MDPI

Review

REVIEW ON IPQ-LWE-TCFs

Firstname Lastname [1,†,‡], Firstname Lastname [2,‡] and Firstname Lastname [2,*]

[1] Affiliation 1; e-mail@e-mail.com
[2] Affiliation 2; e-mail@e-mail.com
* Correspondence: e-mail@e-mail.com; Tel.: (optional; include country code; if there are multiple corresponding authors, add author initials) +xx-xxxx-xxx-xxxx (F.L.)
† Current address: Affiliation.
‡ These authors contributed equally to this work.

Abstract: The recent development of quantum computers poses a significant threat to classical cryptographic systems, necessitating a collaborative effort between quantum computing and cryptography researchers. This review aims to bridge the gap between these fields by providing an accessible overview of Proof of Quantumness (PQ), Trapdoor Claw-free Functions, and the Learning with Errors (LWE) problem for those not experienced in both areas. The PQ protocol offers a method for a classical verifier to validate the quantum capabilities of a quantum prover. Additionally, we explore the definition and applications of Trapdoor Claw-free Functions in constructing quantum-resistant cryptographic protocols and give a brief overview of the foundational aspects of the LWE problem in post-quantum cryptography. Our review underscores the necessity for knowledge exchange between QC and cryptography communities in the quantum era by presenting these complex topics in an approachable manner.

Keywords: interactive proof of quantumness, trapdoor claw-free function, lattice, lwe

Citation: Lastname, F.; Lastname, F.; Lastname, F. Title. *Cryptography* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

1.1. Overview of Quantum Computing and Post-quantum Cryptography

Quantum computing represents a transformative leap in computational capabilities, using the principles of quantum mechanics to process information in fundamentally new ways. Unlike classical computers, which use bits as the smallest unit of data, quantum computers use quantum bits or qubits. These qubits do not exist in a specific state but in a superposition of multiple states, enabling quantum computers to perform certain computations exponentially faster than their classical counterparts.

The need for secure communication has been a constant throughout history, driving the development of cryptographic techniques to protect sensitive information. The security of classical cryptography largely relies on mathematical problems assumed to be computationally hard to solve. These problems, such as integer factorization and the discrete logarithm problem, form the basis of widely used algorithms like RSA and ECC. However, the development of quantum computing presents a significant challenge. Shor's algorithm [1], for example, can efficiently solve the problems mentioned above, making these cryptographic primitives vulnerable to future quantum attacks. While not all classical cryptographic primitives are vulnerable to these attacks, RSA/ECC form the basis of contemporary public-key infrastructure, essential to the functionality of the internet.

In recent years, the rapid development of quantum computing has raised major security concerns in these cryptographic schemes. Consequently, post-quantum cryptography (PQC) is now a thriving field with the view of developing new cryptographic constructions secure against classical and quantum attacks, often referred to as 'quantum resistance'. In the context of post-quantum cryptography, many approaches are using different building blocks: *hash-based, code-based, multivariate-based, isogeny-based and lattice-based schemes:*

- Hash-based cryptography: Utilising the hard-to-invert property of hash functions to build digital signature schemes.
- Code-based constructions derive their security from decoding generic linear codes.
- Multivariate-based cryptography uses the NP-hard Multivariate Quadratic Problem (MQ Problem) over finite fields $\mathbb{F}_p$ to construct cryptosystems.
- Isogeny-based cryptography is the youngest field whose constructions are based on the hardness of finding special maps called isogenies between supersingular elliptic curves.
- Lattice-based cryptography is based on the computational difficulty of mathematical problems associated with lattices: short integer solution (SIS), learning with errors (LWE), and their variants.

Lattice-based cryptography is arguably the most promising branch in post-quantum cryptography today due to its versatility, supporting a wide range of symmetric and asymmetric cryptographic primitives including digital signatures and public-key encryption. Additionally, lattice-based techniques are computationally efficient with compact key sizes.

Another significant field that has emerged in recent years is quantum cryptography, which has a major impact on symmetric key encryption constructions by achieving information-theoretic security. An example of this is Quantum Key Distribution (QKD) [2,3], a technique that allows two parties to generate a shared secret key, which can then be used for symmetric encryption. QKD leverages the principles of quantum mechanics to ensure that any attempt at eavesdropping on the key exchange can be detected; hence providing a level of security called information-theoretic security.

However, this does not apply to asymmetric key encryption because one of the keys always remains public. For asymmetric constructions, developing post-quantum cryptography (PQC) is still the main research goal. Based on "hard-to-solve" assumptions, PQC faces the same problem as classical cryptography, i.e., computational security.

1.2. Communication between a quantum computer and classical devices.

The focus of this review is on the proof of quantumness, which explores the intersection between the classical and quantum realms, where communication between a quantum computer and classical devices is a critical aspect of integrating quantum technology into existing frameworks. This interaction often requires protocols that allow classical devices to verify the quantum power of other devices. One such protocol is the Proof of Quantumness (PQ), which provides a method for a classical verifier to interact with a quantum prover. The PQ protocol ensures that the quantum device can demonstrate its quantum capabilities by performing computations that are infeasible for classical devices, thereby establishing a proof of quantumness. This verification is essential for tasks where quantum advantages need to be reliably authenticated, such as in secure communications and advanced cryptographic protocols. By bridging the gap between quantum and classical worlds, PQ protocol facilitates the trust and usability of quantum computers in practical applications.

One recent approach is to use Trapdoor Claw-free functions to build Proof of Quantumness protocols. Note that Trapdoor Claw-free functions still base their security on hard problems like learning with errors (LWE) and ring learning with errors (RLWE), making them computationally secure but not information-theoretically secure.

1.3. Notation

The following notations are used throughout this paper:

- q: prime integer;
- λ: security parameter;
- $\mathbb{N}$: the set of natural numbers;
- $\mathbb{Z}$: the set of integers;
- $\mathbb{Z}_p$: the ring of integers modulo p;
- $x \leftarrow S$: the action of sampling a uniformly random element x from the set S;

- $p(n)$ a polynomial function associated with the security parameter n. A function $\text{negl}(n)$ is said to be negligible if $\text{negl}(n)$ is smaller than all polynomial fractions for sufficiently large n. We define an event to occur with overwhelming probability when its probability of occurrence is at least $1 - \text{negl}(n)$.
- Vectors are represented using lowercase bold letters (e.g. $\mathbf{s}$), while matrices are represented in uppercase bold (e.g. $\mathbf{A}$).
- The inner product of vectors is denoted using angular brackets, e.g. $\langle \mathbf{a}, \mathbf{s} \rangle$.
- The transpose of a vector or matrix is denoted using a superscript 'T', e.g. $\mathbf{s}^T$ or $\mathbf{A}^T$, respectively.
- $\| \cdot \|$ denotes the Euclidean norm, $\| \cdot \|_\infty$ denotes the infinity norm (the absolute value of the largest component of the vector)
- $\mathcal{L}(\mathbf{A})$: a lattice with basis $\mathbf{A}$.
- For a function $f : X \to \mathbb{R}$ over a finite domain X, the support of f, denoted by SUPP, is the set of points in X where f is non-zero,

$$\text{SUPP}(f) = \{x \in X | f(x) \neq 0\}.$$

2. Quantum computing and communications

2.1. Quantum states

In quantum mechanics, a *quantum state* is a unit vector in the complex space $\mathbb{C}^N$. As in classical computing, all computations are performed using classical bits, 0 and 1. In quantum computing, we use an equivalent term that has more interesting properties: the *quantum bit* or *qubit*. A qubit, a two-level system, can be considered the simplest unit of quantum information. Let us use Dirac notation and denote $|0\rangle$ and $|1\rangle$ as the two basis states of a qubit, similar to the classical case. The $|0\rangle$ and $|1\rangle$ are called 'kets', more explanation about the use of this notation will be proposed soon in this section.

The first special property of a quantum state is *superposition*, where a quantum system is not necessarily in one specific state but can be in a *superposition* of them. Hence, a quantum state $|\psi\rangle$ can be expressed as a linear combination of the two basis states $|0\rangle$ and $|1\rangle$,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \tag{1}$$

where the coefficients α and β are complex numbers called amplitudes, and the pair $\{|0\rangle, |1\rangle\}$ is orthogonal and is called computational basis.
Recall that quantum states can be written as vectors in complex space; hence, using the Dirac notation for describing quantum states simplifies linear algebra operations while working with these states. Note that kets are column vectors, for example:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \; |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{2}$$

The superposition state (1) can be represented as the following vector:

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}. \tag{3}$$

Working with row vectors in complex vector space, we also need to deal with *conjugate transpose*, ie. row vectors. In the Dirac notation, these row vectors are called "bras", for example

$$\langle 0| = \begin{bmatrix} 1 & 0 \end{bmatrix}, \; \langle 1| = \begin{bmatrix} 0 & 1 \end{bmatrix}, \tag{4}$$

and

$$\langle \psi| = \begin{bmatrix} \alpha^* & \beta^* \end{bmatrix} \tag{5}$$

where α^* and β^* are complex conjugates of α and β.

Another property of quantum states is *measurement* where it helps one ascertain information about the quantum states. When one measures the state $|\psi\rangle$ in the computational basis, the state will *collapse* to one of its possible outcomes, either $|0\rangle$ or $|1\rangle$, effectively converting quantum information into classical information. The amplitudes α and β now represent the probability of obtaining the corresponding results, where $|\alpha|^2$ is the probability of collapsing to $|0\rangle$ and $|\beta|^2$ is the probability of collapsing to $|0\rangle$. For this reason, we have that the measurement result is probabilistic and not predetermined.

$$\begin{aligned} \Pr[0] &= |\alpha|^2, \\ \Pr[1] &= |\beta|^2, \\ |\alpha|^2 + |\beta|^2 &= 1. \end{aligned} \tag{6}$$

This collapse of quantum states is not reversible, meaning once the measurement is made, the original superposition state is lost, and we are left with classical information. This property of quantum measurement underlies many quantum algorithms and cryptographic protocols, impacting how information is extracted from quantum systems and ensuring that certain quantum properties, like entanglement and superposition, can be harnessed effectively only until the measurement is performed.

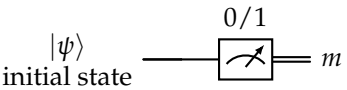


Figure 1: Measuring a quantum state: The thin line denotes quantum information and the double lines denote classical information. The result of the measurement m is a classical result according to a probability distribution (6).

In quantum computing, basis quantum gates are fundamental operations used to manipulate qubits, similar to classical logic gates in classical computing. These quantum gates are represented by unitary matrices $\mathbf{U}$, which preserve the norm of quantum states and ensure reversibility. A matrix $\mathbf{U}$ is called unitary if and only if it satisfies

$$\mathbf{U}\mathbf{U}^\dagger = \mathbf{U}^\dagger\mathbf{U} = \mathbf{I}, \tag{7}$$

where $\mathbf{U}^\dagger$ is the conjugate transpose of $\mathbf{U}$. Among the most important of these quantum gates are the Pauli gates and the Hadamard gate. The Pauli gates consist of the Pauli-X, Pauli-Y, and Pauli-Z gates. The Pauli-X gate, represented by the matrix

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{8}$$

which functions as a quantum equivalent of the classical NOT gate, flipping the state of a qubit from $|0\rangle$ to $|1\rangle$ and vice versa. Concretely,

$$X|\psi\rangle = X(\alpha|0\rangle + \beta|1\rangle) = \alpha|1\rangle + \beta|0\rangle. \tag{9}$$

The Pauli-Z gate, represented by the matrix

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{10}$$

performs a phase-flip, changing the sign of the $|1\rangle$ state while leaving the $|0\rangle$ state unchanged:

$$Z|\psi\rangle = Z(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle - \beta|1\rangle. \tag{11}$$

The Pauli-Y gate, represented by the matrix

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{12}$$

combines a bit-flip and a phase-flip.
The Hadamard gate H, another essential quantum gate, is represented by the matrix

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}. \tag{13}$$

This gate transforms the computational basis states into equal superpositions of $|0\rangle$ and $|1\rangle$, where

$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} = |+\rangle, \tag{14}$$

$$H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} = |-\rangle. \tag{15}$$

When discussing measurements in quantum computing, we consider a specific property of the qubit, and using different bases yields varied outcomes, similar to how measuring classical objects differs based on what is being measured. While the computational basis $(|0\rangle, |1\rangle)$ is commonly used, other bases, like the Hadamard basis $\{|+\rangle, |-\rangle\}$, are also important. For example, measuring the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ using the Hadamard basis, the state will collapse to either $|+\rangle$ or $|-\rangle$ with probability $|\alpha + \beta|^2/2$ and $|\alpha - \beta|^2/2$, respectively.

