

Exploring the Underlying Assumptions of Lattice Constructions: A Theoretical Investigation

Final thesis submitted to the Indian Statistical Institute,
Kolkata, for award of the degree
of

Masters of Technology in Cryptology and Security
by

Arkaprava Datta
[Roll No: CrS2201]

Under the guidance of

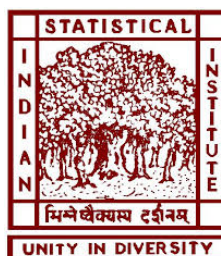
Dr. Rishiraj Bhattacharyya

University of Birmingham

and Institute Supervisor

Dr. Goutam Paul

Indian Statistical Institute, Kolkata



Cryptology and Security Research Unit
INDIAN STATISTICAL INSTITUTE, KOLKATA

July, 2024

Certificate

This is to certify that the thesis entitled “**Exploring the Underlying Assumptions of Lattice Constructions: A Theoretical Investigation**” submitted by **Arkaprava Datta (CrS2201)** is a bona fide record of the research work carried out by him under my supervision and guidance, and that no part of this thesis has been submitted for the award of any other degree or diploma.

Dr. Rishiraj Bhattacharyya
Assistant Professor
School of Computer Science
University of Birmingham



Goutam Paul
Cryptology & Security Research Unit (CSRU)
Indian Statistical Institute, Kolkata

July 12, 2024

Acknowledgements

I would like to express my deepest gratitude to all those who provided me with the support and guidance necessary to complete this master's thesis.

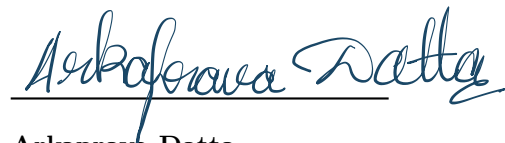
First and foremost, I would like to thank my advisor, Dr. Rishiraj Bhat-tacharyya, for his invaluable expertise, encouragement, and patience throughout this research journey. His insightful feedback and unwavering support have been instrumental in shaping this work.

I am also grateful to my internal supervisor, Dr. Goutam Paul and one of my teachers at the ISI Kolkata, Dr. Pratyay Mukherjee, for their time, constructive critiques, and invaluable suggestions that significantly improved the quality of this thesis.

I extend my heartfelt thanks to my colleagues, seniors, and friends for their camaraderie, intellectual discussions, and moral support. The resources and facilities provided by the Indian Statistical Institute have been crucial to the completion of this work.

Lastly, I am deeply indebted to my family for their unwavering love, understanding, and encouragement. To my parents, Mr. Dipankar Datta, and Mrs. Nivedita Datta, your belief in me has been my greatest source of strength.

Thank you all for your support and encouragement.

A handwritten signature in black ink, reading "Arkaprava Datta". The signature is written in a cursive style and is positioned above a horizontal line.

Arkaprava Datta

Indian Statistical Institute, Kolkata
M.Tech. CrS (CrS2201)

Abstract

Owing to its adaptability in cryptographic protocols and possible defence against quantum attacks, lattice-based cryptography has become a very attractive topic. This survey explores the fundamental hard problems in lattice theory, such as the Shortest Vector Problem (SVP), the Closest Vector Problem (CVP), and the Learning With Errors (LWE) problem, which form the cornerstone of lattice-based cryptosystems. We explore the intricate mathematical structures and specifics of all of these problems, highlighting their computational difficulty and importance.

In addition, we look at the idea of “crypto dark matter,” which refers to cryptographic structures and protocols that function outside of the accepted frameworks for cryptographic analysis and application. Our aim is to gain knowledge regarding the incorporation of lattice-based hard problems into the crypto dark matter framework through a review of the literature and uncover new dimensions of security and functionality that challenge traditional approaches.

This analysis emphasises the application of current developments in lattice-based cryptography in building secure cryptographic primitives while offering a thorough overview of the field. In the era of quantum computing, our studies highlight the importance of lattice-based hard problems as a frontier for innovative cryptography research as well as a solid foundation for strong cryptographic systems. The aim of this study is to help researchers and practitioners better understand how advanced cryptographic applications interact with lattice theory, which will ultimately lead to the development of cryptographic solutions that are more effective and secure.

Contents

Certificate	ii
Acknowledgements	iii
Abstract	v
1 Introduction	1
2 Preliminaries	5
3 Lattice	10
4 Lattice-based “Hard” Problems	11
4.1 <i>Shortest Vector Problem (SVP)</i>	11
4.2 <i>Closest Vector Problem (CVP)</i>	12
4.3 <i>Learning With Error (LWE)</i>	13
4.4 <i>Shortest Integer Solution (SIS)</i>	15
4.5 <i>NTRU Assumption</i>	16
5 Lattice and Crypto Dark Matter	17
5.1 <i>“Exploring Crypto Dark Matter: New Simple PRF Candidates and Their Application”</i>	18
5.1.1 Aim	18
5.1.2 Candidate PRFs	19
5.1.3 Advantages	19
5.1.4 Hardness	20
5.1.5 Hardness Reduction (Elaborated)	21
5.2 <i>“Crypto Dark Matter on the Torus”</i>	23
5.2.1 Aim	24
5.2.2 Candidate OPRF	25
5.2.3 Hardness	27
6 Conclusion and Future Developments	29
Bibliography	31

Introduction

The word “**Cryptography**” comes from the Greek word “**Kryptos**”, which means “hidden” or “secret” and the word “**Graphein**”, which means ‘to write’. So, cryptography literally means “secret writing”. It is the practice and study of hiding information.

Cryptography is critical because it preserves the security and privacy of our data in an age where information is frequently traded online. It keeps sensitive information such as personal information, financial transactions, and confidential conversations from being accessed or modified with by unauthorised parties. Cryptography promotes trust in digital systems by providing secure communication, data integrity, and authentication, so reducing identity theft, fraud, and other nefarious acts. Without cryptography, our digital interactions would be subject to eavesdropping and misuse, therefore it is critical for protecting our personal and professional life in the digital age.

The message, to be communicated, is called **plain text**. In order to communicate securely, the plain text is converted in to a secret message (or code) called **cipher text**, using mathematical tools. This process is called **Encryption** and the tools are called **encryption keys**. On the other hand, the receiver needs to get the original message from the cipher text. The receiver uses a specific mathematical tool corresponding to the key(s), to get the original message back. This process is called **Decryption** and the corresponding tool is called **decryption key**. The entire process of selecting the keys, encryption and decryption, together is called a **Cryptosystem**.

More formally, a cryptosystem is a five-tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where :

- \mathcal{P} is a finite set of possible plain texts
- \mathcal{C} is a finite set of possible cipher texts
- \mathcal{K} , the key space, is a finite set of possible keys
- For each $K \in \mathcal{K}$, there is an encryption rule $e_K \in \mathcal{E}$ and a corresponding decryption rule $d_K \in \mathcal{D}$. Each $e_K : \mathcal{P} \rightarrow \mathcal{C}$ and $d_K : \mathcal{C} \rightarrow \mathcal{P}$ are functions such that $d_K(e_K(x)) = x$ for every plain text element $x \in \mathcal{P}$.

Most cryptography protocols fall into one of the three main categories, or their hybrids:

- **Public Key(PKC)**: PKC involves the use of key pairs, consisting of a public key and a private key. The public key is widely distributed and used for encryption, while the private key is kept secret and used for decryption.

This allows for secure communication between parties without the need for a shared secret key.

- **Symmetric Key:** Symmetric key cryptography uses the same key for both encryption and decryption. It is generally faster than public key cryptography and is commonly used for encrypting large amounts of data efficiently.
- **Hash Functions:** A hash function is a mathematical function that takes an input (or message) and produces a fixed-size string of characters, typically a hash value or digest. It is deterministic with a fixed output size. Given a hash output, it should be computationally infeasible to determine the original input (**Pre-image Resistance**). A small change in the input should result in a significantly different hash output (**Avalanche Effect**). It is computationally infeasible to find two different inputs that produce the same hash output (**Collision Resistance**).

Hash functions are widely used in cryptography for integrity checks, digital signatures, password storage, and various other security applications where data integrity and uniqueness are crucial.

The Evolution of Cryptography: Cryptography has evolved dramatically over time, beginning with archaic ways of protecting sensitive information based on letter substitution and transposition. These approaches were critical for military operations and diplomatic correspondence, providing secrecy through obscurity. The development of more advanced encryption techniques in the twentieth century was a huge step forward in terms of security and complexity. Cryptography now plays an important role in digital security, ensuring data integrity, secrecy, and authenticity across several platforms. Looking ahead, the introduction of quantum computing presents new concerns. Continuing research in post-quantum cryptography attempts to create resilient encryption systems that can withstand quantum assaults, representing the next frontier in securing sensitive information in an increasingly digitised society.

Post-quantum Cryptography (PQC): Let's talk about one of cryptography's hottest topics right now: **post-quantum cryptography (PQC)**. With improvements in quantum computing technology, there is rising concern about quantum computers' ability to break existing encryption techniques that now secure critical data. If quantum computers become feasible and widely available, they might make much of our current digital infrastructure vulnerable to attacks, jeopardising sensitive data, financial transactions, and crucial communication routes. Post-quantum cryptography aims to develop algorithms that can withstand attacks from both classical and quantum computers, protecting data secrecy, integrity, and authenticity against developing threats. It's a proactive step towards preserving trust and security.

Several approaches are being investigated in the development of post-quantum cryptography (PQC), with each seeking to produce new cryptography algorithms that are resistant to quantum computer attacks.

Let's have a brief introduction to a few of these.

