{n \times k}$; and

$$(\mathbf{A}, \mathbf{b}) \in R_q^{n \times k} \times R_q^n$$

sampled uniformly at random

MAIN TAKEAWAYS

- **Worst-Case vs. Average-Case Problems:**
 - Worst-case: Focus on the hardest instances (e.g., SVP)
 - Average-case: Focus on typical instances (e.g., SIS)
- **Shortest Vector Problem (SVP):**
 - Finding the shortest non-zero vector in a lattice
- **Closest Vector Problem (CVP):**
 - Finding the closest lattice vector to a given point
- **Short Integer Solution (SIS):**
 - Finding short integer solutions to modular equations
- **Learning With Errors (LWE):**
 - Recovering a secret vector from noisy linear equations
- **Ring Learning With Errors (R-LWE):** A variant of LWE defined over polynomial rings
 - Recovering a secret polynomial from noisy polynomial equations
- **Module Learning With Errors (M-LWE):** A generalization of LWE and R-LWE that balances efficiency and security across modules

ADDITIONAL RESOURCES & READING

- D. Micciancio. Efficient reductions among lattice problems. In ACM-SIAM symposium on Discrete algorithms (SODA), 2008.

REFERENCES

- [1] M. Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing (STOC)*, 1996.
- [2] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, D. Stehlé. Classical hardness of learning with errors. In *ACM symposium on Theory of Computing (STOC)*, 2013.
- [3] A. Langlois, D. Stehlé. Worst-Case to Average-Case Reductions for Module Lattices, In *Designs, Codes and Cryptography*, 2015.
- [4] V. Lyubashevsky, C. Peikert, O. Regev. On ideal lattices and learning with errors over rings. In *Eurocrypt*, 2010.
- [5] V. Lyubashevsky, C. Peikert, O. Regev. On ideal lattices and learning with errors over rings. In *J. ACM*, 2013.
- [6] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *J. ACM* 56, 6, 2009.