2.2. Hadamard Transform and Quantum Parallelism

Note that when one applies the Hadamard gate to each qubit in an n-qubit system, each initialized to $|0\rangle$, it creates a uniform superposition of all possible 2^n superposition, represented as

$$H^{\otimes n}|\mathbf{0}\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle. \tag{16}$$

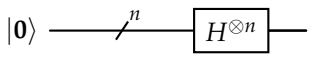


Figure 2: Quantum circuit applying the Hadamard transform to n-qubit system.

This superposition is crucial for quantum computing because it allows a quantum computer to process multiple inputs simultaneously. When a function $f(\mathbf{x})$ is evaluated on this input register, it effectively evaluates the function for all possible inputs $\mathbf{x}$ in parallel, which significantly enhances the computational power of quantum computers. This phenomenon is also called quantum parallelism. Concretely, one first uses an auxiliary qubit register (initialized to $|0\rangle$) to store the function's output then uses a quantum circuit evaluating the classical function U_f that maps each basis state $|\mathbf{x}\rangle$ to $|\mathbf{x}\rangle|f(\mathbf{x})\rangle$:

$$U_f\Big(\frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle|0\rangle\Big) = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle|f(\mathbf{x})\rangle \tag{17}$$

By measuring the output register, which holds $f(\mathbf{x})$, one can obtain the function's value for a specific input $\mathbf{x}$.

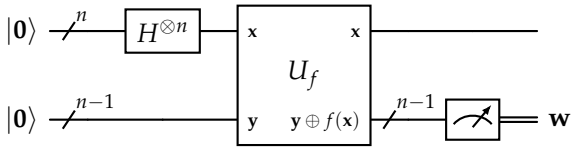


Figure 3: Evaluation of a classical function $f(\mathbf{x})$ on quantum registers.

2.3. No-cloning theorem

The no-cloning theorem is a fundamental principle in quantum mechanics that states it is impossible to create an exact copy of an arbitrary unknown quantum state. The no-cloning theorem has a profound impact on cryptography, especially in quantum key distribution (QKD). The security of the BB84 protocol [2], a well-known QKD construction, relies on the impossibility of an adversary perfectly cloning the quantum states, preventing them from interfering with the states being transferred between honest parties. Furthermore, any attempt to clone a quantum state by measuring it would collapse the state, exposing the eavesdropper's existence.

3. Lattice-based post-quantum cryptography

3.1. Lattices

3.1.1. Definitions

Definition 1. *Lattices in $\mathbb{R}^n$ are discrete subgroups of $\mathbb{R}^n$. Consider integer lattice $\mathcal{L} \subseteq \mathbb{Z}^n$, $\mathcal{L}$ can be represented by a basis $\mathbf{A} = [\mathbf{a}_1, \cdots, \mathbf{a}_n] \in \mathbb{Z}^{m \times n}$, where each vector $\mathbf{a}_i$ is written in column form as follow:*

$$\mathcal{L}(\mathbf{A}) := \left\{ \sum_{i=1}^{n} \mathbf{a}_i x_i | x_i \in \mathbb{Z} \; \forall i = 1, \cdots, n \right\} \subseteq \mathbb{Z}^m. \tag{18}$$

We refer to n as the rank of the lattice $\mathcal{L}$ and m as its dimension. $\mathcal{L}$ is called a full-rank lattice if $n = m$. In most lattice-based cryptographic constructions, we primarily utilize full-rank lattices that include $q\mathbb{Z}^m$, known as q-ary lattices lattices, which are defined as follows

$$\mathcal{L}_q(\mathbf{A}) := \left\{ \mathbf{x} \in \mathbb{Z}^m \text{ s.t. } \exists \mathbf{s} \in \mathbb{Z}_q^n \text{ where } \mathbf{A}^{\mathrm{T}}\mathbf{s} = \mathbf{x} \mod q \right\} \tag{19}$$

$$\mathcal{L}_q^{\perp}(\mathbf{A}) := \{ \mathbf{x} \in \mathbb{Z}^m \text{ s.t. } \mathbf{A}\mathbf{x} = 0 \mod q \} \tag{20}$$

However, to simplify the notation, we will omit q and simply write $\mathcal{L}(\mathbf{A})$ and $\mathcal{L}^{\perp}(\mathbf{A})$ for a given matrix $\mathbf{A}$.

Consider a basis $\mathbf{A}$ of $\mathcal{L}$, we called $P(\mathbf{A}) = \{\mathbf{A}\mathbf{x} | \mathbf{x} \in [0,1)^n\}$ is the fundamental parallelepiped of $\mathbf{A}$. A "good" basis of $\mathcal{L}$ results in a parallelepiped that is more square-like, whereas a "bad" basis produces a very thin parallelepiped. We denote $\|\mathbf{A}\| := \max_{1 \le i \le k} \|\mathbf{a}_i\|$ the maximum l_2 length of the vectors in $\mathbf{A}$ and $\tilde{\mathbf{A}} := \{\tilde{\mathbf{a}}_1, \cdots, \tilde{\mathbf{a}}_k\}$ the Gram-Schmidt orthogonalization of the vectors $\mathbf{a}_1, \cdots, \mathbf{a}_k$ in that order.

For a given lattice $\mathcal{L}$, let $\mathbf{s}$ denote the shortest vector of $\mathcal{L}$. One basic parameter of a lattice is the length of this shortest vector, denoted as $\lambda_1(\mathcal{L}) = \|\mathbf{s}\|$. This parameter is also called *successive minima* if one considers the scenario where λ_1 is the smallest r such that all the lattice points inside a ball of radius r $\overline{\mathcal{B}}(0, r)$ span a space of dimension 1. Similarly, we can define the n^{th} successive minimum $\lambda_n(\mathcal{L})$ this lattice as the smallest number r that all the lattice points inside $\overline{\mathcal{B}}(0, r)$ span a space of dimension n.

Hence, one of the most important problems in lattice-based cryptography is the Shortest Vector Problem (SVP), which asks to find the shortest nonzero vector $\mathbf{s}$ in a given lattice $\mathcal{L}(\mathbf{A})$ with an arbitrary basis $\mathbf{A}$. There are also approximate versions of this problem

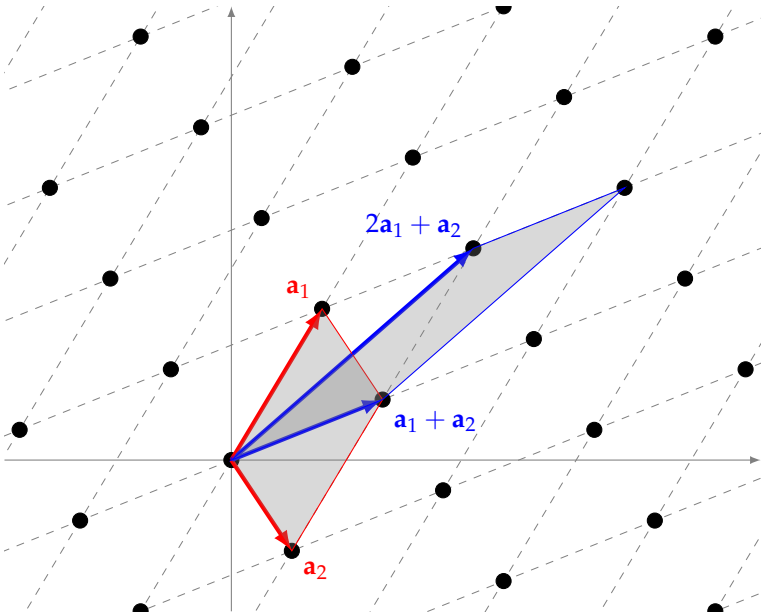


Figure 4: Consider a two-dimensional lattice $\mathcal{L}$, a basis for $\mathcal{L}$ consists of two non-zero vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ such that any vector in $\mathcal{L}$ can be written as a linear combination of $\mathbf{a}_1$ and $\mathbf{a}_2$, e.g. $\mathbf{a}_1 + \mathbf{a}_2$, $2\mathbf{a}_1 + \mathbf{a}_2 \in \mathcal{L}$. The grey areas are the fundamental parallelepipeds formed by different bases of $\mathcal{L}$. A good basis for $\mathcal{L}$ will have vectors with lengths that are close to each other and an angle between them that is close to 90 degrees and forms a more square-like parallelepiped.

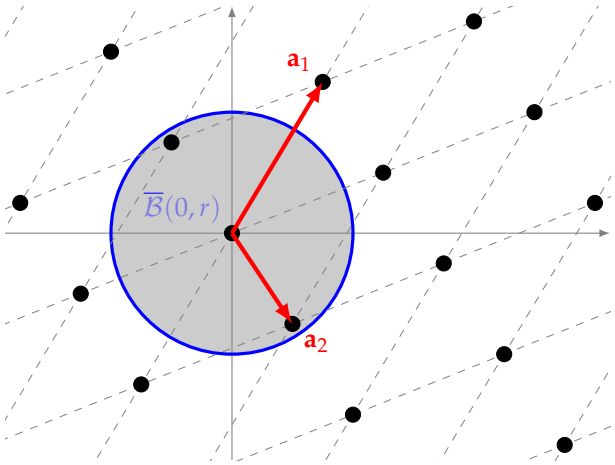


Figure 5: A lattice $\mathcal{L}$ with the successive minimum $\lambda_1 = \|v\| = r$ where all the lattice points inside a ball of radius r $\overline{\mathcal{B}}(0,r)$ span a space of dimension 1.

called SVP_γ where the goal is to find a nonzero vector that is of length at most $\gamma = \gamma(n) \geq 1$ times the length of the optimal solution.

The "decision" versions of the problem ask to determine whether a given number d is the minimum distance of $\mathcal{L}(\mathbf{A})$, i.e., $d \leq \lambda_1(\mathcal{L})$ or $d \geq \lambda_1(\mathcal{L})$. This problem is known to be NP-hard in the $\|\cdot\|_\infty$ norm, meaning there is no known polynomial-time algorithm to solve it [4]. For the $\|\cdot\|$ Euclid norm, it remains a long-standing open problem.

Another important problem is the Shortest Independent Vectors Problem (SIVP_γ) with an approximation factor γ, a given a lattice $\mathcal{L}(\mathbf{A})$ with an arbitrary basis $\mathbf{A}$, SIVP_γ asks to find find n linearly independent lattice vectors of length at most $\gamma \cdot \lambda_n(\mathcal{L})$ where $\lambda_n(\mathcal{L})$ is the n^{th} successive minimum of $\mathcal{L}$. Like SVP, SIVP is also NP-hard.

Many hardness problems in lattice-based cryptography can be **"reduced"** to instances of SVP or SIVP, for more information about reductions between lattice problems, please refer to 3.1.2 and [5].

3.1.2. Reductions

Within the realm of computational hardness challenges, the term "reduction" denotes a methodological approach whereby one problem is transformed into another, such that solving the second problem enables solving the first. This is a formal way to compare the intrinsic difficulty of different problems. Specifically, if problem A can be reduced to problem B, and if one has a solution to B, then A can be solved. This doesn't necessarily imply that A is as hard as B, but that B is at least as hard as A.

In the analysis of problem hardness, distinctions are made between *worst-case* and *average-case* scenarios. The *worst-case* scenario is the most difficult possible instance for a given problem. At the same time, the *average-case* looks at the expected performance of an algorithm across all possible instances of a problem, usually under some reasonable assumption about the distribution of these instances.

In Post-Quantum Cryptography (PQC), reductions are the main technique used to prove the security of cryptographic protocols by reducing their hardness to well-known hard problems. If such a reduction exists, then finding an efficient way to break the system would imply an efficient solution to the underlying hard problem, which is presumed not to exist, thus the security of the protocol holds.

3.2. SIS and LWE problems

Let us first consider solving the linear equation

$$\mathbf{As} = \mathbf{y} \mod q \tag{21}$$

given $n, m \in \mathbb{Z}$, a prime number q, a matrix $\mathbf{A}$ in $\mathbb{Z}_q^{m \times n}$ and $\mathbf{y}$ is a vector in $\mathbb{Z}_q^m$.

We can now view the matrix $\mathbf{A}$ as a basis for a lattice $\mathcal{L}(\mathbf{A})$. Solving Eq. (21) is achievable through Gaussian elimination. However, its variations and many other related lattice problems, are known to be NP-hard (based on the factors of polynomial approximation problems). Here, we highlight some significant lattice problems: **The shortest integer solution (SIS)** and **The Learning With Errors (LWE)** problems, further information on lattice-related problems can be found in [6].