- **Lattice-based Cryptography:** This approach uses the computational complexity of problems related to lattices in high-dimensional spaces. So, what is a lattice? Lattice is an abstract mathematical structure. We will discuss “Lattice” in detail in the later sections.
- **Code-based Cryptography:** Code-based cryptography is a subset of post-quantum cryptography that uses error-correcting codes as the basis for encryption and decryption. The fundamental idea is to exploit the difficulty of decoding certain types of structured codes to protect against assaults, especially those from quantum computers. Error-correcting codes are designed to correct errors that occur during transmission or storage of data, and their security stems from the difficulty of finding the original message given only the encoded message and the public parameters.
- **Isogeny-based Cryptography:** An elliptic curve is given by $y^2 + c_1xy + c_3y = x^3 + c_2x^2 + c_4x + c_6$ [**Weiestrass General Form**]. Informally, a rational map is a function between two curves such that the coordinate functions are each defined by a ratio of polynomials. An **isogeny** is a rational map that preserves the addition operation, defined on the points on the elliptic curve.

These mathematical structures are utilised in this approach to build defenses against quantum attacks.

These examples are not exhaustive. There are other approaches for PQC as well, such as multivariate polynomial cryptography, hash-based cryptography, etc.

Current Developments in PQC: The first question that comes to mind is, “Why is the focus only on lattice here?” The answer lies within the recent breakthroughs in the field of PQC. In 2022, there was an attack on one of the standard isogeny-based protocols. It demonstrated that what was previously considered secure against quantum attacks, could be broken in practice under certain conditions. There are also existing attacks on Code-based cryptography, targeting the decoding problem and the weaknesses in the structures of the codes. There are recent attempts of attack on lattice structures but they are erroneous. So, as of now, there is no major attack on lattice-based structures. The National Institute of Standards and Technology (NIST), a U.S. federal agency that develops and promotes measurement standards, plays a critical role in setting cryptography standards. NIST currently prefers lattice-based cryptography to be the leading candidate for PQC. So, for everyone working in the field of PQC, lattice-based cryptography is currently the primary focus.

Aim and Motivation: This study investigates the “hard” problems of lattice-based constructions. So, what is a “hard” problem? In cryptography, a “hard” problem refers to a mathematical problem that is computationally

difficult to solve. The security of cryptographic algorithms often relies on the assumption that certain problems are infeasible to solve within a reasonable amount of time using current technology and known algorithms. These “hard” problems are essential for cryptography because their computational difficulty ensures that, without the appropriate key or secret, solving the problem (and thus breaking the encryption) is practically impossible with current computing capabilities. These are the building blocks for any cryptographic construction

The ongoing advancement in computational power, especially with the advent of quantum computing, continuously challenges the assumptions about these problems, driving the need for new cryptography techniques and standards. With this in mind, it is obvious to lay our focus on lattice-based “hard” problems, as lattice-based structures are leading candidates for PQC as we discussed before.

Synthesis of Findings: Our study contains thorough survey of the standard lattice-based assumptions and their variants, survey of literature on few under-explored areas of cryptography and how these constructions use lattice-based hard problems. At the end of this study, we will have a much clearer understanding of the bases of these assumptions and how are they used in the state-of-the-arts.

We have discussed the prerequisites of this study in the Preliminaries section. We started with understanding the standard lattice-based “hard” problems and their variants and then we continued to study their use-cases in two constructions. We have tried to explain the constructions in the simplest terms for better understanding and covered some of the computational and logical gaps of the concerned literature.

Preliminaries

We begin by introducing few notations and definitions which will be recalled later on.

Notations: We use $\{0, 1\}^m$ to denote the set of m-bit strings. We denote the set of integers by \mathbb{Z} and the set of real numbers by \mathbb{R} . The set of integers modulo p (i.e., $\{0, 1, \dots, (p-1)\}$) is denoted by \mathbb{Z}_p . We write $[n]$ to denote the set of integers $\{1, 2, \dots, n\}$. The notation $a \in A$ implies the element ‘a’ belongs to the set ‘A’. The notation a^{-1} denotes the inverse of an element, i.e. $a \times a^{-1} = 1$. Bold small letters (such as, \mathbf{a}) denote vectors and bold capital letters (such as, \mathbf{A}) denote matrices. We use $\lfloor \cdot \rfloor$, $\lceil \cdot \rceil$ and $\text{round}(\cdot)$ to denote the standard floor, ceiling and rounding to the nearest integer functions (rounding down in case of a tie).

The p-adic decomposition of an integer $x \geq 0$ is a tuple $(x_i)_{0 \leq i < \lceil \log_p(x) \rceil}$ with $0 \leq x_i < p$ such that $x = \sum_i p^i \cdot x_i$.

Random variables are used to quantify the outcomes of random processes. A probability distribution is a mathematical function that provides the probabilities of occurrence of different possible outcomes in an experiment. It describes how the probabilities are distributed over the values of the random variable. A probability mass function is a function that gives the probability value of a discrete random variable.

For a finite set S , we write $x \leftarrow S$ to denote that ‘x’ is drawn uniformly at random from S . For a probability distribution \mathcal{D} , we write $x \leftarrow \mathcal{D}$ to denote that x is drawn from \mathcal{D} .

We say that a function $f(\lambda)$ is negligible in λ if $f(\lambda) = o(1/\lambda^c)$ for all $c \in \mathbb{N}$. We write $f(\lambda) = \text{poly}(\lambda)$ to denote that f is bounded by some fixed polynomial in λ . $[o(g(n)) = \{f(n): \text{for any positive constant } c, \text{ there exists positive constant } n_0 \text{ such that } 0 \leq f(n) < cg(n) \text{ for all } n \geq n_0\}, \text{ where } n_0, c \in \mathbb{N}.]$ We say that an algorithm is efficient if it runs in probabilistic polynomial time in the length of its input. An algorithm is said to be polynomially bounded if it terminates after $\text{poly}(\lambda)$ steps and uses $\text{poly}(\lambda)$ sized memory.

The discrete Gaussian distribution over \mathbb{Z} with parameter $\sigma > 0$ has the probability mass function proportional to the Gaussian function $\rho_\sigma(x) := e^{-\pi x^2 / \sigma^2}$.

Algebraic Structures: We say ‘*’ is a binary operation if for a set A , it is defined as, $* : A \times A \rightarrow A$. We say $(G, *)$ is a group [where G is a set and ‘*’ is a binary operation] if the followings hold:

- **Closure:** For g_1 and $g_2 \in G$, $g_1 * g_2 \in G$ [Holds as ‘*’ is a binary operation.]
- **Associativity:** For g_1, g_2 and $g_3 \in G$, $g_1 * (g_2 * g_3) = (g_1 * g_2) * g_3$.

- **Identity:** There exists an element 'e' in G such that, $e * g = g * e = g$ for any $g \in G$.
- **Inverse:** For every $g \in G$ there exists some $h \in G$ such that $g * h = h * g = e$. We write $h = g^{-1}$.

We say a group is **Abelian** if the group elements are commutative, i.e., $g_1 * g_2 = g_2 * g_1$ for all $g_1, g_2 \in G$.

For examples, we can consider $(\mathbb{Z}, +)$, (\mathbb{R}, \times) , $(\mathbb{Z}_2, +)$ etc. \mathbb{Z}_n denote the set of all remainders when an integer is divided by 'n', i.e., $\{0, 1, 2, \dots, (n - 1)\}$.

Define a function ϕ between two groups $(G, *)$ and (H, \cdot) ($\phi : G \rightarrow H$). ϕ is called a **group homomorphism** if the following property holds [for all $g_1, g_2 \in G$]:

$$\phi(g_1 * g_2) = \phi(g_1) \cdot \phi(g_2)$$

ϕ is called an **group isomorphism** if it is a group homomorphism and a bijection.

We say, $(R, +, \cdot)$ is a **ring** when R is an Abelian group under '+' and closure and distributive properties (for all $a, b, c \in R$, $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(a + b) \cdot c = a \cdot c + b \cdot c$) hold for ' \cdot ' operation.

For example, $(\mathbb{Z}, +, \cdot)$ is a ring. If, R is a commutative ring with unity (multiplicative), we denote $R[X]$ to be the set of all polynomial functions ($f(x) = \sum_{i \in \{0, 1, \dots, n\}} a_i \cdot x^i$) with coefficients from the ring R . $R[X]$ with polynomial addition and multiplication, is called a **polynomial ring**.

A polynomial is **truncated** if the coefficients of the higher degree terms of the original polynomial is set to 0. A **monomial** is a polynomial with exactly one term. A function $f : A \rightarrow B$ is **linear** if $f(a_1 + a_2) = f(a_1) + f(a_2)$ for all $a_1, a_2 \in A$. **Multi-linear polynomial** is a polynomial which is linear in each of its variables separately (For example, $f(x, y) = a_0 + a_1x + a_2y + a_3xy$).

A ring R is called a **field** if every non-zero element of the ring has a multiplicative inverse within the ring. For example, we have the field of real numbers $(\mathbb{R}, +, \cdot)$.

Let R be a ring with multiplicative identity 1. A **left R-module** m consists of an Abelian group $(M, +)$ and an operation ' \cdot ' : $R \times M \rightarrow M$ such that for all $r, s \in R$ and $x, y \in M$, we have,

- $r \cdot (x + y) = r \cdot x + r \cdot y$
- $(r + s) \cdot x = r \cdot x + s \cdot x$
- $(r \cdot s) \cdot x = r \cdot (s \cdot x)$
- $1 \cdot x = x$

A **right module** is defined similarly in terms of the operation ' \cdot ' : $M \times R \rightarrow M$. An **(R,S)-bimodule** is an Abelian group together with left R -module and right S -module.

A **Toeplitz matrix** or **diagonal-constant matrix**, named after Otto Toeplitz, is a square matrix in which each descending diagonal from left to right is constant.

Example of a Toeplitz matrix:
$$\begin{bmatrix} a & b & c & d \\ e & a & b & c \\ f & e & a & b \\ g & f & e & a \end{bmatrix}$$

A Toeplitz matrix is called **circulant** if, every row consists of the exact same elements.

Example of a Circulant matrix:
$$\begin{bmatrix} a & b & c & d \\ d & a & b & c \\ c & d & a & b \\ b & c & d & a \end{bmatrix}$$

A **vector space** over a field F is a non-empty set V together with a binary operation and a binary function that satisfies the followings: [In this context, the elements of V are commonly called vectors, and the elements of F are called scalars.]

- V is an Abelian group under vector addition.
- $1 \cdot \mathbf{v} = \mathbf{v}$ for all $\mathbf{v} \in V$.
- $a \cdot (\mathbf{u} + \mathbf{v}) = a \cdot \mathbf{u} + a \cdot \mathbf{v}$ for all $\mathbf{u}, \mathbf{v} \in V$ and $a \in F$.
- $(a + b) \cdot \mathbf{v} = a \cdot \mathbf{v} + b \cdot \mathbf{v}$ for all $\mathbf{v} \in V$ and $a, b \in F$.
- $a \cdot (b \cdot \mathbf{v}) = (a \cdot b) \cdot \mathbf{v}$ for all $\mathbf{v} \in V$ and $a, b \in F$.

Given a vector space V over a field F , a norm on V is a real-valued function $\|\cdot\| : V \rightarrow \mathbb{R}$ such that:

- $\|\mathbf{v}\| \geq 0$ and $\|\mathbf{v}\| = 0 \Rightarrow \mathbf{v} = 0$ for all $\mathbf{v} \in V$
- $\|s \cdot \mathbf{v}\| = |s| \cdot \|\mathbf{v}\|$ for all $s \in F$ and $\mathbf{v} \in V$
- $\|\mathbf{u} + \mathbf{v}\| \leq \|\mathbf{u}\| + \|\mathbf{v}\|$ for all $\mathbf{u}, \mathbf{v} \in V$.

Time Complexity: In theoretical computer science, the time complexity is the computational complexity that describes the amount of computer time it takes to run an algorithm. Time complexity is commonly estimated by counting the number of elementary operations performed by the algorithm, supposing that each elementary operation takes a fixed amount of time to perform. [$O(f(n)) = g(n) \Rightarrow$ there exists a positive real M such that, $|f(n)| \leq M \cdot g(n)$ for all large enough $n \in \mathbb{N}$.]

An algorithm is said to be of **polynomial time** if its running time is upper bounded by a polynomial expression in the size of the input for the algorithm, that is, $T(n) = O(n^k)$ for some positive constant k . An algorithm is **exponential time** if $T(n)$ is bounded by $O(2^{n^k})$ for some constant k . The term

sub-exponential time is used to express that the running time of some algorithm may grow faster than any polynomial but is still significantly smaller than an exponential.

Computational Hardness: In computational complexity theory, a computational hardness assumption is the hypothesis that a particular problem cannot be solved efficiently.

An **average-case hardness** says that a specific problem is “hard” on most instances from some explicit distribution. A **worst-case hardness** only says that the problem is “hard” on some instances. So an average-case hardness assumption is stronger than a worst-case hardness assumption for the same problem.

The general class of questions that some algorithm can answer in polynomial time is “**P**”. The class of questions where an answer can be verified in polynomial time is **NP**, standing for “**nondeterministic polynomial time**”. A decision problem H is **NP-hard** when for every problem L in **NP**, there is a polynomial-time many-one reduction from L to H .

Circuit Classes: Circuit class \mathbf{AC}^0 consists of all the circuits with constant depth, polynomial-size, and unbounded fan-in gates (containing only **AND**, **OR**, and **NOT** gates). The circuit class \mathbf{TC}^0 consists of all the circuits with constant depth, polynomial-size, and unbounded fan-in and threshold gates. For any integer m , the \mathbf{MOD}_m gate outputs 1 if m divides the sum of its inputs, and 0 otherwise.

For integers $m_1, m_2, \dots, m_k > 1$, we say that a language \mathcal{L} is in $\mathbf{ACC}^0[m_1, \dots, m_k]$ if there exists a circuit family $\{C_n\}_{n \in \mathbb{N}}$ with constant depth, polynomial size, and consisting of unbounded fan-in **AND**, **OR**, **NOT**, and $\mathbf{MOD}_{m_1}, \dots, \mathbf{MOD}_{m_k}$ gates that decides \mathcal{L} . We write \mathbf{ACC}^0 to denote the class of all languages that is in $\mathbf{ACC}^0[m_1, \dots, m_k]$ for some $k \geq 0$ and integers $m_1, \dots, m_k > 0$.

Gaussian Elimination: In mathematics, **Gaussian elimination** is an algorithm for solving systems of linear equations. It consists of a sequence of row-wise operations performed on the corresponding matrix of coefficients. This method can also be used to compute the rank of a matrix, the determinant of a square matrix, and the inverse of an invertible matrix.

There are three types of elementary row operations:

- Swapping two rows.
- Multiplying a row by a nonzero number.
- Adding a multiple of one row to another row.

We say a matrix A is **upper-triangular** if $A_{ij} = 0$ for all $i > j$.

A square matrix A is **invertible** if it can be reduced to the identity matrix using elementary row operations.

To find the inverse of an invertible matrix \mathbf{A} , we reduce the matrix to identity matrix using a sequence of elementary row operations. Then we use the same sequence of elementary row operations on the identity matrix to get the inverse \mathbf{A}^{-1} .

Pseudo-Random Functions (PRF) [GGM84]: Let $\mathcal{K} = \{\mathcal{K}_\lambda\}_{\lambda \in \mathbb{N}}$, $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$, and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$ be ensembles of finite sets indexed by the security parameter λ . Let $\{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be an efficiently-computable collection of functions $\mathcal{F}_\lambda : \mathcal{K}_\lambda \times \mathcal{X}_\lambda \rightarrow \mathcal{Y}_\lambda$. Then we say that the function family $\{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ is a (t, ε) -strong PRF if for all adversaries \mathcal{A} running in time $t(\lambda)$ and taking $k \leftarrow \$ \mathcal{K}_\lambda$ and $f_\lambda \leftarrow \$ \text{Func}[\mathcal{X}_\lambda, \mathcal{Y}_\lambda]$, we have that

$$|Pr[\mathcal{A}^{F_\lambda(k, \cdot)}(1^\lambda) = 1] - Pr[\mathcal{A}^{f_\lambda(\cdot)}(1^\lambda) = 1]| \leq \varepsilon(\lambda)$$

We say that the function family is an (l, t, ε) -weak PRF if for all adversaries \mathcal{A} running in time $t(\lambda)$ and taking $k \leftarrow \$ \mathcal{K}_\lambda$ and $f_\lambda \leftarrow \$ \text{Func}[\mathcal{X}_\lambda, \mathcal{Y}_\lambda]$, $x_1, x_2, \dots, x_l \leftarrow \$ \mathcal{X}_\lambda$ we have that,

$$|Pr[\mathcal{A}(1^\lambda, \{(x_i, F_\lambda(k, x_i))\}_{i \in [l]})] - Pr[\mathcal{A}(1^\lambda, \{(x_i, f_\lambda(x_i))\}_{i \in [l]})]| \leq \varepsilon(\lambda)$$

We say that a weak PRF is **exponentially secure** if the distinguishing advantage of any adversary of size 2^λ is bounded by $2^{-\Omega(\lambda)}$ [$\Omega(f(n)) = g(n) \Rightarrow g(n) = O(f(n))$].

Partially Oblivious Pseudo-Random Functions (POPRF): A partially oblivious PRF \mathcal{F} is a tuple of PPT (probabilistic polynomial time) algorithms

$$(\mathcal{F}.Setup, \mathcal{F}.KeyGen, \mathcal{F}.Request, \mathcal{F}.BlindEval, \mathcal{F}.Finalise, \mathcal{F}.Eval)$$

The setup and key generation algorithm generate public parameter ‘pp’ and a public/secret key pair (pk,sk). Oblivious evaluation is carried out as an interactive protocol between client (C) and server (S), here presented as algorithms $\mathcal{F}.Request, \mathcal{F}.BlindEval, \mathcal{F}.Finalise$ working as follows:

- ‘First, C runs the algorithm $\mathcal{F}.Request_{pp}(pk, t, x)$ taking a public key pk, a tag or public input t and one private input x. It outputs a local state st and a request message req, which is sent to the server.
- S runs $\mathcal{F}.BlindEval_{pp}(sk, t, req)$ taking as input a secret key sk, a tag t, and the request message req. It produces a response message rep, sent back to C.
- Finally, C runs $\mathcal{F}.Finalise(rep, st)$ which takes the response message and its previously constructed state st and outputs a PRF evaluation or \perp if rep is rejected.

The unblinded evaluation algorithm $\mathcal{F}.Eval$ is deterministic and takes as input a secret key sk, an input pair (t,x) and outputs a PRF evaluation z.

Fixing the tag t, e.g. $t = \perp$, recovers the definition of an **OPRF**.

Now, let’s talk about some basic concepts of lattices.

Lattice

Definition: An n -dimensional lattice \mathcal{L} is any subset of \mathbb{R}^n that satisfies the following conditions:

- It is an additive subgroup of \mathbb{R}^n .
- Every $x \in \mathcal{L}$ has a neighbourhood in \mathbb{R}^n in which x is the only lattice point (Discreteness).

Examples:

- Integer lattice: (\mathbb{Z}^n) .
- Checkerboard lattice: $\{x \in \mathbb{Z}^n : \sum_i x_i \text{ is even}\}$

The minimum distance of a lattice \mathcal{L} is the length of a shortest non-zero lattice vector $\mathbf{v} \in \mathcal{L}$:

$$\lambda_1(\mathcal{L}) := \min_{v \in \mathcal{L}, v \neq 0} \|v\|$$

Base of a lattice: Although every (non-trivial) lattice is infinite, it is always finitely generated as the integer linear combinations of some linearly independent vectors $B = \{b_1, b_2, \dots, b_k\}$:

$$\mathcal{L}(B) := B \cdot \mathbb{Z}^k = \left\{ \sum_{i=1}^k z_i b_i : z_i \in \mathbb{Z} \right\}$$

The set B is called the basis of the lattice.

$$\bullet \begin{bmatrix} \vec{b}_1 & \vec{b}_2 & \dots & \vec{b}_m \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix} = B; \text{ [This is an } (n \times m) \text{ matrix.]}$$

$L(B)$ is called a full rank lattice when $m=n$.