3.2.1. The shortest integer solution (SIS)

Recall that initially, we are attempting to find a solution for the equation $\mathbf{As} = \mathbf{y} \mod q$. Consider an additional constraint that $\mathbf{s}$ is "short", i.e $\mathbf{s}$ lies in the subset $S = \{0,1\}^n \subset \mathbb{Z}_q^n$; or more generally, $S = [-B, \ldots, B]^n$ ($B \ll q/2$). Since we are seeking a *short solution* to linear equations, this problem is referred to as the *Short Integer Solution* (SIS) problem. Many cryptographic protocol deal with the case $\mathbf{y} = \mathbf{0}$; hence the $\mathbf{SIS}_{n,m,q,B}$ problem can now be stated as follows:

Definition 2. *Let $n, m, q, B \in \mathbb{N}$ be positive integers. Given a uniformly sampled matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, the shortest integer solution problem asks to find a none zero vector $\mathbf{s} \in \mathbb{Z}_q^m$ such that $\mathbf{As} = \mathbf{0} \mod q$ and $\|\mathbf{s}\|_\infty \leq B$.*

One should think of n as the security parameter that defines the hardness of the problem (the bigger n is, the harder the problem becomes), m usually depends on the specific application, generally $m \gg n$, $q = poly(n)$ and B should be set large enough to ensure that a solution exists, but not trivial to solve. It is proven that solving the $\mathbf{SIS}_{n,m,q,B}$ problem is at least as hard as solving the SVP_γ on worst-case n-dimensional lattices with high probability for some approximation factor γ [6].

3.2.2. The learning with errors problem (LWE)

This subsection recalls the definition of the Learning With Errors (LWE) problem [6]. Let's revisit Eq. (21), we can consider a slight variant of it where the input pair $(\mathbf{A} \in \mathbb{Z}_q^{m\times n}, \mathbf{y} \in \mathbb{Z}_q^m)$ satisfies

$$\mathbf{As} + \mathbf{e} = \mathbf{y} \mod q \tag{22}$$

where n, m, q are positive integers and $\mathbf{e}$ is an additional small noise term sample from an error distribution χ over $\mathbb{Z}_q^m$.

The "search" $\mathsf{LWE}_{n,m,q,\chi}$ problem asks to find $\mathbf{s} \in \mathbb{Z}_q^n$ and the "decision" version asks to to distinguish between the distribution $(\mathbf{A}, \mathbf{y} = \mathbf{As} + \mathbf{e} \mod q)$ and a uniformly random sample $(\mathbf{A}, \mathbf{u})$.

The error term $\mathbf{e}$ effectively affects the exact relationship between the secret $\mathbf{s}$ and $\mathbf{y}$. Specifically, this term $\mathbf{e}$ ensures that although anyone can observe the pairs $(\mathbf{A}, \mathbf{y})$, extracting the information of $\mathbf{s}$ is computationally infeasible if the problem is properly parameterized. The hardness of the $\mathsf{LWE}_{n,m,q,\chi}$ problem (and thus the security of many LWE-based cryptosystems) relies on the assumption that finding $\mathbf{s}$ with random errors $\mathbf{e}$ is as difficult as solving certain worst-case problems on lattices.

3.2.3. Distribution over lattices

Note that the choice of the error distribution χ is crucial. A widely utilized distribution in numerous lattice-based cryptosystems is the Discrete Gaussian distribution. Typically, the discrete Gaussian is chosen in a way that ensures the errors $\mathbf{e}$ are small enough to allow for the correct extraction of the secret information but large enough to prevent adversaries from solving the problem.

For any positive integer n and real $\sigma > 0$, which is taken to be 1 when omitted, the *Gaussian function* $\rho_\sigma : \mathbb{R}^n \to \mathbb{R}^+$ of parameter (or width) σ is defined

$$\rho_\sigma(\mathbf{s}) := \exp(-\pi\|\mathbf{s}\|^2/\sigma^2). \tag{23}$$

A **discrete Gaussian probability distribution** $D_{\mathcal{L},\sigma}$ over a lattice $\mathcal{L}$ is given by

$$D_{\mathcal{L},\sigma}(\mathbf{s}) = \frac{\rho_\sigma(\mathbf{s})}{\rho_\sigma\mathcal{L}}, \forall \mathbf{s} \in \mathcal{L}. \tag{24}$$

The utilisation of Gaussian distributions comes from its ability to effectively connect the Learning With Errors (LWE) problem to the worst-case scenario of the lattice NP-hard problems, thus providing a solid foundation for cryptographic security. More concretely, a long sequence of works has established progressively stronger results about the worst-case lattice problems. It is shown that for any $\alpha > 0$ such that $\sigma = \alpha q \geq 2\sqrt{n}$, the $\mathsf{LWE}_{n,m,q,D^m_{\mathbb{Z}_q,\sigma}}$, where $D_{\mathbb{Z}_q,\sigma}$ is a discrete Gaussian distribution, is at least as hard as approximating the shortest independent vector problem (SIVP_γ) to within a factor of $\gamma = \tilde{O}(n)$ [7,8].

3.2.4. Trapdoor mechanism

In cryptography, a **trapdoor** refers to a concealed backdoor embedded within an algorithm or a specific piece of data that can be used to extract information about some secret. The idea is that, except for the individuals authorised to hold the trapdoor, others cannot extract any information about both the trapdoor and the secret. This characteristic enables secure communication between parties. This subsection revisits certain trapdoor mechanisms to provide a fundamental understanding of the relationship between lattices and their trapdoor in lattice-based cryptography.

The first approach to constructing trapdoors for lattices involves the use of short bases, as these lead to more efficient algorithms for solving lattice problems. Algorithms designed to tackle challenges such as the Shortest Vector Problem (SVP) or the Closest Vector Problem (CVP) become more practical with basis vectors that are short and nearly orthogonal 3.1.1.

This is because the computations involved are simpler and can be executed more swiftly. Specifically, for solving the LWE problem using a short basis, one strategy employs Babai's nearest plane algorithm [9]. The initial step involves utilizing a short basis $\mathbf{T_A}$ of a lattice $\mathcal{L}(\mathbf{A})$ to identify the lattice vector closest to $\mathbf{y}$. Given that the error is sufficiently "small", it becomes feasible to extract the secret vector $\mathbf{s}$'s information. A sequence of works has been carried out to develop more trapdoor mechanisms. Two of the notable approaches are [10], which created "short bases trapdoors," and [11], which constructed "gadget-based trapdoors."

[10] presented an algorithm called $\mathsf{TrapGen}(n, m, q)$ to sample a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ together with an associated basis $\mathbf{T}$ of $\mathcal{L}(\mathbf{A})$ such that $\mathbf{AT_A} = \mathbf{0}$ as the public matrix and its secret trapdoor. The idea is to sample a uniformly random matrix $\mathbf{A}_1$ that has dimension m_1 smaller than m, then extend $\mathbf{A}_1 \in \mathbb{Z}_q^{n\times m_1}$ to $\mathbf{A} = [\mathbf{A}_1|\mathbf{A}_2] \in \mathbb{Z}_q^{n\times m}$ by generating $\mathbf{A}_2 \in \mathbb{Z}_q^{n\times m_2}$ together with a short matrix $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$.

Algorithm 1: $\mathsf{TrapGen}(n, m, q)$ generates a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ together with its trapdoor $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ such that $\mathbf{AT_A} = \mathbf{0}$

function TRAPGEN((n, m, q))

$m = m_1 + m_2$

▷ *Sample a uniformly random matrix* $\mathbf{A}_1$ ◁

$\mathbf{A}_1 \leftarrow \mathbb{Z}_q^{n\times m_1}$

▷ *Generate component matrices* ◁

$\mathbf{U} \in \mathbb{Z}^{m_2\times m_2}$, $\mathbf{R} \in \mathbb{Z}^{m_1\times m_2}$, $\mathbf{P} \in \mathbb{Z}^{m_2\times m_1}$ and $\mathbf{C} \in \mathbb{Z}^{m_1\times m_1}$ such that $\mathbf{U}$ is non-singular and $(\mathbf{GP} + \mathbf{C}) \subset \mathcal{L}^{\perp}(\mathbf{A}_1)$.

▷ *Generate matrix* $\mathbf{A}_2$ ◁

$\mathbf{A}_2 = -\mathbf{A}_1 \cdot (\mathbf{R} + \mathbf{G}) \in \mathbb{Z}_q^{n\times m_2}$.

▷ *Generate the trapdoor* $\mathbf{T_A}$ ◁

$$\mathbf{T_A} = \begin{bmatrix} (\mathbf{G}+\mathbf{R})\mathbf{U} & \mathbf{RP}) - \mathbf{C} \\ \mathbf{U} & \mathbf{P} \end{bmatrix}$$

return $(\mathbf{A} = [\mathbf{A}_1|\mathbf{A}_2], \mathbf{T_A})$

The second approach is the trapdoor technique from [11], called the gadget trapdoor, which is simpler, has a more elegant nature, and supports more efficient cryptographic constructions. [11] first defined a "primitive vector" $\mathbf{g} = \{g_1, \ldots, g_k\} \in \mathbb{Z}_q^k$ where $\gcd(g_1, \ldots, g_k, q) = 1$ and $n \cdot k = m$, then generates the "primitive matrix" (or gadget matrix) $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^T \in \mathbb{Z}_q^{n\times m}$. The gadget trapdoor of a lattice $\mathcal{L}(\mathbf{A})$ is a matrix $\mathbf{R} \in \mathbb{Z}^{(m-\omega)\times\omega}$ such that

$$\mathbf{A}\begin{pmatrix} \mathbf{R} \\ \mathbf{I} \end{pmatrix} = \mathbf{HG} \tag{25}$$

for some invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$. We refer to $\mathbf{H}$ as the **tag** or **label** of the trapdoor. The paper also presented an efficient algorithm $\mathsf{GenTrap}(1^n, 1^m, q)$ that returns a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a trapdoor $\mathbf{T_A} = \mathbf{R}$ such that the distribution of $\mathbf{A}$ is negligibly close to the uniform distribution.

In some cases, the pair of vector $(\mathbf{s}, \mathbf{e})$ given an LWE problem instance $(\mathbf{A}, \mathbf{y} = \mathbf{As} + \mathbf{e})$ is also used as the secret trapdoor.

By using the "strong" trapdoor $\mathbf{R}$, one also can efficiently generate the "weak" trapdoor $\mathbf{T_A}$ or $\mathbf{s}$. Since in lattice-based PQC, these trapdoors $\mathbf{R}, \mathbf{T_A}, \mathbf{s}$ can be used as the secret keys of hierarchical parties in some hierarchical cryptosystems.

3.3. RLWE problem

There is a variant of the LWE problem that has been widely used in recent years called the Ring Learning With Error problem (RLWE). The primary benefit of using polynomial rings in RLWE offer more efficient and practical implementations of cryptographic schemes

Algorithm 2: $\mathsf{Gentrap}(n, m, q)$ generates a (pseudo) random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ together with its gadget trapdoor $\mathbf{T_A} = \mathbf{R}$ satisfies (25)

function GENTRAP((n, m, q))
- ▷ *Sample a uniformly random matrix* $\mathbf{A}_1$ ◁
- $\mathbf{A}_1 \leftarrow \mathbb{Z}_q^{n\times m_1}$
- ▷ *Choose a desired tag* $\mathbf{H}$ ◁
- $\mathbf{H} \leftarrow \mathbb{Z}_q^{n\times n}$ such that $\mathbf{H}$ is an invertible matrix. The algorithm can also set $\mathbf{H} = \mathbf{I}$.
- ▷ *Sample the gadget trapdoor* $\mathbf{R}$ *from a random distribution* $\mathcal{D}$ *over* $\mathbb{Z}^{(m-\omega)\times\omega}$ ◁
- $\mathbf{R} \leftarrow \mathcal{D}$.
- ▷ *Compute matrix* $\mathbf{A}$ ◁
- $\mathbf{A} = [\mathbf{A}_1|\mathbf{HG} - \mathbf{A}_1\mathbf{R}] \in \mathbb{Z}_q^{n\times m}$
- **return** $(\mathbf{A}, \mathbf{T_A} = \mathbf{R})$

with significantly smaller keys and signature sizes while maintaining strong security guarantees.

3.3.1. The Ring-LWE problem

Here, we define the Ring-LWE in a way similar to the LWE problem. More details about the technical aspects of RLWE can be found in Appendix A.

For integers $q \geq 2$, a power of two n, and an error distribution χ over short elements in $R_q = \mathbb{Z}_q[x]/(x^n + 1)$, the (average-case, search) Ring-LWE problem (RLWE) is defined as follows: Given $a_1, \ldots, a_m \in R_q$ sampled independently and uniformly at random together with $b_1, \ldots, b_m \in R_q$ where $b_i = a_i s + e_i \mod R_q$ for $s \in R_q$, and $e_i \leftarrow \chi$, find s.