- We say that two bases (B_1, B_2) are **equivalent** if and only if we have $L(B_1) = L(B_2)$.
- A “good” basis is expected to consist of short and almost orthogonal vectors (perpendicular to each other). A basis with very long vectors or non-orthogonal vectors is considered to be a “bad” basis.

Now, as we have learned about the basics of lattice, let’s discuss some traditional “hard” problems within the lattice paradigm.

Lattice-based “Hard” Problems

“Hard” problems are the building blocks for any cryptography approach. In this section, we will elaborately discuss about some well-known lattice-based “hard” problems and their variants. [Note: We will assume the norm to be Euclidean norm if not mentioned otherwise.]

4.1 Shortest Vector Problem (SVP)

The **Shortest Vector Problem** asks to find the shortest length vector of a given lattice. There are some variants of this problem, which are discussed below.

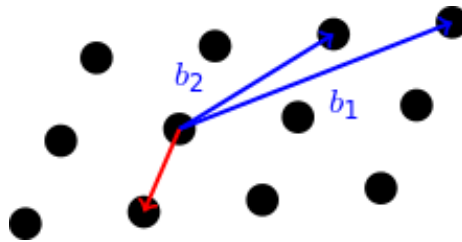


Figure 1: Standard SVP (Blue lines indicate basis vectors and the red line indicates the shortest vector from the origin, [Assuming the initial point to be the origin])

- **Standard SVP:** Given a basis B (typically a bad basis) of an n -dimensional lattice \mathcal{L} , find the shortest non-zero vector in \mathcal{L} , i.e., $\|\mathbf{v}\| = \lambda(\mathcal{L})$.

[Remember that $\lambda(\mathcal{L})$ = length of the shortest non-zero vector in the lattice.]

As, the origin always belongs to any lattice, the problem basically is to find the nearest lattice point from the origin.

- **Approximate SVP:** In the γ -approximation version, we have to find a non-zero vector in \mathcal{L} such that $\|\mathbf{v}\| \leq \gamma \cdot \lambda(\mathcal{L})$ where $\gamma \geq 1$.

If γ is constant, this problem becomes NP-hard. If γ is exponential, then it is an easy problem. If γ is sub-exponential or polynomial, the problem shows average case hardness.

- **Unique SVP (uSVP):** In the standard SVP, the shortest vector may not be unique, and there can be multiple vectors of similar length. Formally, given a lattice \mathcal{L} , and a real number γ (gap parameter), the problem is to

find the shortest vector \mathbf{v} in \mathcal{L} under the condition that any other non-zero vector \mathbf{w} in \mathcal{L} satisfies $\|\mathbf{w}\| \geq \gamma\|\mathbf{v}\|$.

- **Gap-SVP:** The problem Gap-SVP is defined as a decision problem where given a lattice \mathcal{L} , a positive real number d , and a gap factor γ , the problem is to decide whether $\lambda(\mathcal{L}) \leq d$ or $\lambda(\mathcal{L}) > \gamma \cdot d$
- **Shortest Independent Vectors Problem (SIVP):** In SIVP, the goal is to find a set of linearly independent lattice vectors that are collectively short. Formally, given a lattice \mathcal{L} and a dimension n , find n linearly independent vectors v_1, v_2, \dots, v_n in \mathcal{L} such that the length of the longest vector in the set is minimized.
- **Shortest Basis Problem (SBP):** Recall the definitions of good basis and bad basis for a lattice. This problem asks us to provide a good basis for a given bad basis of a lattice.

4.2 Closest Vector Problem (CVP)

The **Closest Vector Problem** asks to find the closest vector on the given lattice from a given point. This is a generalization of the SVP. There are some variants of this problem, which are discussed below.

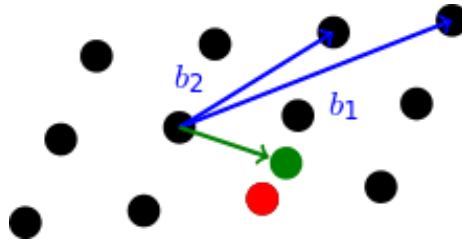


Figure 2: Standard CVP (Blue lines indicate the basis vectors, the green point is the given point \mathbf{v} on \mathbb{R}^n [$n=2$ here], and the red point is the closest lattice point from \mathbf{v} [Assuming the initial point to be the origin]).

- **Standard CVP:** Given a basis B (typically a bad basis) of an n -dimensional lattice \mathcal{L} , as well as a vector \mathbf{v} in \mathbf{R}^n but not necessarily in \mathcal{L} , the problem is to find a vector in \mathcal{L} closest to \mathbf{v} .

If we shift the origin to the vector \mathbf{v} , this problem becomes the standard SVP.

- **Approximate Closest Vector Problem (Approx-CVP):** In the γ -approximation version, the problem is to find a lattice vector at a distance at most γ .

- **Bounded Distance Decoding (BDD) Problem:** This variant is a special case of Approx-CVP where the target vector \mathbf{t} is guaranteed to be within a certain distance δ of the lattice. The problem is typically easier than the general CVP when δ is small relative to the shortest vector in the lattice.
- **Inhomogeneous Closest Vector Problem (ICVP):** This variant considers a modified lattice problem where the goal is to find the closest vector to a given target when the lattice is shifted by a fixed vector. Formally, given a lattice \mathcal{L} , a shift vector \mathbf{u} , and a target vector \mathbf{t} , find a lattice vector $\mathbf{v} \in \mathcal{L}$ such that, $\|(\mathbf{v} + \mathbf{u}) - \mathbf{t}\|$ is minimized.
- **Preimage CVP (pCVP):** The goal is to find a lattice vector \mathbf{v} such that when a given linear transformation is applied to \mathbf{v} , the result is as close as possible to a target vector. Formally, given a lattice \mathcal{L} , a linear transformation matrix \mathbf{A} , and a target vector \mathbf{t} , find a lattice vector $\mathbf{v} \in \mathcal{L}$ such that, $\|(\mathbf{A} \cdot \mathbf{v} - \mathbf{t})\|$ is minimized.

There are other variants as well. The standard problems and all the variants are computationally difficult, especially in high-dimensional lattices.

4.3 Learning With Error (LWE)

LWE is one of the most important lattice-based hard problems. A very important work of Regev from 2005 introduced the average case LWE problem for the first time. The problem involves solving a system of linear equations that have been perturbed by small random errors.

Take a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$, and a small error \mathbf{e} randomly from a discrete Gaussian distribution χ . Compute $\mathbf{b} = [\mathbf{A}] \cdot [\mathbf{s}] + \mathbf{e}$. Now, for known \mathbf{A} and \mathbf{b} , it is a “hard” problem to find \mathbf{s} , known as **LWE**.

This is also known as search-LWE. There is a decisional version of the LWE problem which is defined as follows:

Given the parameters similar to the search-LWE problem, it is difficult to distinguish between $(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$ and (\mathbf{A}, \mathbf{u}) where \mathbf{u} is taken uniformly at random from \mathbb{Z}_q^m .

Let’s discuss about some of the variants of the LWE problem.

- **Ring-LWE:** Ring-LWE is a variant of LWE where the same problem is defined over polynomial rings rather than integer matrices. Lyubashevsky, Peikart and Regev introduced RIng-LWE in a work published in 2010.
- **Module-LWE [LS15]:** Module-LWE generalizes LWE by considering module structures over rings. It provides a trade-off between the efficiency of RLWE and the security of standard LWE.

$$\mathbf{A} \mathbf{s} + e \pmod{p} \approx \mathbf{u}$$

Figure 3: Decision LWE

- **Learning With Rounding (LWR):** LWR is a variant of LWE where the noise is introduced through a rounding function instead of additive noise. The learning with rounding (LWR) problem was introduced by Banerjee et al in 2012 [BPR12].

Take a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$, and \mathbf{u} where \mathbf{u} is taken uniformly at random from \mathbb{Z}_q^m . It is difficult to computationally distinguish between $(\mathbf{A}, r(\mathbf{A}\mathbf{s}))$ and $(\mathbf{A}, r(\mathbf{u}))$, where ‘r’ is the rounding function. This is the LWR problem.

- **Learning Parity with Noise (LPN):** In 1996, Oded Goldreich, Shafi Goldwasser, and Dana Ron introduced the LPN assumption in their work. LPN is almost similar to the LWE problem with similar problem structure, but there are some key differences.

LPN is defined over binary fields (mod 2), whereas LWE is typically defined over larger fields or rings.

In LPN, noise is added directly to the linear equations in a binary field, while LWE generally involves Gaussian or uniform noise in larger domains.

LWE is a generalization of LPN.

Hardness: The obvious question that comes to mind is, “How is the LWE problem related to lattice constructions?” Even if there is apparently no direct link between LWE and lattice-based hard problems, the LWE problem can be shown to be as “hard” as worst-case lattice problem, using reduction. In 2005, Regev showed that the decision version of the LWE is hard assuming the quantum hardness of the lattice problems **GapSVP** and **SIVP**. In 2009, Peikert proved a similar result assuming only the classical hardness of the related problem **GapSVP**.

Worst-case hardness is not really helpful for cryptography. The beauty of LWE lies in the fact that LWE generates average-case hardness from the worst-case hardness of GapSVP.

We can get many LWE-hard problems with different error distributions, but only a discrete Gaussian distribution of errors relate the LWE problem to lattice problems for certain parameters.

4.4 Shortest Integer Solution (SIS)

The **SIS** problem was first introduced in the seminal work of Ajtai [Ajt96]. This is one of the main average-case hard problem from the lattice domain. Let’s discuss the problem and few of its variants.

- **Standard SIS:** Given m uniformly random vectors forming the columns of a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, and positive integers n, q , find a non zero vector $\mathbf{z} \in \mathbb{Z}^m$ with $\|\mathbf{z}\| \leq \beta$ (a given real number) such that, $f_{\mathbf{A}}(\mathbf{z}) := \mathbf{A} \cdot \mathbf{z} = \mathbf{0} \in \mathbb{Z}_q^n$.

The decision version of the problem is to distinguish between a valid short integer solution and a random vector.