3.3.2. Trapdoor algorithms in the ring setting

The trapdoor mechanism techniques from [11] can be generalised to the ring setting where there exists an efficient randomised algorithm $\mathsf{GenTrap}(1^n, 1^m, q)$ that returns $\mathbf{a} = (a_1, \ldots, a_m) \in R_q^m$ and a trapdoor $t_\mathbf{a}$ such that the distribution of $\mathbf{a}$ is negligibly (in n) close to the uniform distribution. Moreover, there is an efficient algorithm Invert and a universal constant C_T that, on input $\mathbf{a}$, $t_\mathbf{a}$ and $\mathbf{a} \cdot s + \mathbf{e}$ where $s \in R_q$ is arbitrary and $\|\mathbf{e}\| \leq q/(C_T\sqrt{n \log q})$, outputs s with overwhelming probability.

4. Trapdoor Claw Free Functions - TCFs

4.1. One-way function

Informally, one-way functions are functions that are "easy" to compute but "hard" to invert in a "computational sense." With these special properties, one-way functions are among the most fundamental tools for building classical and post-quantum cryptographic protocols, ranging from authentication, and signature to public key encryption schemes. More formally:

Definition 3 (One-way function)**.** *A function $f : \{0,1\}^* \to \{0,1\}^*$ is called one-way if two conditions hold:*

- ***Easy to compute****: Given x, one can compute $f(x)$ in polynomial time.*
- ***Hard to invert****: For every probabilistic polynomial-time algorithm $\mathcal{A}$, given $f(x)$, the probability that $\mathcal{A}$ finds a preimage of $f(x)$ is negligible, i.e.,*

$$\Pr[\mathcal{A}(f(x)) \in f^{-1}(f(x))] = \mathsf{negl}(\cdot). \tag{26}$$

Given a one-way function $f(x)$, one question is raised: *does the output of a one-way function, $y = f(x)$, truly hide all information of its preimage $x \in f^{-1}(f(x))$?*. The following example illustrates the fact that the output of a one-way function could reveal some information about a certain bit of a bit string $x = x_1 \ldots x_n$. Assuming that $g(x)$ is another one-way function, it follows that $f(x) = x_1||g(x)$ remains a one-way function since one

must still invert $g(x)$ to determine the whole x. The first bit x_1 is now exposed but does not reveal any information about the other bits of x.

For this reason, one needs a Boolean function (a Boolean predicate) $\mathcal{B} : \{0,1\}^* \to \{0,1\}$ that remains completely hidden given the output of a one-way function. This leads us to the definition of the hardcore bit property.

More formally, the hardcore bit property (or hard-core predicate) ensures that even with the information of $y = f(x)$, guessing $\mathcal{B}(x) \in \{0,1\}$ is as challenging as finding x.

Definition 4 (Hardcore bit property). *$\mathcal{B} : \{0,1\}^* \to \{0,1\}$ is a hardcore bit predicate of a one way function $f : \{0,1\}^* \to \{0,1\}^*$ if*

- *$\mathcal{B}$ is efficiently computable,*
- *It is "hard" to compute a bit $\mathcal{B}(x)$ of x given $f(x)$. Formally, for all adversaries $\mathcal{A}$*

$$\Pr_{x \leftarrow \{0,1\}^*}[\mathcal{A}(f(x)) = \mathcal{B}(x)] \leq \frac{1}{2} + \mathsf{negl}(\cdot).$$

4.2. Trapdoor Claw Free Functions

Trapdoor claw-free function (TCF), Fig. 6, is a two-to-one one-way function $f_{\mathcal{I}}(x)$ such that it is hard for a (quantum) polynomial-time adversary to simultaneously find the preimage pair (x_0, x_1) and $x_0 \neq x_1$ - *"a claw"*, given any image $y = f_{\mathcal{I}}(x_0) = f_{\mathcal{I}}(x_1)$ where $\mathcal{I}$ is some problem instance. Sometimes, TCFs can also be defined as a pair of functions $f_{\mathcal{I},0}(x)$, $f_{\mathcal{I},1}(x)$ that is both one-to-one and have the same image space $\mathcal{Y}$ and it is cryptographically hard to find two inputs mapping to the same output. One special property of TCFs is that given a problem instance $\mathcal{I}$ sampled together with a suitable trapdoor t_k, it is possible for one to find all the pre-images of y using the trapdoor t_k.

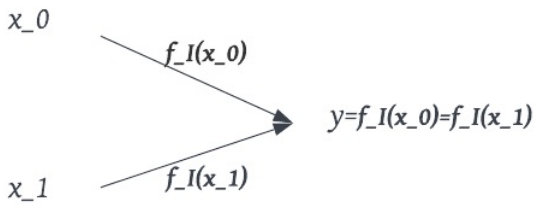


Figure 6: "A claw" of a TCFs $f_{\mathcal{I}}(x)$ is a pair of (x_0, x_1) such that $y = f_{\mathcal{I}}(x_0) = f_{\mathcal{I}}(x_1)$, $x_0 \neq x_1$.

More formally, we can define the trapdoor claw-free function as follows:

Definition 5 (Trapdoor Claw-Free Function (TCFs)). *Let $\mathcal{X}$, $\mathcal{Y}$ be finite sets, we require the family of functions $\mathcal{F}$ satisfies the following properties:*

- *For each public key k related to a problem instance $\mathcal{I}$, there are two functions $\{f_{\mathcal{I},b} : \mathcal{X} \to \mathcal{Y}\}_{b \in \{0,1\}}$ that are both injective and have the same range. These two functions are invertible given a suitable trapdoor t_k (i.e. t_k can be used to efficiently compute x given b and $y = f_{\mathcal{I},b}(x)$).*
- *The pair of functions should be claw-free, i.e. it is hard to find the pair $(x_0, x_1) \in \mathcal{X}^2$ such that*

$$y = f_{\mathcal{I},0}(x_0) = f_{\mathcal{I},1}(x_1) \text{ and } x_0 \neq x_1. \tag{27}$$

Note that a quantum device can not invert the function a find the claw (x_0, x_1), but it can hold these two preimages together with w in a superposition

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)\,|w\rangle\,. \tag{28}$$

This is obtained by first preparing a uniform superposition over all $x \in \mathcal{X}$ via the Hadamard transform

$$\frac{1}{\sqrt{\mathcal{X}}} \sum_{x \in \mathcal{X}} |x\rangle \,, \tag{29}$$

then evaluating $f_{\mathcal{I},b}(x)$ into an output register to yield

$$\frac{1}{\sqrt{\mathcal{X}}} \sum_{x \in \mathcal{X}} |x\rangle \, |f_{\mathcal{I},b}(x)\rangle \,. \tag{30}$$

If one measures the output register of (30) and obtains the result y, the state would collapse to the (28) state such that $w = f_{\mathcal{I},0}(x_0) = f_{\mathcal{I},1}(x_1)$. Measuring the (28) state in the standard basis (Z basis), one would obtain a random preimage as the measurement result, i.e. $m = x_0$ or $m = x_1$, but not both, since measuring a quantum state means collapsing that state to one possible outcome. Besides, measuring the (28) state in the X basis yields a measurement result m and a bit c that satisfies $m \cdot (x_0 \oplus x_1) = c$. One may observe that the measurement result using the X basis (m, c) can be used as the hardcore bit, as in the case of a one-way function. However, there is a subtle point in using TCFs in cryptographic protocols: not all TCFs possess the hardcore bit property, just as not all one-way functions successfully hide the information of the input. In all existing TCFs constructions, the only one that satisfies the hardcore bit property is the one utilising LWE problem, we will go into more detail and explaination in the next subsection.

Note that these special properties of TCFs could be used to construct a Proof of Quantumness protocol, the details of which will be discussed in section 5. However, a fully constructive TCFs that satisfies most cryptographic protocol requirements has not been explored yet. The paper [12] considers a "relaxed" version of the TCFs, referred to as **Noisy trapdoor claw-free functions (NTCFs)**. For more detail and technical information about NTCFs, please refer to the Appendix B.

4.3. TCFs from LWE

4.3.1. Definition

For a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ and vectors $\mathbf{s} \in \{0,1\}^n$, $\mathbf{e} \in \{0,1\}^m$, consider the LWE instance $\mathcal{I} = (\mathbf{A}, \mathbf{y} = \mathbf{A}\mathbf{s} + \mathbf{e})$ with the trapdoor $t_k = \{\mathbf{s}, \mathbf{e}\}$.

We can define the TCFs family $\{f_{\mathcal{I},b} : \{0,1\}^n \to \mathbb{Z}^m\}_{b \in \{0,1\}}$ by letting

$$f_{\mathcal{I},0}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{e}_0 \tag{31}$$

$$f_{\mathcal{I},1}(\mathbf{x}) = \mathbf{A}(\mathbf{x} + \mathbf{s}) + \mathbf{e}_0 + \mathbf{e}. \tag{32}$$

If $\mathbf{e}_0 = \mathbf{0}$, the the claw are related by

$$\mathbf{w} = f_{\mathcal{I},1}(\mathbf{x}_1) = f_{\mathcal{I},0}(\mathbf{x}_0) \tag{33}$$

$$\mathbf{x}_1 = \mathbf{x}_0 - \mathbf{s}. \tag{34}$$

However, we need to hide the information of $\mathbf{x}$, it is essential to sample $\mathbf{e}_0$ from a Gaussian distribution much wider than $\mathbf{e}$ such that it ensures that the distribution of $f_{\mathcal{I},1}(\mathbf{x}_1)$ and $f_{\mathcal{I},0}(\mathbf{x}_0)$ are statistically close. This is why the construction of the paper [12] used the term "noisy" for its TCFs. For simplicity of our review, we will not dive into the details of NTCFs in the main sections and assume that one can efficiently find the claw $(\mathbf{x}_0, \mathbf{x}_1)$ satisfies (34) using the trapdoor $t_k = \{\mathbf{s}, \mathbf{e}\}$.

4.3.2. Adaptive hardcore bit property for TCFs from LWE

Recall that in the TCFs from LWE, measuring the output register of the (30) state, obtaining the measurement outcome $\mathbf{w}$ and the superposition

$$\frac{1}{\sqrt{2}}(|\mathbf{x}_0\rangle + |\mathbf{x}_1\rangle)|\mathbf{w}\rangle. \tag{35}$$

Recall that when measuring the $\mathbf{x}$ register in the X basis, we obtain the measurement result $\mathbf{m} \in \{0,1\}^n$ and a bit $c \in \{0,1\}$ such that $\mathbf{m} \cdot (\mathbf{x}_0 \oplus \mathbf{x}_1) = c$. This equation provides some information about both preimages. Consider the pre-images $\mathbf{x}_0$ and $\mathbf{x}_1$ in (34), we have that $\mathbf{s}$ is some fixed secret determined by the LWE problem instance $\mathcal{I} = (\mathbf{A}, \mathbf{y} = \mathbf{As} + \mathbf{e})$. Hence, this equation can be rewritten as $\mathbf{m}' \cdot \mathbf{s} = c$. By repeating the entire procedure, one can have sufficient information about $\mathbf{m}'$ to recover $\mathbf{s}$ with high probability. Hence, we need the structure of the function $f_{\mathcal{I},b}$ to be a little more complicated so that even knowledge about $(\mathbf{m}, c)$, one cannot determine what the equation is about without additional knowledge of at least a preimage. As mentioned in the previous subsection, $\mathbf{m} \cdot (\mathbf{x}_0 \oplus \mathbf{x}_1) = c$ can be use as hardcore bit that ensures it is "hard" to compute a bit of $\mathbf{x}_0$ or $\mathbf{x}_1$ given $\mathbf{w}$. Moreover, if an adversary can simultaneously find this predicate and a preimage $\mathbf{x}_b$, it could find the whole claw $(\mathbf{x}_0, \mathbf{x}_1)$.

Moreover, we allow the adversary $\mathcal{A}$ to ***adaptively choose the string*** $\mathbf{m}'$. This also means that more power is given to the adversary. It is in this sense that the hardcore bit property is **adaptive**. Concretely, in the LWE-based construction of [12], the adversary $\mathcal{A}$ can adaptively choose the string $\mathbf{m}'$ after being given access to the LWE sample $(\mathbf{A}, \mathbf{As} + \mathbf{e})$.