Notice that, finding the vector \mathbf{z} with any constraint on $\|\mathbf{z}\|$ is easy via Gaussian elimination method. However, the problem becomes “hard” when the vector \mathbf{z} needs to be “short”.

Similarly, we must take $\beta < q$, otherwise $\mathbf{z} = (q, 0, \dots, 0) \in \mathbb{Z}^m$ will always be a legitimate (but trivial) solution. Also, β must be large enough such that \mathbf{z} exists.

Assume that we are given $\mathbf{y} = \mathbf{A} \cdot \mathbf{z}$ where \mathbf{A}, \mathbf{z} are taken uniformly at random from $\mathbb{Z}_q^{m \times n}$ and $\{0, 1\}^m$ respectively.

Now, notice that the maximum possible number of $\mathbf{y} = q^n$ and if $m \geq \lfloor n \cdot \log q \rfloor + 1$ and $\beta \geq \sqrt{\lfloor n \cdot \log q \rfloor + 1}$ then number of $\mathbf{z} > q^n$, as $\|\mathbf{z}\| < \beta \leq \sqrt{m}$ (equality holds when every coordinate is 1).

So, using the Pigeonhole Principle, there exist $\mathbf{z}_i \neq \mathbf{z}_j$ such that $y_{z_i} = y_{z_j}$, i.e., $A(\mathbf{z}_i - \mathbf{z}_j) = 0$.

Hence, there is at least one solution for this lower bound of β .

- **Inhomogeneous SIS (ISIS):** It is a more generalized version of the SIS problem.

Given $\mathbf{Y} \in \mathbb{Z}_q^n$ and $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, find \mathbf{X} such that $\mathbf{A} \cdot \mathbf{X} = \mathbf{Y}$ with $\|\mathbf{X}\| \leq \beta$ and $\mathbf{X} \neq 0$.

ISIS is “hard”, assuming SIS is “hard”.

- **Ring-SIS:** Ring-SIS is a variant of SIS defined over polynomial rings rather than integer matrices, similar to the Ring-LWE problem.
- **Module-SIS [LS15]:** Module-SIS generalizes SIS by considering module structures over rings. It provides a trade-off between the efficiency of Ring-SIS and the security of standard SIS.

Hardness: A seminal work by Miklós Ajtai showed that the SIS problem is “hard” in an average case if the approximate shortest vector problem (approx-SVP) is “hard” in a worst-case scenario.

4.5 NTRU Assumption:

The assumption relies on the presumed difficulty of factoring certain polynomials in a truncated polynomial ring into a quotient of two polynomials having very small coefficients.

Formally, given integers n, p, q, β , where p and q are coprime, the matrix-NTRU assumption states that it is computationally hard to distinguish between \mathbf{A} and \mathbf{B} where,

- \mathbf{A} is chosen uniformly at random from $\mathbb{Z}_q^{n \times n}$
- $\mathbf{B} = p^{-1} \cdot \mathbf{G}^{-1} \cdot \mathbf{F} \pmod{q}$ with, \mathbf{F} chosen uniformly at random from $\{0, \pm 1, \dots, \pm \beta\}^{n \times n}$ and \mathbf{G} chosen uniformly at random from $\{0, \pm 1, \dots, \pm \beta\}^{n \times n} \cap (\mathbb{Z}_q^{n \times n})^*$, where $(\mathbb{Z}_q^{n \times n})^*$ denotes the set of invertible $(n \times n)$ matrices over \mathbb{Z}_q .

The original formulation [HPS96] of the NTRU problem considers rings of integers of number fields or polynomial rings instead of integer matrices.

Hardness: The hardness of the NTRU assumption relies on the hardness of the SVP in a specific type of lattice constructed from truncated polynomials over a ring. As SVP is computationally hard to solve, this connection underscores its resilience against both current and future cryptographic attacks, ensuring its relevance in the field of secure communications and data protection.

Now that we have studied the traditional lattice-based “hard” problems, we can introduce some under-explored cryptographic constructions and see how these problems are related to these, serving as building blocks.

Lattice and Crypto Dark Matter

The term “**crypto dark matter**” describes cryptographic techniques or algorithms that have not been extensively studied or widely recognised outside of niche communities or academic circles. Little-known cryptographic primitives, cutting-edge encryption techniques, or obscure cryptographic protocols that rarely get utilised or addressed in mainstream cryptography are a few examples of these. Though they are relatively obscure or not popular, the phrase “dark matter” is used metaphorically to indicate their presence and potential significance.

These crypto dark matter components are being developed actively or are waiting to be adopted more widely while their efficiency and security are further examined. Although they might have special qualities or be superior to traditional cryptography techniques, they usually need to undergo a thorough evaluation by experts before being widely accepted for usage in practical applications.

In this section, we will discuss a couple of “crypto dark matter” constructions and how they have used the “hard” lattice problems. Our study includes a weak pseudo-random function (Weak-PRF) and an oblivious pseudo-random function (OPRF) construction.

What is a **pseudo-random function (PRF)** [GGM86](#)? It is a function that is indistinguishable from a truly random function by any efficient adversary. Formally, a function $\mathbf{F} : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^m$ is a PRF if, for a randomly chosen key k , the function $\mathbf{F}_k(x)$ [x is the input] is computationally indistinguishable from a truly random function with the same domain and range.

A **weak PRF** has a similar goal as a PRF but with a weaker security guarantee. Specifically, a weak PRF is computationally indistinguishable from a truly random function when queried with inputs chosen from a limited set.

A **Strong PRF** provides a stronger security guarantee than a standard PRF. In addition to being indistinguishable from a random function, a Strong PRF maintains this property even if the adversary is given oracle access to both the function and its inverse.

An **oblivious PRF (OPRF)** is a special type of pseudo-random function that allows one party (user) to evaluate the PRF on an input without learning anything about the PRF key, while the other party (server, holding the key) does not learn the input or the output. If the user can be convinced that the output is correct (i.e. the evaluation is performed under the correct key) then the function is “verifiable oblivious” or **VOPRF**.

Sometimes, it is sufficient or even necessary to only hide a part of the user input. In this case, the public and private inputs are separated by requiring an additional public input “ t ”, called the **tag**. We write, $PRF(k, x, t) := \mathbf{F}_k(t; x)$, where the input part “ x ” is hidden from the server. This function is called a **partially oblivious PRF (POPRF)**.

An **encoded-input PRF (EI-PRF)** is similar to a strong PRF. The main difference is that its input domain is restricted to an efficiently recognizable set. EI-PRFs can be viewed as an intermediate between strong and weak PRFs that combine the security advantages of the former and efficiency advantages of the later. [NR95]

As we have explained the general ideas of different types of PRFs, let’s see how we can exploit the lattice-based hard problems to construct PRFs. The classic LPN and LWE assumptions are natural starting points for building PRFs. However, the main problem of using the traditional “learning-with-noise” assumptions to construct deterministic PRFs is finding a way to introduce errors into exponentially many function outputs of the PRF, while also keeping a polynomial key-size. Although constructing randomized weak PRFs, is possible directly from LWE (Shown by Applebaum et al. [ACPS09]). Lattice-based constructions of PRFs have thus relied on the “derandomized” variant of LWE, i.e., LWR. [BPR12, BLMR13, BP14].

The approach of any cryptographic construction should focus on greater efficiency. The following literature, “**Exploring Crypto Dark Matter: New Simple PRF Candidates and Their Applications**” [BIP⁺18], departs from traditional approaches, and circumvents some of the limitations of these approaches in search of greater efficiency. This work claims that the proposed main weak PRF candidate has a structural similarity with the lattice-based “hard” problem LPN, but the connection of this PRF candidate with LPN is unclear. An alternative weak PRF candidate is also proposed which corresponds to the LWR problem. So, here we introduce the first literature and try to describe the work in simple terms with attempt to cover any gaps in computation.

5.1 “Exploring Crypto Dark Matter: New Simple PRF Candidates and Their Application”

5.1.1 Aim

The aim is to rely on assumptions that are simple to state and yet, breaking them would likely require new techniques that may have other applications on their own. This work explores whether the idea of mixing linear functions over different moduli can be a source of hardness. In this way, they have tried to maximize the simplicity of their candidate PRF and minimize the complexity.

5.1.2 Candidate PRFs

This new weak PRF candidate has a lot of structural similarities with the weak PRF candidate previously proposed by Akavia et al. [\[ABG⁺14\]](#).

Let, λ be a security parameter, and $m = m(\lambda)$ and $n = n(\lambda)$. The weak PRF candidate is a function $F_\lambda : \mathbb{Z}_2^{m \times n} \times \mathbb{Z}_2^n \rightarrow \mathbb{Z}_3$ where $\mathbb{Z}_2^{m \times n}$ is the key space, \mathbb{Z}_2^n is the domain and \mathbb{Z}_3 is the output space. For a key \mathbf{A} , we write $F_{\mathbf{A}}(x)$ to denote the function $F_\lambda(\mathbf{A}, x)$. Define $F_{\mathbf{A}}(x)$ as follows:

- On input $x \in \mathbb{Z}_2^n$, compute $y' = \mathbf{A}x \in \mathbb{Z}_2^m$.
- Now, $F_{\mathbf{A}}(x) := \text{map}(\mathbf{A}x) = \text{map}(y')$ where $\text{map} : \{0, 1\}^m \rightarrow \mathbb{Z}_3$ is defined as, $\text{map}(y) := \sum_{i \in [m]} y_i \pmod{3}$

[Note that the matrix vector product $\mathbf{A}x$ is computed over \mathbb{Z}_2 , and then the values are re-interpreted as integers before adding them over \mathbb{Z}_3 .]

This can be generalized by replacing 2 and 3 with primes ‘p’ and ‘q’ respectively with $p \neq q$. The key space will be $\mathbb{Z}_p^{m \times n}$, the domain will be \mathbb{Z}_p^n and the output space will be \mathbb{Z}_q . The map function will transform into $\text{map} : \{0, 1, \dots, (p-1)\}^m \rightarrow \mathbb{Z}_q$ such that, $\text{map}_{p,q}(y) := \sum_{i \in [m]} y_i \pmod{q}$. Finally, we can define, $F_{\mathbf{A}}(x) := \text{map}_{p,q}(\mathbf{A}x)$.