Due to the *leakage resilience* property of the LWE problem, the information of the $\mathbf{s}$ remains secret. The idea is that instead of sampling a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ together with its trapdoor $\mathbf{s}, \mathbf{e}$, we replace $\mathbf{A}$ by a "lossy mode" matrix $\tilde{\mathbf{A}} \in \mathbb{Z}_q^{m \times n}$, where $\tilde{\mathbf{A}}$ is sampled from a special distribution so that $\tilde{\mathbf{A}} = \mathbf{BC} + \mathbf{E}$ for a skinny matrix $\mathbf{B}$, a wide matrix $\mathbf{C}$ and $\mathbf{E} \leftarrow \mathcal{D}_{\mathbb{Z}_q}^{m \times n}$. It holds that $\tilde{\mathbf{A}}$ is indistinguishable from a uniformly random $\mathbf{A}$. Now, when we multiply $\tilde{\mathbf{A}}$ by $\mathbf{s}$, the matrix $\mathbf{C}$ compresses the information of $\mathbf{s}$. To be more specific, provided the matrix $\mathbf{C}$ is a uniformly random matrix with sufficiently few rows, the distribution $(\mathbf{C}, \mathbf{Cs})$ for an arbitrary secret $\mathbf{s}$ does not reveal any parity of $\mathbf{s}$. Hence, even having the power to adaptively repeat the process of choosing $\mathbf{m}'$, an adversary can not find $\mathbf{s}$ (or information of both preimages $\mathbf{x}_0$ and $\mathbf{x}_1$). More information on the proof can be found in [12].

4.4. TCFs from RLWE

Two years later from the first definition of the TCFs from LWE was published, Brakerski continued his work in [13] and showed that using the ring version of the LWE problem yielded better results since RLWE-based primitives exhibit greater efficiency and yield smaller key sizes than their LWE-based counterparts, due to the ring dimension. This dimension plays a crucial role in determining the size of the challenge space. The larger the challenge space, the less the soundness error of the scheme has and the fewer times the underlying protocol has to repeat.

Let $R_q = \mathbb{Z}[x]/(x^n + 1)$, $\mathbf{a}$ is a vector of polynomials in R_q^m, $\mathbf{s} \in R_q$ and $\mathbf{e}$ is an error term sampled from a Discrete Gaussian distribution over R_q, consider the RLWE instance $\mathcal{I} = (\mathbf{a} \cdot s + \mathbf{e})$ with the trapdoor $t_k = (s, \mathbf{e})$.

The TCFs from RLWE problem can be defined as follows:

$$f_{\mathcal{I},0}(x) = \mathbf{a} \cdot x + \mathbf{e}_0, \tag{36}$$

$$f_{\mathcal{I},1}(x) = \mathbf{a} \cdot (x + s) + \mathbf{e}_0 + \mathbf{e}, \tag{37}$$

where $\mathbf{e}_0$ is sampled from a wider Gaussian distribution compared to $\mathbf{e}$.

Despite RLWE problems being considered more efficient than LWE problems, RLWE-based TCFs do not satisfy the adaptive hardcore bit property due to the absence of the lossiness argument in the RLWE problem.

5. Proof of Quantumness

5.1. Interactive proof

A proof serves as a means of persuading someone that a specific statement is true. We typically involve two parties when discussing proofs: *a prover* and *a verifier*. The prover aims to convince the verifier of the validity of a given statement, while the verifier's goal is to accept only accurate arguments. Before defining the interactive proof, we first recall that a language is simply a set of strings $L \subseteq \{0,1\}^*$. The definition of an interactive proof for a statement x in a language L is defined as follows:

Definition 6 (Interactive proof)**.** *An interactive proof system for a language L is a protocol between a computationally unrestricted prover $\mathcal{P}$ and a polynomial-time verifier $\mathcal{V}$ such that on input a statement x:*

- ***Completeness**: If $x \in L$, then an honest prover $\mathcal{P}$ that follows protocol should be able to convince the verifier $\mathcal{V}$ about the validity of the statement.*

$$\forall x \in L, \Pr[(\mathcal{V} \leftrightarrow \mathcal{P})(x) accepts\ ] = 1.$$

- ***Soundness** ϵ: If $x \notin L$, then no malicious prover $\mathcal{P}'$ should be able to convince $\mathcal{V}$ with a success probability greater than ϵ.*

$$\forall x \notin L, \forall \mathcal{P}', \Pr[(\mathcal{V} \leftrightarrow \mathcal{P}')(x) accepts] \leq \epsilon.$$

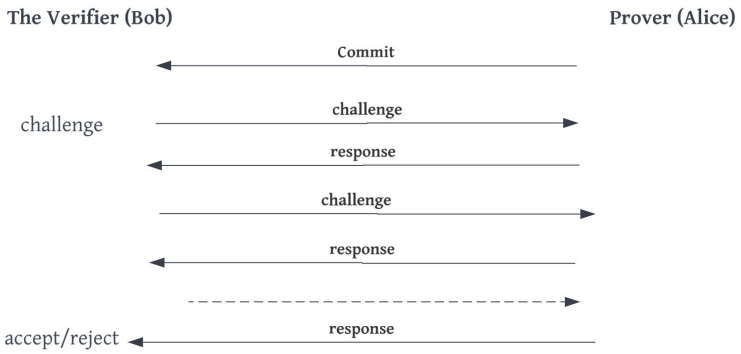


Figure 7: An interactive proof between a prover (Alice) and a verifier (Bob) typically has three main phases: commit, challenge, and response. The protocol may repeat the challenge-response phase multiple times to ensure the soundness property.

5.2. Proof of quantumness

In recent years, under the efforts of building quantum computers, the verification of quantum behaviour has become significantly crucial. This underscores the necessity for a protocol in which a classical party can efficiently certify the quantum advantage of an untrusted device - the Proof of Quantumness. An interactive proof of quantumness [12,14] is an interactive quantum verification protocol between two parties: a quantum prover $\mathcal{P}$ and a classical verifier $\mathcal{V}$. The verifier $\mathcal{V}$'s goal is to test the "quantumness" of the prover through an exchange of classical information. As mentioned in the end of the subsection 4.2, a common recent approach is to build the Interactive Proof of Quantumness protocol using TCFs.

5.3. Interactive Proof of quantumness from LWE-based TFCs

The basic idea of employing the TFCs from LWE to build the interactive proof of quantumness (IPQ) protocol is presented as follows:

1. The verifier samples an LWE instance $\mathcal{I} = (\mathbf{A}, \mathbf{y} = \mathbf{As} + \mathbf{e})$ together with a trapdoor $t_k = (\mathbf{s}, \mathbf{e})$, and sends $\mathcal{I}$ and the description of the TCFs $f_{\mathcal{I},b}(\cdot)$ to the prover.
2. The prover then:
   - prepares a uniform superposition over $\mathbf{x} \in \{0,1\}^n$

$$\frac{1}{\sqrt{2}^n} \sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle \,, \tag{38}$$

   - evaluates $f_{\mathcal{I},b}(\cdot)$ into an output register yields the state

$$\sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle \, |f_{\mathcal{I},b}(\mathbf{x})\rangle \,, \tag{39}$$

   - measures the output register to obtain $\mathbf{w} \in \mathcal{Y}$. The prover sends $\mathbf{w}$ to the verifier as the commitment for its quantum power.
3. The remaining state after measuring on the output register of the prover now is

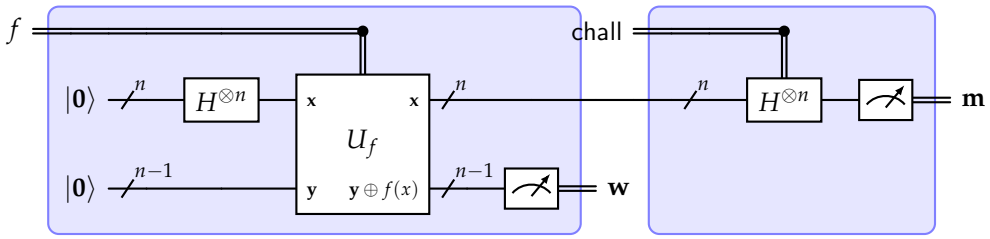
$$\frac{1}{\sqrt{2}}(|\mathbf{x}_0\rangle + |\mathbf{x}_1\rangle) \tag{40}$$

   where $\mathbf{x}_0$, $\mathbf{x}_1$ are two preimages of $\mathbf{w}$.
4. The verifier challenges by asking the prover to measure on either the standard basis (Z basis) ($\mathsf{chall} = 0$) or the Hadamard basis (X basis) ($\mathsf{chall} = 1$).
5. According to the verifier's challenge, the measurement outcome of the prover's state is:
   - Standard basis: either $\mathbf{m} = \mathbf{x}_0$ or $\mathbf{m} = \mathbf{x}_1$,
   - Hadamard basis : $(\mathbf{m} \in \{0,1\}^n), c \in \{0,1\}$ such that $\mathbf{m} \cdot (\mathbf{x}_0 \oplus \mathbf{x}_1) = c$.
6. The verifier checks these tests:
   - Standard basis: simply check that $f_{\mathcal{I},b}(\mathbf{m}) = \mathbf{w}$.
   - Hadamard basis: the verifier can recover $\mathbf{x}_0$, $\mathbf{x}_1$ from $\mathbf{y}$ using the trapdoor $t_k = (\mathbf{s}, \mathbf{e})$, and check whether the pair $(\mathbf{m}, c)$ satisfies $\mathbf{m} \cdot (\mathbf{x}_0 + \mathbf{x}_1) = c$.

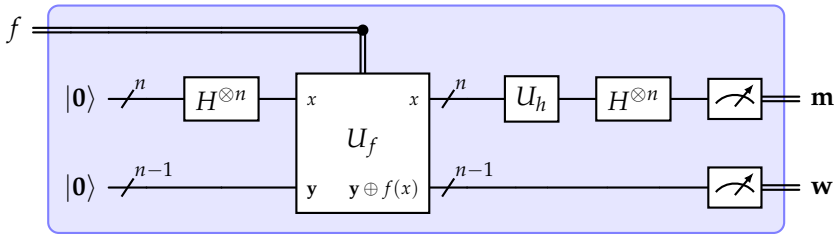
   The protocol proceeds for λ rounds where λ is the security parameter to ensure the soundness of the protocol. At the end of each round, the verifier samples a new problem instance and sends the new public key to the prover. In a recent paper by Zhu [14], for each of the verifier's possible choices, the experiment is repeated approximately 10^3 times to collect statistics.

Analysis. For the proof of quantumness protocol, the post-quantum cryptography hardness assumption here is employed in a different way, where we do not require the prover to break something that the classical machine cannot. The protocol is constructed in a way that the security is preserved even against both *classical* and *quantum* prover. We use the *rewinding* technique to distinguish between the classical and quantum cases. If a classical prover can cause the verifier to accept the statement, it can rewind and thus break the underlying hardness assumption which is computationally infeasible due to the hardness of the LWE problem. However, if the prover is "quantum," it can persist with the verifier without breaking the underlying cryptographic assumption while performing measurements that cannot be reversed due to the no-cloning theorem.

Note the adaptive hardcore bit property plays an important role in this IPQ construction. Suppose there exists a prover capable of successfully navigating both verifier challenges. Specifically, given the problem instance $\mathcal{I}$, this prover can generate a tuple $(\mathbf{w}, b, \mathbf{x}_b, (\mathbf{m}, c))$ with $b, c \in 0, 1$ that satisfies two checks with a probability significantly



(a) The IPQ protocol, where f is the TCFs chosen by the verifier. The prover executes the first round of the protocol, measuring the output register to obtain classical outcome $\mathbf{w}$. After receiving $\mathbf{w}$ from the prover the verifier communicates a random measurement basis Z or X by sending a bit ($\mathsf{chall} \in \{0,1\}$) for the prover to measure the $\mathbf{x}$ register (the random challenge). This process yields an outcome $\mathbf{m} \in \{0,1\}^n$. Knowing the secret trapdoor, the verifier can efficiently verify the prover's response.



(b) Non-interactive PQ protocol, replacing the randomly chosen challenge $\mathsf{chall} \in \{0,1\}$ by using a hash-function-based quantum random oracle.

Figure 8: Quantum circuits for proofs-of-quantumness protocols.
Blue boxes contain all quantum operations within the protocol with classical inputs and outputs.

higher than $1/2$, i.e., $f_{\mathcal{I},b}(\mathbf{x}_b) = \mathbf{w}$, $\mathbf{m} \cdot (\mathbf{x}_0 \oplus \mathbf{x}_1) = c$, and $\mathbf{m} \neq \mathbf{0}^n$. It is important to note that in this scenario, the prover does not discover the entire claw $(\mathbf{x}_0, \mathbf{x}_1)$ but rather identifies one preimage $\mathbf{x}_b$ along with a linear function of the other preimage $\mathbf{x}_{1-b}$. Therefore, the **hardcore bit** property is indispensable, asserting that it is computationally challenging to find even a single bit of $\mathbf{x}_{1-b}$—meaning the prover cannot determine a single bit of $\mathbf{x}_{1-b}$, let alone the entire claw.