[Note that for certain choices of p,q, the output might not be balanced and pseudo-randomness is then defined with respect to the corresponding input distribution.]

In many situations, PRFs with longer output are required. In those cases, we can simply take the vector $\mathbf{A}x \in \mathbb{Z}_2^m$, re-interpret that as vector $y' \in \mathbb{Z}_3^m$ and output $\mathbf{G}y' \in \mathbb{Z}_3^t$ where $\mathbf{G} \in \mathbb{Z}_3^{t \times m}$ is a fixed public matrix.

Now, let’s have a look at the alternative weak PRF candidate:

Let, λ be a security parameter, and $n = n(\lambda)$ be the key length (and input length). The weak PRF candidate is a function $F_\lambda : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \mathbb{Z}_2$ where $\{0, 1\}^n$ is the key space and the domain and \mathbb{Z}_2 is the output space. For a key \mathbf{k} , we write $F_{\mathbf{k}}(x)$ to denote the function $F_\lambda(\mathbf{k}, x)$. Define $F_{\mathbf{k}}(x)$ as follows: On input $x \in \{0, 1\}^n$,

$$F_{\mathbf{k}}(x) = [\sum_{i \in [n]} k_i x_i \pmod{2} + \sum_{i \in [n]} k_i x_i \pmod{3}] \pmod{2}$$

5.1.3 Advantages

1. The efficiency can be improved by taking the key to be a structured matrix rather than any random matrix. For example, if we take the matrix to be a uniformly random Toeplitz matrix, then the size of the PRF key is reduced from “mn” to “m+n”. A similar optimization for using a random Toeplitz matrix in place of a random matrix was previously proposed to reduce the key size in authentication schemes based on the learning parity with noise (LPN) problem [[GRS08](#), [Pie12](#)]. But, it has been shown

that when \mathbf{A} is chosen to be a circulant matrix, the security of the scheme is degraded.

2. This PRF candidate is the first one that can be computed by an ACC^0 circuit and plausibly satisfy exponential security. This is in fact computable by a depth-2 ACC^0 circuit, which is the minimal depth for any existing PRF candidate.
3. This main construction is very MPC (Multi-Party Computation)-friendly. [Multi-Party Computation (MPC) is a cryptographic protocol that allows multiple parties to jointly compute a function over their inputs while keeping those inputs private. The key goal of MPC is to enable collaborative computation without revealing individual inputs to any party.]

5.1.4 Hardness

- First of all, we will discuss the hardness of the proposed alternative weak PRF candidate. This candidate is closely related to traditional lattice-based “hard” problems. Let’s have a look:

$$F_{\mathbf{k}}(x) = [\sum_{i \in [n]} k_i x_i \pmod{2} + \sum_{i \in [n]} k_i x_i \pmod{3}] \pmod{2}$$

This can be thought of as an LPN instance with noise rate $(1/3)$, except the noise is generated using a deterministic, key-dependent, and input-dependent computation (i.e., noise is added to exactly one out of three coordinates of the output vector).

The noise is 1 if and only if $\sum_{i \in [n]} k_i x_i = 1 \pmod{3}$.

Again, note that $F_{\mathbf{k}}(x) = 1$ if and only if $\sum_{i \in [n]} k_i x_i \pmod{6} \in \{3, 4, 5\}$

It is easy to check that,

If $\sum_{i \in [n]} k_i x_i \pmod{6} \in \{0, 1, 2\}$; $F_{\mathbf{k}}(x) = 0$ and,

If $\sum_{i \in [n]} k_i x_i \pmod{6} \in \{3, 4, 5\}$; $F_{\mathbf{k}}(x) = 1$

[For example, we can see,

$$\sum_{i \in [n]} k_i x_i \pmod{6} = 0$$

$$\Rightarrow \sum_{i \in [n]} k_i x_i \pmod{2} = 0 \text{ and } \sum_{i \in [n]} k_i x_i \pmod{3} = 0$$

$$\Rightarrow F_{\mathbf{k}}(x) = 0]$$

Now, we can express the PRF function as, $F_{\mathbf{k}}(x) = \lfloor \cdot \rfloor_2$, where $\lfloor \cdot \rfloor_2 : \mathbb{Z}_6 \rightarrow \mathbb{Z}_2$ is the rounding operator, and we view the key \mathbf{k} and input \mathbf{x} as binary vectors over \mathbb{Z}_6 .

We can compare this with the definition of **LWR**. Take a matrix $\mathbf{k} \in \mathbb{Z}_6^{1 \times n}$, a secret vector $\mathbf{x} \in \mathbb{Z}_6^n$, and \mathbf{u} where \mathbf{u} is taken uniformly at random from \mathbb{Z}_6 . If it is difficult to computationally distinguish between $(\mathbf{k}, \lfloor \mathbf{k} \cdot \mathbf{x} \rfloor_2)$ and $(\mathbf{k}, \lfloor \mathbf{u} \rfloor_2)$, where ‘ $\lfloor \cdot \rfloor_2$ ’ is the rounding function, this follows the definition of a PRF.

Multiple works have studied the hardness of LWR in several parameter settings [BPR12, AKPW13, BGM⁺16, ASA16]. Although, it is important to mention that the known reductions from LWR to worst-case lattice problems, do not apply in the constant-size composite modulus setting.

- Now let's discuss the hardness of the main weak PRF candidate. The hardness lies in the observation that, the sum of m -binary valued variables modulo 3 is actually a high-degree polynomial over \mathbb{Z}_2 . The beauty of this idea is that it is simple to describe, potentially achieves strong security guarantees and minimizes complexity measures relevant to natural cryptographic applications.

Some formal conjectures regarding the hardness of the main weak PRF candidate, are provided in this literature.

1. Let λ be a security parameter. Then, there exist m, n (polynomial in security parameter λ) such that for all l, t (polynomial in security parameter λ), any adversary running in time $t(\lambda)$ and taking $l(\lambda)$ queries, can not computationally distinguish between the main weak PRF candidate and an random function uniformly drawn from the function space $F[\mathbb{Z}_2^n, \mathbb{Z}_3]$.
2. The main weak PRF candidate also satisfies exponential hardness.

This literature has also established a connection between the above-mentioned conjectures and the **hardness of interpolating sparse multivariate polynomials over \mathbb{Z}_3** . Even though the reduction is correct, We believe a few queries are there to be elaborated.

The literature also suggests that, there exists a structural similarity of this main weak PRF candidate with the **LPN** problem. We found no further discussion on this claim.

Our work includes an elaborate explanation of the above-mentioned reduction steps and find the structural similarity between the LPN problem and the main weak PRF candidate.

5.1.5 Hardness Reduction (Elaborated)

(A) Hardness of Interpolating Sparse Polynomials over \mathbb{Z}_3 :

Let's see how the behavior of the main weak PRF candidate corresponds to evaluating a sparse multi-linear polynomial over \mathbb{Z}_3 .

Take a matrix $\mathbf{A} \in \mathbb{Z}_2^{m \times n}$ and an input $x \in \mathbb{Z}_2^n$.

$x = (x_1, x_2, \dots, x_n)$, i.e., $x_i \in \mathbb{Z}_2$ for $i \in \{1, 2, \dots, n\}$

Now, consider the function,

$\phi : \mathbb{Z}_2 \rightarrow (\{-1, +1\}, \cdot)$ [Note that $(\{-1, +1\}, \cdot)$ is a group]

such that, $\phi(0) = 1$ and $\phi(1) = -1$

ϕ satisfies the condition $\phi(a + b) = \phi(a) \cdot \phi(b)$ [Group homomorphism]

We can extend this as, $\phi(m \times a + n \times b) = \phi(m \times a) \cdot \phi(n \times b) = \phi(a)^m \cdot \phi(b)^n$
[Property of group homomorphism]

Note that, in the modulo 3 setting, $\phi(a) = 1 + a \pmod{3}$ [$2 \equiv -1 \pmod{3}$]
 $\Rightarrow a = \phi(a) - 1 \pmod{3}$

Now, $F_{\mathbf{A}}(x) = \text{map}(\mathbf{A}x) = \sum_{i \in [m]} \mathbf{A}_i x \pmod{3}$

$= \sum_{i \in [m]} [\sum_{j \in [n]} A_{ij} x_j] \pmod{3}$

$= \sum_{i \in [m]} [\prod_{j \in [n]} \phi(x_j)^{A_{ij}} - 1] \in \mathbb{Z}_3$ [Group homomorphism property]

Hence, $F_{\mathbf{A}}(x) = \text{map}(\mathbf{A}x) = \text{map}(\phi(\mathbf{A}x) - 1) \pmod{3}$

[Basically, $\phi(\mathbf{A}x) - 1$ is the identity function in this context.]

So, using this representation we can say, for every choice of the key \mathbf{A} , the PRF $F_{\mathbf{A}}$ implements a sparse multi-linear polynomial over \mathbb{Z}_3 with n variables of degree at most n and containing at most m non-zero monomials. A multi-linear polynomial over n variables can have up to 2^n monomials. So, this weak PRF candidate is very sparse, as it contains at most m (polynomial of n) monomials.

Hence, hardness conjectures of this weak PRF implies that it should be hard to approximate sparse multi-linear polynomials over \mathbb{Z}_3 given random evaluations on inputs drawn uniformly at random from $\{\pm 1\}^n$.

Numerous works have studied the problem of interpolating sparse polynomials over both finite fields [Wer94, [GS09, AGR14]] and over fields of characteristic zero [Zip79, BOT88, KY88, Zip90]. However, existing algorithms rely on making structured, and often adaptively-chosen, queries to the underlying polynomial. The existing algorithms do not generalize to the setting where they only have access to random evaluations of the polynomial over a restricted subset of the domain. In fact, these conjectures imply an even stronger requirement: it should be difficult to test whether a particular function can be represented by a sparse polynomial.