5.4. Non-interactive proof of quantumness from the RLWE problem

Recall that up till now, the hardcore bit property has only been shown for TCFs based on the LWE problem. However, it is still possible to construct a **non-interactive proof of quantumness** protocol based on other hardness assumptions. [13] employs the **random oracle heuristic** as a tool to reduce the round complexity of the proof of quantumness protocol which also makes it possible to implement the protocol in a single round and eliminates the need for the hard-core bit property. The quantum device must evaluate the random oracle on a quantum superposition.

Assume we have a hash function $h(\cdot)$. In the unitary model of quantum computing, consider this means that there exists a unitary U_h that performs the mapping

$$U_h : |\mathbf{x}\rangle |\mathbf{y}\rangle \mapsto |\mathbf{x}\rangle |\mathbf{x} \oplus h(\mathbf{x})\rangle$$

on the basis states.

The proof of quantumness protocol is now parameterised by a hash function $h : \{0,1\}^n \to \{0,1\}$ as follows:

1. The verifier generates a problem instance $\mathcal{I}$ and trapdoor t_k and sends $\mathcal{I}$ to the prover.
2. The prover sends λ tuple $\{(\mathbf{w}_i, c_i, \mathbf{m}_i)\}_{i\in[\lambda]}$, where λ is the security parameter. The verifier initialises $\mathsf{count} = 0$ and performs the following checks:
   - It checks that all value $\{\mathbf{w}_i\}_{i\in[\lambda]}$ are distinct.

- It uses the trapdoor to compute $\mathbf{x}_{i,b}$ for each $i \in [\lambda]$ and $b \in \{0,1\}$. The verifier then checks $c_i = \mathbf{m}_i \cdot (\mathbf{x}_{i,0} \oplus \mathbf{x}_{i,1}) \oplus h(\mathbf{x}_{i,0}) \oplus h(\mathbf{x}_{i,1})$. If $(c_i, \mathbf{m}_i)$ satisfies this equation, it increases the value of count by 1.

3. If count $> 0.75\lambda$, the verifier accepts, else it rejects.

The second figure in Figure 8 represents the non-interactive proof of quantumness, which removes the interaction between the prover and the verifier during the challenge phase using the hash function $h(\cdot)$, as demonstrated above. Using the RLWE problem, we can significantly reduce the size of keys and the commitment of the construction.

Why can using the hash function can replace the hardcore bit property? Recall that in the LWE-based proof of quantumness construction [12], the prover applies the TCFs $f_{\mathcal{I},b}(\mathbf{x})$ on a uniform superposition of inputs and then measures the image register. It then obtains the value $\mathbf{w}$, which will be sent to the verifier. The prover now has a state that is a superposition over $\mathbf{x}_0$ and $\mathbf{x}_1$. The prover then has to measure this state according to the challenge of the verifier: measure using the standard basis or Hadamard basis. A malicious prover that can break the protocol must be able to answer both challenges by the verifier simultaneously, which violates the *hardcore bit property* and the hardness assumption. Using a one-bit hash function $h(\mathbf{x}) : \{0,1\}^n \to \{0,1\}$, [13] generates the superposition over $(0, \mathbf{x}_0, h(\mathbf{x}_0))$ and $(1, \mathbf{x}_1, h(\mathbf{x}_1))$. The prover is then asked to measure the resulting state on a Hadamard basis only and send the outcome to the verifier, including a bit c and the measurement outcome $\mathbf{m}$. It holds that $(c, \mathbf{m})$ satisfies the hardcore predicate $c = \mathbf{m} \cdot (\mathbf{x}_0 \oplus \mathbf{x}_1) \oplus h(\mathbf{x}_0) \oplus h(\mathbf{x}_1)$. The verifier employs the trapdoor to recover $(\mathbf{w}, \mathbf{x}_0, \mathbf{x}_1)$ and verifies whether the above equation is satisfied. The security of the protocol is upheld because an adversary cannot query the random oracle for both $\mathbf{x}_0$ and $\mathbf{x}_1$; thus, at least one value $h(\mathbf{x}_0)$ or $h(\mathbf{x}_1)$ remains random. This implies the malicious prover cannot compute $(c, \mathbf{x})$ satisfying the equation with a probability greater than $1/2$. Consequently, this also eliminates the need for the hardcore bit property.

6. Comparison on experimental results

This section gives the connection between result measurements in the original and experimental papers.

Table 1. Comparison between the Proof of Quantumness protocols.

Paper	Problem	Adaptive hardcore bit	Interactive	Using hash function	Gate count	Adversary type
[12]	LWE	✓	✓	✗	$n^2 \log^2 n$	classical
[14]	LWE	✓	✓	✗	$n^2 \log^2 n$	classical
[13]	RLWE	✗	✗	✓	$n \log^2 n$	quantum
[15]	RLWE	✗	✗	✓	$n \log^2 n$	quantum

7. Discussion

8. Conclusions

Author Contributions: For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used "Conceptualization, X.X. and Y.Y.; methodology, X.X.; software, X.X.; validation, X.X., Y.Y. and Z.Z.; formal analysis, X.X.; investigation, X.X.; resources, X.X.; data curation, X.X.; writing—original draft preparation, X.X.; writing—review and editing, X.X.; visualization, X.X.; supervision, X.X.; project administration, X.X.; funding acquisition, Y.Y. All authors have read and agreed to the published version of the manuscript.", please turn to the CRediT taxonomy for the term explanation. Authorship must be limited to those who have contributed substantially to the work reported.

Funding: Please add: "This research received no external funding" or "This research was funded by NAME OF FUNDER grant number XXX." and and "The APC was funded by XXX". Check

carefully that the details given are accurate and use the standard spelling of funding agency names at https://search.crossref.org/funding, any errors may affect your future funding.

Institutional Review Board Statement: In this section, you should add the Institutional Review Board Statement and approval number, if relevant to your study. You might choose to exclude this statement if the study did not require ethical approval. Please note that the Editorial Office might ask you for further information. Please add "The study was conducted in accordance with the Declaration of Helsinki, and approved by the Institutional Review Board (or Ethics Committee) of NAME OF INSTITUTE (protocol code XXX and date of approval)." for studies involving humans. OR "The animal study protocol was approved by the Institutional Review Board (or Ethics Committee) of NAME OF INSTITUTE (protocol code XXX and date of approval)." for studies involving animals. OR "Ethical review and approval were waived for this study due to REASON (please provide a detailed justification)." OR "Not applicable" for studies not involving humans or animals.

Informed Consent Statement: Any research article describing a study involving humans should contain this statement. Please add "Informed consent was obtained from all subjects involved in the study." OR "Patient consent was waived due to REASON (please provide a detailed justification)." OR "Not applicable" for studies not involving humans. You might also choose to exclude this statement if the study did not involve humans.

Written informed consent for publication must be obtained from participating patients who can be identified (including by the patients themselves). Please state "Written informed consent has been obtained from the patient(s) to publish this paper" if applicable.

Data Availability Statement: We encourage all authors of articles published in MDPI journals to share their research data. In this section, please provide details regarding where data supporting reported results can be found, including links to publicly archived datasets analyzed or generated during the study. Where no new data were created, or where data is unavailable due to privacy or ethical restrictions, a statement is still required. Suggested Data Availability Statements are available in section "MDPI Research Data Policies" at https://www.mdpi.com/ethics.

Acknowledgments: In this section you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments).

Conflicts of Interest: Declare conflicts of interest or state "The authors declare no conflicts of interest." Authors must identify and declare any personal circumstances or interest that may be perceived as inappropriately influencing the representation or interpretation of reported research results. Any role of the funders in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript; or in the decision to publish the results must be declared in this section. If there is no role, please state "The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results".

Abbreviations

The following abbreviations are used in this manuscript:

PQC	Post-quantum cryptography
RSA	Rivest-Shamir-Adleman algorithm
ECC	Elliptic Curve cryptography
QKD	Quantum Key Distribution
IPQ	Interactive proof of quantumness
SVP	Shortest vector problem
SIVP	Shortest independent vectors problem
CVP	Closest vector problem
SIS	Shortest integer solution
LWE	Learning with error
RLWE	Ring-Learning with error
TFCs	Trapdoor claw-free functions
NTCFs	Noisy trapdoor claw-free functions

Appendix A Ring Learning With Errors problem (RLWE)

Here, let's first recall some basic ideas of the RLWE problem.

Appendix A.0.1 The initial idea.

Let's recall that the SIS problem in Eq. (21) yields a very simple collision-resistant hash function that is provably secure if worst-case lattice problems are hard:

$$h_{\mathbf{A}}(\mathbf{s}) = \mathbf{A}\mathbf{s} \mod q \tag{A1}$$

where $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is uniformly random and $\mathbf{s} \in \{0,1\}^n$.

However, this is inefficient, since just reading the public description of $\mathbf{A}$ takes time roughly $mn \log q > n^2$. The goal is now to show a variant of $h_{\mathbf{A}}$ whose running time is roughly linear in n.

A trivial idea is taking some short, uniformly random seed r and then setting $\mathbf{A} = H(\mathsf{seed})$ for some suitable expanding function H. However, we need to describe H so that it can be efficiently used in cryptography constructions.

Let's first take the random seed to be $\ell = m/n$ uniformly random vectors $\mathbf{a}_1, \ldots, \mathbf{a}_\ell \in [\mathbf{q}]^{\mathbf{n}}$. Consider the "cyclic rotations" of $\mathbf{a}_i$, i.e. for a single vector $\mathbf{a} = (a_1, \ldots, a_n)^T \in \mathbb{Z}^n$, we define

$$\mathsf{Rot}(\mathbf{a}) = \begin{pmatrix} a_1 & a_n & \ldots & a_3 & a_2 \\ a_2 & a_n & \ldots & a_4 & a_3 \\ a_3 & a_n & \ldots & a_5 & a_4 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-2} & a_{n-3} & \ldots & a_n & a_{n-1} \\ a_{n-1} & a_{n-2} & \ldots & a_1 & a_n \\ a_n & a_{n-1} & \ldots & a_2 & a_1 \end{pmatrix} \in \mathbb{Z}^{n \times n}, \tag{A2}$$

where each column is a simple cyclic permutation of the previous column. $\mathbf{A}$ can now be rewritten as

$$\mathbf{A} = \begin{pmatrix} \mathsf{Rot}(\mathbf{a}_1) \\ \mathsf{Rot}(\mathbf{a}_2) \\ \vdots \\ \mathsf{Rot}(\mathbf{a}_\ell) \end{pmatrix} \in \mathbb{Z}^{m \times n}. \tag{A3}$$

Due to the property of $\mathsf{Rot}(\mathbf{a}$, we can write $\mathsf{Rot}(\mathbf{a}) = (\mathbf{a}), \mathbf{X}\mathbf{a}, \ldots, \mathbf{X}^{\mathbf{n-1}}\mathbf{a})$ where

$$\mathbf{X} = \begin{pmatrix} 0 & 0 & \ldots & 0 & 1 \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix} \in \{\mathbf{0}, \mathbf{1}\}^{\mathbf{n} \times \mathbf{n}} \tag{A4}$$

is the "cyclic shift" matrix.

Consider the set of all integer cyclic matrices, $\tilde{R} := \{\mathsf{Rot}(\mathbf{a}) : \mathbf{a} \in \mathbb{Z}^n\}$. Hence we have that $\tilde{R}$ is a lattice with rank n and basis $\{\mathbf{I_n}, \mathbf{X}, \mathbf{X^2}, \ldots, \mathbf{X^{n-1}}\}$. We have that $\tilde{R}$ is a commutative ring. We can instead think of $\mathsf{Rot}(\mathbf{a}) \in \tilde{R}$ as the corresponding polynomial $a \in R = \mathbb{Z}[x]/(x^n - 1)$. Hence, we can therefore identify the matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ with a tuple of ring

elements $(a_1, \ldots, a_\ell)^T \in R_q^\ell = R = \mathbb{Z}_q^\ell[x]/(x^n - 1)$. Similarly, $\mathbf{s}$ is a tuple of ring elements $(s_1, \ldots, s_\ell)^T \in R_{\{0,1\}}^\ell$. Therefore, $h_\mathbf{A}$ can be written as

$$h_a(s) = h_{a_1,\ldots,a_\ell}(s_1, \ldots, s_\ell) = a_1 s_1 + \cdots + s_\ell s_\ell \mod qR. \tag{A5}$$

The Ring-SIS problem, which is the analogue of SIS in this setting, can be defined as follows: For a ring R, integer modulus $q \geq 2$, and integer $\ell \geq 1$. Given $a_1, \ldots, a_\ell \in R_q$ sampled independently and uniformly at random. The search Ring-SIS problem asks to output $e_1, \ldots e_\ell \in R_{\{-1,0,1\}}$ not all zero such that $h_{a_1,\ldots,a_\ell}(e_1, \ldots, e_\ell) = a_1 e_1 + \cdots + a_\ell e_\ell = 0 \mod qR$.

However, for security reasons, there exist attacks on h_a over $\mathbb{Z}[x]/(x^n - 1)$ since $(x^n - 1)$ has a nontrivial factor over the integers. So, it is natural to try replacing $(x^n - 1)$ with an irreducible polynomial $p(x)$.

If n is a power of 2, we have that $x^n + 1$ is an irreducible polynomial and $R = \mathbb{Z}[x]/(x^n + 1)$ is an integral domain. From the matrix perspective mentioned above, this corresponds to taking

$$\mathsf{Rot}(\mathbf{a}) = \begin{pmatrix} a_1 & -a_n & \ldots & -a_3 & -a_2 \\ a_2 & a_n & \ldots & -a_4 & -a_3 \\ a_3 & a_n & \ldots & -a_5 & -a_4 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-2} & a_{n-3} & \ldots & -a_n & -a_{n-1} \\ a_{n-1} & a_{n-2} & \ldots & -a_1 & -a_n \\ a_n & a_{n-1} & \ldots & a_2 & a_1 \end{pmatrix} \in \mathbb{Z}^{n \times n}, \tag{A6}$$

and

$$\mathbf{X} = \begin{pmatrix} 0 & 0 & \ldots & 0 & -1 \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix} \in \{-\mathbf{1}, \mathbf{0}, \mathbf{1}\}^{\mathbf{n} \times \mathbf{n}}. \tag{A7}$$

Matrices of the form $\mathsf{Rot}(\mathbf{a})$ as above are occasionally called "*anti-cyclic*". Ring-SIS is hard over this ring, under a reasonable worst-case complexity assumption.

Appendix A.0.2 Ideal lattices and computational problems

Recall that a lattice is an additive subgroup of $\mathbb{Z}^n$,i.e., a subset of $\mathbb{Z}^n$ closed under addition and subtraction. An ideal $\mathcal{I} \subset R = \mathbb{Z}[x]/(x^n + 1)$ is an additive subgroup of R that is closed under multiplication by any ring element. Concretely, $\mathcal{I}$ is closed under addition and subtraction, and for any $y \in \mathcal{Y}$ and $r \in R$, we have that $ry \in \mathcal{I}$.

For the choice of ring, we can view $\mathcal{I}$ as a lattice by embedding R in $\mathbb{Z}^n$ via the trivial embedding that maps x_i to the unit vector $\mathbf{e}_i$. We have that $\mathcal{I} \subset \mathbb{Z}^n$ is a lattice such that $(y_1, \ldots, y_n)^T \in \mathcal{I}$ if and only if $(-y_n, y_1 \ldots, y_{n-1})^T \in \mathcal{I}$. Such lattices are sometimes called "anti-cyclic" lattices.

This view of ideals as lattices is useful since we can extend computational lattice problems to ideals. Concretely, for some fixed ring R, we can define the computational problems IdealSVP, IdealSIVP, GapIdealSVP, etc., similar to SVP, SIVP, etc. as the corresponding computational problems restricted to ideal lattices.

Appendix B Noisy Trapdoor Claw-free Functions (NTCFs)

In NTCFs, the range of functions $f_{k,0}$ and $f_{k,1}$ is represented by a distribution $\mathcal{D}$ over $\mathcal{Y}$, denoted as $\mathcal{D}_\mathcal{Y}$. In other words, each function returns a density rather than a point. Instead of considering the range of fk, b, we consider the support of the output densities,

$\mathsf{SUPP}(f_{\mathcal{I},b}(x))$. If y lies in this support, a party that holds the trapdoor t_k can use it to invert the function and find the preimages of y.

Definition A1 (Noisy Trapdoor Claw-Free family)**.** *Let λ be a security parameter. Let $\mathcal{X}$ and $\mathcal{Y}$ be finite sets. Let $\mathcal{K}_{\mathcal{F}}$ be a finite set of keys. A family of functions*

$$\mathcal{F} = \{f_{\mathcal{I},b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}\}_{b\in\{0,1\}} \tag{A8}$$

is called a noisy trapdoor claw-free (NTCF) family if the following conditions hold:

1. ***Efficient Function Generation.*** *There exists an efficient probabilistic algorithm $\mathsf{GEN}_{\mathcal{F}}$ which generates problem instance $\mathcal{I}$ together with a trapdoor t_k:*

$$\mathsf{GEN}_{\mathcal{F}}(1^{\lambda}) \to (\mathcal{I}, t_k). \tag{A9}$$

2. ***Trapdoor Injective Pair.***
   - *Trapdoor: There exists an efficient deterministic algorithm Invert $\mathsf{INV}_{\mathcal{F}}$ such that with overwhelming probability over the choice of $(\mathcal{I}, t_k)$, the following holds:*
   $$\text{for all } b \in \{0,1\}, \mathbf{x} \in \mathcal{X} \text{ and } y \in \mathsf{SUPP}(f_{\mathcal{I},b}(x)),\ \mathsf{INV}_{\mathcal{F}}(t_k, b, y) = x.$$
   - *Injective pair: There exists a perfect matching set for an instance $\mathcal{I}$ denoted $\mathcal{R}_{\mathcal{I}} \subseteq \mathcal{X} \times \mathcal{X}$ such that $f_{\mathcal{I},0}(x_0) = f_{\mathcal{I},0}(x_1)$ if and only if $(x_0, x_1) \in \mathcal{R}_{\mathcal{I}}$.*
3. ***Efficient Range Superposition.*** *For all keys $k \in \mathcal{K}_{\mathcal{F}}$ and $b \in \{0,1\}$ there exists a function $f'_{\mathcal{I},b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}$ such that the following hold:*
   - *For all $(x_0, x_1) \in \mathcal{R}_k$ and $y \in \mathsf{SUPP}(f'_{\mathcal{I},b})(x_b)$, $\mathsf{INV}_{\mathcal{F}}(t_k, b \oplus 1, y) = x_{x\oplus 1}$.*
   - *There exists an efficient deterministic procedure $\mathsf{CHK}_{\mathcal{F}}$ that given set of input $\mathcal{I}$, $b \in \{0,1\}$, $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, check whether y is an element of the support of f or not. This procedure returns 1 if $y \in \mathsf{SUPP}(f'_{\mathcal{I},b}(x))$ and 0 otherwise.*
   - *There exists an efficient procedure $\mathsf{SAMP}_{\mathcal{F}}$ that on input the problem instance $\mathcal{I}$ and $b \in \{0,1\}$, prepares the state*
   $$\frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x\in\mathcal{X}, y\in\mathcal{Y}} \sqrt{(f'_{\mathcal{I},b}(x)(y))}|x\rangle|y\rangle. \tag{A10}$$
4. ***Claw-free Property.*** *There no probabilistic polynomial time adversary $\mathcal{A}$ can find the "claw" (x_0, x_1, y). More formally, there exists a negligible function $\mathsf{negl}(\cdot)$ such that the following holds:*

$$\Pr[(x_0, x_1) \in \mathcal{R}_{\mathcal{I}} : (\mathcal{I}, t_k) \leftarrow \mathsf{GEN}_{\mathsf{F}}(1^{\lambda}), (x_0, x_1) \leftarrow \mathcal{A}(\mathcal{I})] \leq \mathsf{negl}(\lambda). \tag{A11}$$

Appendix B.1 NTCFs from LWE

[12] built a NTCFs family from the LWE problem as follows:

Definition A2 (NTCFs based on LWE [12])**.** *Let λ be a security parameter. Let $q \geq 2$ be a prime and $\ell, n, m \geq 1$ are polynomially bounded functions of λ, and B_L, B_V, B_P be positive integers such that the following conditions hold:*

- $n = \Omega(\ell \log q + \lambda)$,
- $m = \Omega(n \log q)$,
- $B_P = \dfrac{q}{2C_T\sqrt{mn \log q}}$, *for C_T is a universal constant in the $\mathsf{GenTrap}$ algorithm.*

1. ***Efficient Function Generation:*** *The $\mathsf{GEN}_{\mathcal{F}}(1^{\lambda})$ is constructed as follows:*
   - *Using the trapdoor mechanism $\mathsf{GenTrap}(1^n, 1^m, q)$ to return a matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ and a trapdoor $t_k = \mathbf{R}$, $m \geq \Omega(n \log q)$ such that the distribution of $\mathbf{A}$ is negligibly (in n) close to the uniform distribution on $\mathbb{Z}_q^{m\times n}$.*

- *Sample a uniformly random* $\mathbf{s} \leftarrow \{0,1\}^n$ *and a vector* $\mathbf{e} \leftarrow \mathbb{Z}_q^m$ *from the distribution* $D^m_{\mathbb{Z}_q, B_V}$.
- $\mathsf{GEN}_{\mathcal{F}}(1^\lambda)$ *returns*

$$(\mathcal{I}, t_k) = ((\mathbf{A}, \mathbf{As} + \mathbf{e}), t_k = \mathbf{R}. \tag{A12}$$

Note that the trapdoor $t_k = \mathbf{R}$ *can be referred to as a "strong" trapdoor since it can be used to reconstruct* $\mathbf{s}$ *and* $\mathbf{e}$. *In most existing cryptographic constructions using NTCFs, for simplicity, the trapdoor is usually set as* $t_k = (\mathbf{s}, \mathbf{e})$.

2. ***Trapdoor Injective Pair***

- *For any pair* $(\mathcal{I}, t_k) = ((\mathbf{A}, \mathbf{As} + \mathbf{e}), t_k = (\mathbf{s}, \mathbf{e})$, *a bit* $b \in \{0,1\}$, *the density function of* $f_{\mathcal{I},b}(\mathbf{x})$ *is defined as follows:*

$$\forall \mathbf{y} \in \mathcal{D}_{\mathcal{Y}}, (f_{\mathcal{I},b}(\mathbf{x}))(\mathbf{y}) = \mathcal{D}^m_{\mathbb{Z}_q, B_P}(\mathbf{y} - \mathbf{A}(\mathbf{x} + b\mathbf{s})) \tag{A13}$$

where $\mathbf{y} \in \mathbb{Z}_q^m$. *The support of the density function* $f_{\mathcal{I},b}(\mathbf{x})$ *is*

$$\mathsf{SUPP}(f_{\mathcal{I},0}(\mathbf{x})) = \{\mathbf{Ax} + \mathbf{e}_0 : \|\mathbf{e}_0\| \leq B_P\sqrt{m}\}. \tag{A14}$$

$$\mathsf{SUPP}(f_{\mathcal{I},0}(\mathbf{x})) = \{\mathbf{A}(\mathbf{x} + \mathbf{s}) + \mathbf{e}_0 : \|\mathbf{e}_0\| \leq B_P\sqrt{m}\}. \tag{A15}$$

The inversion algorithm $\mathsf{INV}_{\mathcal{F}} \equiv \mathsf{INVERT}(t_k, \mathbf{A}, \mathbf{y})$ *finds* $(\mathbf{s}_0, \mathbf{e}_0)$ *such that* $\mathbf{y} = \mathbf{As}_0 + \mathbf{e}_0$ *and returns* $\mathbf{s}_0 - b \cdot \mathbf{s} \in \mathcal{X}$. *Due to the parameter selections and the choice of* $\mathbf{e}_0$, *we have that* $\mathsf{INV}_{\mathcal{F}}$ *returns a preimage of the NTCFs.*
We have that, for all $\mathbf{y} \in \mathsf{SUPP}(f_{\mathcal{I},b}(\mathbf{x}))$, $\mathsf{INVERT}(t_k, \mathbf{A}, \mathbf{y}) = \mathbf{x} + b\mathbf{s}$. *Hence, it follows that* $\mathsf{INV}_{\mathcal{F}}(t_k, b, \mathbf{y}) = \mathbf{x}$.

- *Injective pair: Let* $\mathcal{I} = (\mathbf{A}, \mathbf{As} + \mathbf{e})$. *From the construction, it follows that* $f_{\mathcal{I},0}(\mathbf{x}_0) = f_{\mathcal{I},0}(\mathbf{x}_1)$ *if and only if* $\mathbf{x}_1 = \mathbf{x}_0 + \mathbf{s}$. *Also, the matching set is defined as* $\mathcal{R}_{\mathcal{I}} = \{(\mathbf{x}, \mathbf{x} + \mathbf{s}) | \mathbf{x} \in \mathbb{Z}_q^n\}$.