(B) Structural Similarity with LPN:

First of all, let's state the formal definition of "Approximate Polynomial Interpolation".

Definition: Let \mathbb{F} be a finite field, and fix parameters m , n , and d . Let $\mathcal{F} \subseteq \mathbb{F}[x_1, x_2, \dots, x_n]$ be a family of polynomials over n variables of degree at most d and containing at most m non-zero coefficients. We say that \mathcal{F} can be efficiently approximated given evaluations from a distribution \mathcal{D} if there exists an algorithm \mathcal{A} such that for every $f \in \mathcal{F}$ and every $0 < \varepsilon < 1/2$, $0 < \delta < 1$, if we set $g \leftarrow \mathcal{A}(\varepsilon, \delta, \{x_i, f(x_i)\}_{i \in [N]})$ for some N (polynomial in $n, m, d, 1/\varepsilon, 1/\delta$) with probability at least $(1 - \delta)$, the function g is ε -close to f , and moreover, the running time of \mathcal{A} is bounded by some polynomial in $(n, m, d, 1/\varepsilon, 1/\delta)$.

The hardness of interpolating sparse multi-linear polynomial over \mathbb{Z}_3 implies that, finding such approximation function g , will be “hard”.

i.e., $\|f - g\| < \varepsilon$ [The norm is the corresponding norm of the function space].

we are able to rule out low-degree polynomial approximations by appealing to the classic Razborov-Smolensky lower bounds for ACC^0 [Raz87, Smo87], which essentially says that for distinct primes p and q , MOD_p gates cannot be computed in $ACC^0[q^l]$ for any $l \geq 1$.

Now, recall the definition of LPN (similar to LWE, on binary fields).

It is hard to find \mathbf{s} , given \mathbf{A} and $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$

or, we can say it is hard to find $\mathbf{A} \cdot \mathbf{s}$ from \mathbf{A} and \mathbf{b} , as, if we can get $\mathbf{A} \cdot \mathbf{s}$, it is easy to find \mathbf{s} using Gaussian elimination method.

Now, say $f_{\mathbf{A}} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ be a function. We need to approximate $\mathbf{A} \cdot \mathbf{s}$ by using evaluations drawn uniformly at random from \mathbb{Z}_2^n . If, $g_{\mathbf{A}}$ is our approximation function, then we can say that, for any $0 < \varepsilon < 1/2$, $0 < \delta < 1$, $\|f_{\mathbf{A}} - g_{\mathbf{A}}\| < \|\mathbf{e}\|$, i.e. $\|f_{\mathbf{A}} - g_{\mathbf{A}}\| < \varepsilon$, with probability $(1 - \delta)$. [Assuming, $\|\mathbf{e}\| < \varepsilon$]

The hardness of the LPN problem implies that, this approximation will be “hard” as well.

We have stated the LPN problem as a learning problem, similar to how we stated the hardness of the main weak PRF. We have reduced the hardness of the main PRF as, it should be hard to approximate sparse multi-linear polynomials over \mathbb{Z}_3 given random evaluations on inputs drawn uniformly at random from $\{\pm 1\}^n$. Now, it is easy to see that \mathbb{Z}_2 and $(\{\pm 1\}, \cdot)$ are isomorphic, and hence, \mathbb{Z}_2^n and $(\{\pm 1\}, \cdot)^n$ are also isomorphic [Component-wise]. Hence, the inputs for evaluations can be thought of as elements of \mathbb{Z}_2^n .

So, following the definition of “Approximate Polynomial Interpolation” and by restating the LPN problem, we can see that these two are structurally similar to each other.

We will now introduce another literature “**Crypto Dark Matter on the Torus, Oblivious PRFs from shallow PRFs and TFHE**” [ADDG24], that constructs an POPRF using this main weak PRF candidate.

5.2 “Crypto Dark Matter on the Torus”

This literature can be viewed as a continuation of the previous one. An OPRF construction from the previous main weak PRF candidate will be discussed here. Before we visit the construction, there are some modern lattice-based cryptographic techniques we should know.

Fully Homomorphic Encryption (FHE): FHE allows to perform computations on plain-texts by performing operations on ciphertexts. This allows us to perform multiple computations on encrypted data. This scheme was introduced by Craig Gentry in 2009 [Gen09].

An FHE scheme consists of four algorithms: key generation ($FHE.KeyGen$), encryption ($FHE.Enc$), Evaluation ($FHE.Eval$), and decryption ($FHE.Dec$). Together they provide privacy and decryption correctness.

Given two cipher-texts, $C_1 = Enc_{pk}(M_1)$ and $C_2 = Enc_{pk}(M_2)$ and, two plain-texts, $M_+ = M_1 + M_2$ and $M_* = M_1 * M_2$ are well-defined.

We say that the encryption scheme is fully homomorphic if both of the following hold:

- $FHE.Eval(pk, (C_0 + C_1)) = C_+$ such that, $FHE.Dec(C_+) = M_1 + M_2$
- $FHE.Eval(pk, (C_0 * C_1)) = C_*$ such that, $FHE.Dec(C_*) = M_1 * M_2$

Most solutions for fully homomorphic encryption rely on hard lattice problems. Accordingly, the resulting cipher-texts must contain a certain level of noise to guarantee the security of the encryption.

The issue though is that computing homomorphically increases the noise level in the cipher-text. As long as the noise is below a certain threshold, the cipher-text can be decrypted. If the noise grows too much, it can overflow on the data itself, rendering decryption impossible.

Bootstrapping: The noise on each step of FHE is assumed to be not overflowing, but the noise gets accumulated very fast, especially in the case of multiplication. So, we want to refresh the cipher-text, using a noise-reduction operation so that the message stays encrypted but the associated noise becomes lower. This technique is called **Bootstrapping**.

TFHE: TFHE (Fast Fully Homomorphic Encryption over the Torus) scheme is a powerful version of FHE. The scheme operates over the **torus** ($\mathbb{T} = \mathbb{R}/\mathbb{Z}$). This allows for efficient arithmetic operations. Plain-text bits are encoded into torus values and then encrypted using a secret key. The encryption process includes adding noise to ensure security.

The security of the scheme is based on the hard lattice problem **Learning With Errors (LWE)**, and its variants, such as **Ring-LWE**. In fact, the majority of FHE schemes used nowadays are **LWE**-based and use noisy cipher-texts.

Programmable Bootstrapping: TFHE is distinguished from the other FHE schemes because it is optimized for speed, as it proposes a special bootstrapping which is very fast and able to evaluate a function at the same time as it reduces the noise. This bootstrapping technique is called programmable bootstrapping.

5.2.1 Aim:

The aim of this literature is to construct an OPRF and an round-optimal (optimizing the number of communication rounds) POPRF based on the before-mentioned main weak PRF candidate using some FHE-based techniques.

5.2.2 Candidate OPRF

They start with the generalised main weak PRF, $F_{weak} : \mathbb{Z}_p^{m_p \times n_p} \times \mathbb{Z}_p^{n_p} \rightarrow \mathbb{Z}_q$

$$F_{weak}(\mathbf{A}, \mathbf{x}) = \sum_{j=0}^{m_p-1} (\mathbf{A} \cdot \mathbf{x} \bmod p)_j \bmod q \text{ [where } p < q \text{ are primes]}$$

They introduce a fixed public matrix $\mathbf{G}_{inp} \in \mathbb{Z}_q^{n_q \times n}$ and a p-adic decomposition operation, $decomp : \mathbb{Z}_q^{n_q} \rightarrow \mathbb{Z}_p^{[\log_p(q)] \cdot n_q}$, where $[\log_p(q)] \cdot n_q = n_p$

The strong PRF candidate is $F_{one} : \mathbb{Z}_p^{m_p \times n_p} \times \mathbb{Z}_p^n \rightarrow \mathbb{Z}_q$ such that,

$$F_{one}(\mathbf{A}, \mathbf{x}) := F_{weak}(\mathbf{A}, decomp(\mathbf{G}_{inp} \cdot \mathbf{x} \bmod q)).$$

As we have seen in the previous literature study, that to extend the small output of the PRF construction, they introduce another matrix, $G_{out} \in \mathbb{Z}_q^{m \times m_p}$ (with $m < m_p$). Hence, the full PRF becomes $F_{strong} : \mathbb{Z}_p^{m_p \times n_p} \times \mathbb{Z}_p^n \rightarrow \mathbb{Z}_q^m$, where,

$$F_{strong}(\mathbf{A}, \mathbf{x}) := G_{out} \cdot (\mathbf{A} \cdot decomp(\mathbf{G}_{inp} \cdot \mathbf{x} \bmod q) \bmod p) \bmod q$$

Note that, $decomp(\mathbf{G}_{inp} \cdot \mathbf{x} \bmod q) \in \mathbb{Z}_p^{n_p}$ does not depend on the PRF key. Thus, in an OPRF construction it could be precomputed and submitted by the client knowing \mathbf{x} .

This strong PRF candidate maps 0 to 0, (which holds with negligible probability for a random function) and thus trivially distinguished from a random function.

They thus define,

$$F_{strong}(\mathbf{A}', \mathbf{x}) := G_{out} \cdot (\mathbf{A} \cdot (decomp(\mathbf{G}_{inp} \cdot \mathbf{x} \bmod q), \mathbf{1}) \bmod p) \bmod q$$

for $\mathbf{A}' \in \mathbb{Z}_p^{m_p \times (n_p+1)}$, i.e. extended by one column.