3. ***Efficient Range Superposition***

On input $(\mathbf{A}, \mathbf{As} + \mathbf{e})$, $b \in \{0,1\}$, $\mathbf{x} \in \mathbb{Z}_q^n$ *and* $\mathbf{y} \in \mathbb{Z}_q^m$, *the procedure* $\mathsf{CHK}_{\mathcal{F}}$ *operates as follows.*

- *If* $b = 0$, *it computes* $\mathbf{e}' = \mathbf{y} - \mathbf{Ax}$. *If* $\|\mathbf{e}'\| \leq B_P\sqrt{m}$, *the procedure outputs* 1, *else output* 0.
- *If* $b = 1$, *it computes* $\mathbf{e}' = \mathbf{y} - \mathbf{Ax} - (\mathbf{As} + \mathbf{e})$. *If* $\|\mathbf{e}'\| \leq B_P\sqrt{m}$, *the procedure outputs* 1, *else output* 0.

The procedure $\mathbf{SAMP}_{\mathcal{F}}$ *prepares the quantum state as per the following steps:*

- *Create the following superposition:*

$$\sum_{\mathbf{e}_0 \in \mathbf{Z}_q^m} \sqrt{\mathcal{D}_{\mathbb{Z}_q^m, B_P}(\mathbf{e}_0)} \, |\mathbf{e}_0\rangle \, . \tag{A16}$$

- *Create a uniform superposition over* $\mathbf{x} \in \mathbf{Z}_q^n$ *to obtain the state:*

$$\frac{1}{\sqrt{q^m}} \sum_{\mathbf{x} \in \mathbb{Z}_q^n, \mathbf{e}_0 \in \mathbf{Z}_q^m} \sqrt{\mathcal{D}_{\mathbb{Z}_q^m, B_P}(\mathbf{e}_0)} \, |\mathbf{x}\rangle \, |\mathbf{e}_0\rangle \, . \tag{A17}$$

- *Now, using the problem instance* $\mathcal{I} = (\mathbf{A}, \mathbf{As} + \mathbf{e})$ *and the bit* b *to create the state*

$$\frac{1}{\sqrt{q^m}} \sum_{\mathbf{x} \in \mathbb{Z}_q^n, \mathbf{e}_0 \in \mathbf{Z}_q^m} \sqrt{\mathcal{D}_{\mathbb{Z}_q^m, B_P}(\mathbf{e}_0)} \, |\mathbf{x}\rangle \, |\mathbf{e}_0\rangle \, |\mathbf{Ax} + \mathbf{e}_0 + b \cdot (\mathbf{As} + \mathbf{e})\rangle \tag{A18}$$

- *Since* $\mathbf{e}_0$ *can be computed from* x *and the last register,* b *and the problem instance* $\mathcal{I}$*, one can uncompute the register containing* $\mathbf{e}_0$ *and obtain the state*

$$\frac{1}{\sqrt{q^m}} \sum_{\mathbf{x}\in\mathbb{Z}_q^n, \mathbf{e}_0\in\mathbf{Z}_q^m} \sqrt{\mathcal{D}_{\mathbb{Z}_q^m, B_P}(\mathbf{e}_0)}\, |\mathbf{x}\rangle\, |\mathbf{A}\mathbf{x} + \mathbf{e}_0 + b\cdot(\mathbf{A}\mathbf{s}+\mathbf{e})\rangle \tag{A19}$$

$$= \frac{1}{\sqrt{q^m}} \sum_{\mathbf{x}\in\mathbb{Z}_q^n, \mathbf{e}_0\in\mathbf{Z}_q^m} \sqrt{(f'_{\mathcal{I},b})(\mathbf{w})}\, |\mathbf{x}\rangle\, |\mathbf{y}\rangle\,. \tag{A20}$$

4. ***Claw-free Property*** *Suppose there exists an adversary* $\mathcal{A}$*, on input* $\mathbf{y} = \mathbf{A}\mathbf{s} + \mathbf{e}$ *can output the pair* $(\mathbf{x}_0, \mathbf{x}_1) \in \mathcal{R}_k$*. This adversary can be used to break the* LWE *problem since* $\mathbf{x}_1 - \mathbf{x}_0 = \mathbf{s}$*.*

Appendix B.2 NTCFs from RLWE

Definition A3 (TCFs from RLWE problem [13])**.** *Let* λ *be the security parameter,* $n = 2^{\log\lambda}$*. Set*

- *Ring* $R = \mathbb{Z}[x]/(x^n+1)$*.*
- *Modulus* $q = \mathsf{poly}(n)$*,* $R_q = R/qR$*.*
- $m = \Omega(\log q)$*: determines the dimension of range space*
- $\chi = D_{\mathbb{Z}_q^n, B_V}$*: the noise distribution.*
- B_P*: the noise bound for function evaluation satisfies the constraints:*
  - $B_P \geq \Omega(nmB_V)$*,*
  - $2B_P\sqrt{nm} \leq q/(C_T\sqrt{n\log q})$ *for some constant* C_T*.*

  The domain is $R_q = \mathbb{Z}_q[x]/(x^n+1)$ *and the range is* R_q^m*.*
  The problem instance $\mathcal{I} = (\mathbf{a}, \mathbf{a}\cdot s + \mathbf{e})$*, where* $s \in R_q$*,* $a_i, e_i \in R_q$ *for all* $i \in [m]$*,* $\mathbf{a} = [a_1, \ldots, a_m]^T$*,* $\mathbf{e} = [e_1, \ldots, e_m]^T$*.*
  For $b \in \{0,1\}$*, the density function* $f_{\mathcal{I},b}(x)$ *is defined as follows:*

$$\forall \mathbf{y} \in R_q^m, (f_{\mathcal{I},b}(x))(\mathbf{y}) = D_{\mathbb{Z}^{nm}, B_P}(\mathbf{y} - \mathbf{a}\cdot x - b\cdot\mathbf{a}\cdot s), \tag{A21}$$

  where $\mathbf{y} = [y_1, \ldots, y_m]^T$*, and each* y_i *can be represented as element in* $\mathbb{Z}_q^n$*; similarly for* $\mathbf{a}\cdot x$ *and* $\mathbf{a}\cdot s$*.*

1. ***Efficient Key Generation***
   - *The algorithm* $\mathsf{GenTrap}(1^n, 1^m, q)$ *returns* $\mathbf{a} \in R_q^m$ *and a trapdoor* t_k *such that the distribution of* $\mathbf{a}$ *is negligibly (in* n*) close to the uniform distribution on* R_q^m*.*
   - *Sample a uniformly random* $\mathbf{s} \leftarrow R_q$ *and a vector* $\mathbf{e} \leftarrow \chi^m$*.*
   - *Compute* $\mathbf{a}\cdot s + \mathbf{e}$*.*
   - $\mathsf{GEN}_{\mathcal{F}}(1^\lambda)$ *returns*

$$(\mathcal{I}, t_k) = ((\mathbf{a}, \mathbf{a}\cdot s + \mathbf{e}), (s, \mathbf{e})). \tag{A22}$$

2. ***Trapdoor Injective Pair***
   - *The support of the density function of* $f_{\mathcal{I},b}(x)$ *is*

$$\mathsf{SUPP}(f_{\mathcal{I},b}(x)) = \{\mathbf{y} \in R_q^m : \|\mathbf{y} - \mathbf{a}\cdot x - b\cdot\mathbf{a}\cdot s\| \leq B_P\sqrt{nm}\}. \tag{A23}$$

     For $\mathcal{I} = (\mathbf{a}, \mathbf{a}\cdot s + \mathbf{e})$*, the inversion algorithm* $\mathsf{INV}_{\mathcal{F}}(t_k, b, \mathbf{y} \in R_q^m) \equiv \mathsf{INVERT}(t_k, \mathbf{a}, \mathbf{y}) - b\cdot s$*. We have that, for all* $\mathbf{y} \in \mathsf{SUPP}(f_{\mathcal{I},b}(x))$*,* $\mathsf{INVERT}(t_k, \mathbf{a}\, y) = x + b\cdot s$*. Hence, it follows that* $\mathsf{INV}_{\mathcal{F}}(t_k, b, \mathbf{y}) = x$*.*
   - *Injective pair: From the construction, it follows that* $f_{\mathcal{I},0}(x_0) = f_{\mathcal{I},0}(x_1)$ *if and only if* $x_1 = x_0 + s$*. Denote the matching set* $\mathcal{R}_{\mathcal{I}} = \{(x, x+s) | x \in R_q\}$*.*
3. ***Efficient Range Superposition*** *Perform similar operations as NTCFs from LWE, on input* $(\mathbf{a}, \mathbf{a}\cdot s + \mathbf{e})$*,* $b \in \{0,1\}$*,* $x \in R^q$ *and* $\mathbf{w} \in R_q^m$*, the procedure* $\mathsf{CHK}_{\mathcal{F}_{\mathsf{RLWE}}}$ *outputs* 1 *if*

$\mathbf{y} \in \mathsf{SUPP}(f'_{\mathcal{I},b}(x))$ *and* 0 *otherwise. Note that the condition that* $\mathsf{CHK}_{\mathcal{F}_{\mathsf{RLWE}}}$ *needs to check is* $\|\mathbf{w} - \mathbf{a} \cdot x - b \cdot \mathbf{y}\| \leq B + P\sqrt{n \cdot m}$. *The procedure* $\mathbf{SAMP}_{\mathcal{F}}$ *prepares the quantum states similar to the one built on LWE in the previous section.*

4. ***Claw-free Property*** *Suppose there exists an adversary* $\mathcal{A}$, *on input* $\mathbf{y} = \mathbf{a} \cdot s + \mathbf{e}$ *can output the pair* $(x_0, x_1) \in \mathcal{R}_k$. *This adversary can be used to break the* RLWE *problem since* $x_1 - x_0 = s$.

References

1. Shor, P. Algorithms for quantum computation: discrete logarithms and factoring. In Proceedings of the Proceedings 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 124–134. https://doi.org/10.1109/SFCS.1994.365700.
2. Bennett, C.H.; Brassard, G. Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science* **2014**, *560*, 7–11. Theoretical Aspects of Quantum Cryptography – celebrating 30 years of BB84, https://doi.org/https://doi.org/10.1016/j.tcs.2014.05.025.
3. Ekert, A.K. Quantum cryptography based on Bell's theorem. *Phys. Rev. Lett.* **1991**, *67*, 661–663. https://doi.org/10.1103/PhysRevLett.67.661.
4. Goldwasser, S. On the Hardness of the Shortest Vector Problem **1999**.
5. Stephens-Davidowitz, N. Dimension-preserving reductions between lattice problems.
6. Peikert, C. A Decade of Lattice Cryptography. *Found. Trends Theor. Comput. Sci.* **2016**, *10*, 283–424. https://doi.org/10.1561/0400000074.
7. Regev, O. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. *J. ACM* **2009**, *56*. https://doi.org/10.1145/1568318.1568324.
8. Peikert, C.; Regev, O.; Stephens-Davidowitz, N. Pseudorandomness of Ring-LWE for Any Ring and Modulus. In Proceedings of the Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, New York, NY, USA, 2017; STOC 2017, p. 461–473. https://doi.org/10.1145/3055399.3055489.
9. Babai, L. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica* **1986**, *6*, 1–13. https://doi.org/10.1007/BF02579403.
10. Alwen, J.; Peikert, C. Generating shorter bases for hard random lattices **2009**. pp. 75–86.
11. Micciancio, D.; Peikert, C. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In Proceedings of the Advances in Cryptology – EUROCRYPT 2012; Pointcheval, D.; Johansson, T., Eds., Berlin, Heidelberg, 2012; pp. 700–718.
12. Brakerski, Z.; Christiano, P.; Mahadev, U.; Vazirani, U.; Vidick, T. A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device. *J. ACM* **2018**, *68*. https://doi.org/10.1145/3441309.
13. Brakerski, Z.; Koppula, V.; Vazirani, U.; Vidick, T. Simpler Proofs of Quantumness, 2020, [arXiv:quant-ph/2005.04826].
14. Zhu, D.; Kahanamoku-Meyer, G.; Lewis, L.; Noel, C.; Katz, O.; Harraz, B.; Wang, Q.; Risinger, A.; Feng, L.; Biswas, D.; et al. Interactive cryptographic proofs of quantumness using mid-circuit measurements. *Nature Physics* **2023**, *19*, 1–7. https://doi.org/10.1038/s41567-023-02162-9.
15. Lewis, L.; Zhu, D.; Gheorghiu, A.; Noel, C.; Katz, O.; Harraz, B.; Wang, Q.; Risinger, A.; Feng, L.; Biswas, D.; et al. Experimental implementation of an efficient test of quantumness. *Phys. Rev. A* **2024**, *109*, 012610. https://doi.org/10.1103/PhysRevA.109.012610.