Furthermore, they want to support an additional input $\mathbf{t} \in \mathbb{Z}_p^n$ to be submitted in the clear. They deploy a key-derivation function to derive a fresh key per tag \mathbf{t} [CHL22, JKR18]. Let, $RO_{key} : \mathbb{Z}_p^{m_p \times n_p} \times \mathbb{Z}_p^n \rightarrow \mathbb{Z}_p^{m_p \times (n_p+1)}$ be a random oracle. The PRF candidate $F_{\mathbf{A}}^{RO_{key}}(\mathbf{t}, \mathbf{x})$ is defined by the algorithm:

Input: $\mathbf{A} \in \mathbb{Z}_p^{m_p \times n_p}$, $\mathbf{x} \in \mathbb{Z}_p^n$, $\mathbf{t} \in \mathbb{Z}_p^n$
Output: $F_{\mathbf{A}}(\mathbf{t}, \mathbf{x})$
 $\mathbf{A}_{\mathbf{t}} \leftarrow RO_{key}(\mathbf{A}, \mathbf{t})$
 $\mathbf{y} \leftarrow decomp(\mathbf{G}_{inp} \cdot \mathbf{x} \bmod q)$
 $\mathbf{z} \leftarrow \mathbf{G}_{out} \cdot (\mathbf{A}_{\mathbf{t}} \cdot (\mathbf{y}, \mathbf{1}) \bmod p) \bmod q$
 return \mathbf{z}

Now, the (P)OPRF construction algorithm is shown according to the formal definition of (P)OPRF. [Note that, **NIZK (Non-Interactive Zero Knowledge)** function is used to prove the validity of a statement without revealing any further informations.]

- $\mathcal{F}.Setup(1^\lambda)$ $\mathcal{F}.KeyGen(1^\lambda)$
 $\mathbf{A}_{pp} \leftarrow \$ \mathbb{Z}_Q^{N \times N}$ $pk \leftarrow \perp$
 $pp \leftarrow \mathbf{A}_{pp}$ $sk \leftarrow \$\mathbb{Z}_p^{m_p \times n_p}$
return pp return(pk, sk)
- $\mathcal{F}.Request(pk, \mathbf{t}, \mathbf{x}; pp)$
 $FHE.pk^{pp}, FHE.sk \leftarrow \$FHE.KeyGen^{pp}()$
 $\mathbf{y} \leftarrow decomp(\mathbf{G}_{inp} \cdot \mathbf{x} \bmod p)$
 $ct \leftarrow \$FHE.Enc(FHE.sk, \mathbf{y})$
 $W \leftarrow \$NIZK(FHE.pk^{pp}, ct; FHE.sk, x)$
 $req \leftarrow (FHE.pk^{pp}, ct, W, t)$
return req
- $\mathcal{F}.BlindEval(sk=\mathbf{A}, \mathbf{t}, req; pp)$
 $(FHE.pk^{pp}, ct, W) \leftarrow req$
 $\mathbf{A}_t \leftarrow RO_{key}(\mathbf{A}, \mathbf{t})$
 $ct' \leftarrow \mathcal{F}.HEEval(FHE.pk^{pp}, \mathbf{A}_t, \mathbf{t}, ct)$
if W does not verify then $ct' = \perp$
return rep
- $\mathcal{F}.Finalise(rep = ct)$
if ct' not a ciphertext, return \perp
 $z' \leftarrow FHE.Dec(FHE.sk, ct)$
 $z \leftarrow RO_{fin}(t, x, z')$
return z
- $\mathcal{F}.Eval(sk=\mathbf{A}, \mathbf{t}, \mathbf{x})$
 $z' := F_{\mathbf{A}}^{RO_{key}}(t, x)$
 $z := RO_{fin}(t, x, z')$
return z

The main point of interest is that the input LWE ciphertexts have plaintext space \mathbb{Z}_p and the output LWE ciphertexts have plaintext space \mathbb{Z}_q in the below mentioned step. In order to perform the plaintext modulus switch, they use a variant of the standard TFHE programmable bootstrapping. $\mathcal{F}.HEEval$ function is used for verification of client's correct input using NIZK scheme.

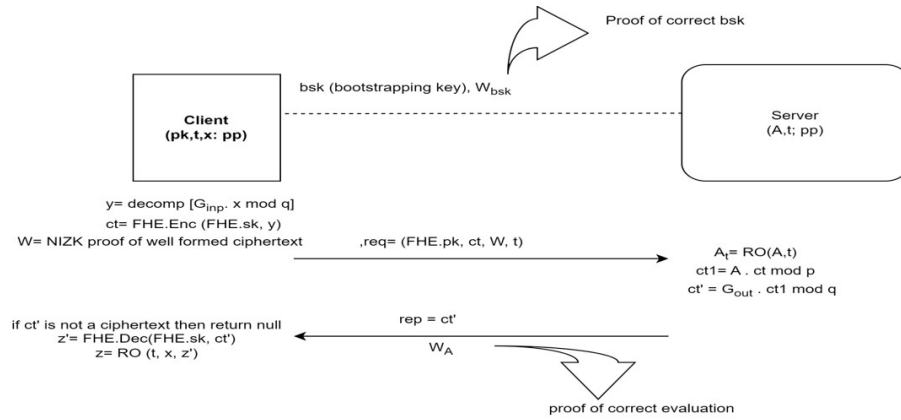


Figure 4: Interaction between client and server

5.2.3 Hardness

- Clearly, the entire construction is critically dependent on FHE scheme and bootstrapping. **LWE** assumption is the core of the FHE scheme used in this context.
- **Matrix-NTRU** assumption is required to prove the above-mentioned construction is indeed an POPRF.
- To achieve security against “honest-but-curious” server, **LWE** assumption is needed once again.

After the thorough study of these cryptographic protocols, we have learned about the different “crypto dark matter” constructions and their underlying hard problems. The conclusions, which we have drawn from here, are stated in the next section.

Conclusion and Future Developments

Conclusion

This survey provides a detailed study of lattice and the traditional lattice-based “hard” problems such as LWE, SIS, NTRU and their variants. We have explained what is meant by “crypto dark matter” and then we have thoroughly studied couple of crypto dark matter constructions (weak PRFs and POPRFs). We have also found out how the traditional “hard” problems have been used in these constructions. Finally, few gaps in computations were identified and have been worked on to provide for simple understanding of the protocols.

Future Developments

- Finding new reductions and weaknesses of these traditional assumptions. For example, the reduction of LWR in constant-size composite modulus setting.
- We have come across an unorthodox source of hardness which is, mixing linear functions over different moduli. Cryptanalysis of this may need new techniques that may themselves have other applications.
- Verification or refinement of the hardness conjectures proposed for the construction of the main weak PRF candidate.
- An efficient bunch of novel cryptographic primitives may get constructed, inspired by these assumptions.
- Getting insights into the security landscape of lattice-based cryptography, including the resilience of proposed schemes against quantum attacks and their practical feasibility using reduction-based proofs and other mathematical techniques.
- We will further study different “crypto dark matter” constructions to gain better understanding of how to use lattice-based assumptions as building blocks for these under-explored cryptographic protocols.

References

- [ABG⁺14] Adi Akavia, Andrej Bogdanov, Siyao Guo, Akshay Kamath, and Alon Rosen. Candidate weak pseudorandom functions in $AC^0 \circ MOD_2$. In ITCS 2014, pages 251-260, 2014.
- [ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In CRYPTO, pages 595-618, 2009.
- [ADDG24] Albrecht, M.R., Davidson, A., Deo, A., Gardham, D.: Crypto dark matter on the Torus: oblivious PRFs from shallow PRFs and FHE. Cryptology ePrint Archive, Report 2023/232 (2023). <https://eprint.iacr.org/2023/232>
- [AGR14] Andrew Arnold, Mark Giesbrecht, and Daniel S Roche. Sparse interpolation over finite fields via low-order roots of unity. In Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation, pages 27-34. ACM, 2014.
- [AKPW13] Joel Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In CRYPTO, pages 57-74, 2013.
- [ASA16] Jacob Alperin-Sheriff and Daniel Apon. Dimension-preserving reductions from LWE to LWR. Cryptology ePrint Archive, Report 2016/589, 2016. <http://eprint.iacr.org/2016/589>.
- [BGM⁺16] Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. On the hardness of learning with rounding over small modulus. In TCC 2016-A, pages 209-224, 2016.
- [BIP⁺18] Dan Boneh, Yuval Ishai, Alain Passelegue, Amit Sahai, and David J. Wu. Exploring crypto dark matter: New simple PRF candidates and their applications. In TCC, pages 699-729, 2018.
- [BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In CRYPTO, pages 410-428, 2013.
- [BOT88] Michael Ben-Or and Prasoona Tiwari. A deterministic algorithm for sparse multivariate polynomial interpolation (extended abstract). In ACM STOC, pages 301-309, 1988.
- [BP14] Abhishek Banerjee and Chris Peikert. New and improved key-homomorphic pseudorandom functions. In CRYPTO, pages 353-370, 2014.
- [BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In EUROCRYPT, pages 719-737, 2012.

-
- [Gen09] Craig Gentry: A fully homomorphic encryption scheme. Stanford University, USA, 2009
- [GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In CRYPTO, pages 276-288, 1984.
- [GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792-807, October 1986.
- [GRS08] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. HB#: Increasing the security and efficiency of HB+. In EUROCRYPT, pages 361-378, 2008.
- [GS09] Sanchit Garg and Eric Schost. Interpolation of polynomials given by straight-line programs. *Theoretical Computer Science*, 410(27-29):2659-2662, 2009.
- [KY88] Erich Kaltofen and Lakshman Yagati. Improved sparse multivariate polynomial interpolation algorithms. In *International Symposium on Symbolic and Algebraic Computation*, pages 467-474. Springer, 1988.
- [NR95] Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. In FOCS, pages 170-181, 1995.
- [Pie12] Krzysztof Pietrzak. Cryptography from learning parity with noise. In SOFSEM, pages 99-114, 2012.
- [Raz87] Alexander A. Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition (russian). *Matematicheskie Zametki*, 41(4):598-607, 1987. English translation in *Mathematical Notes of the Academy of Sci. of the USSR*, 41(4):333-338, 1987.
- [Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In ACM STOC, pages 84-93, 2005.
- [Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In ACM STOC, pages 77-82, 1987.
- [Wer94] Kai Werther. The complexity of sparse polynomial interpolation over finite fields. *Applicable Algebra in Engineering, Communication and Computing*, 5(2):91-103, 1994.
- [Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In EUROSAM, pages 216-226, 1979.
- [Zip90] Richard Zippel. Interpolating polynomials from their values. *J. Symb. Comput.*, 9(3):375-403, 1990.