Heuristic Ideal Obfuscation Scheme based on LWE Problem, its Variants and Quantum Oracle

Zhuang Shan(单壮)[1], Leyou Zhang(张乐友)[1,*], Qing Wu(吴青)[2]

March 4, 2024

Abstract

This paper presents a heuristic ideal obfuscation scheme based on the learning problem, which differs from that of Jain, Lin, and Luo [JLLW23]. The paper adopts a method similar to Brakerski, Dottling, Garg, and Malavolta [BDGM22, BDGM20] for constructing iO. It first introduces a variant of the LWR problem and proves its pseudorandomness. Based on the variant of the LWR problem, it constructs LHE, then combines it with sFHE constructed from the LWE problem to further construct the ideal obfuscation scheme. In comparison to the approach by Jain et al., this paper is relatively more specific. Additionally, the paper incorporates the quantum random oracle construction by Jelle Don, et al.[DFMS22] to provide a more concrete quantum random oracle used in the proposed obfuscation scheme.

Keywords: Ideal obfuscation; Split fully homomorphic encryption scheme; Quantum oracle; Learning with error problem; Learning with rounding.
2020 MSC: 47J20; 90C25; 90C30; 90C52.

1 Introduction

Virtual Black Box Obfuscation, or VBB. In 2000, Hada[Had00] first introduced the definition of virtual black box obfuscation, which is essential for embedding a circuit C into an opaque black box that cannot be opened. By inputting x into one end of the black box, the other end automatically outputs $C(x)$. Since the entire circuit is hidden inside the black box, no specific information about the construction of C can be obtained. The only action we can take is to provide input and observe the output on the other side.

VBB functions like a virtualized black box, where a circuit C obfuscated by VBB prevents us from obtaining any information related to its construction through the obfuscated output.

[1] School of Mathematics and Statistics, Xidian University, Xi'an 710126, China; arcsec30@163.com

[2] School of Automation, Xi'an University of Posts and Telecommunications, Xi'an 710121, China

The only action possible is to provide input x and compute $C(x)$[Yue20]. Unfortunately, Barak et al.[BGI+01] have proven that virtual black box obfuscation does not exist.

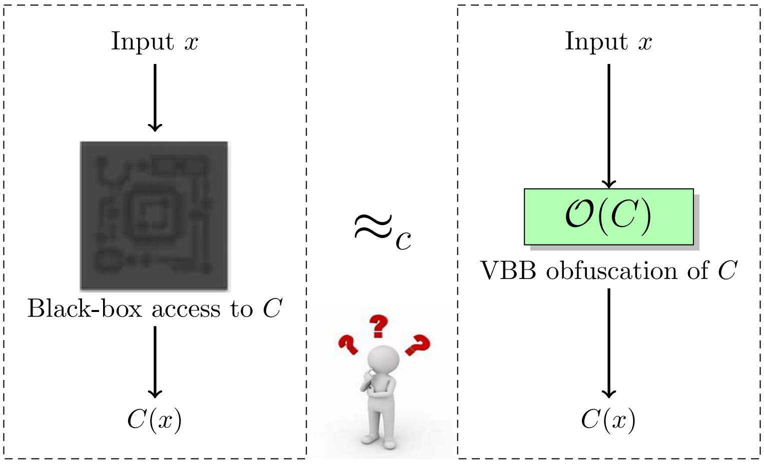

Figure 1: VBB obfuscation

Indistinguishability Obfuscation, or $i\mathcal{O}$. In 2001, while Barak et al. proved the nonexistence of virtual black box obfuscation, they also presented a new definition for obfuscation: to obfuscate two circuits C_1 and C_2 such that the obfuscated circuits have the same functionality and an adversary cannot distinguish between the two circuits. This is known as indistinguishable obfuscation.

In 2013, Garg et al. introduced indistinguishable obfuscation based on multilinear maps [GGH+13b] and applied it to functional encryption. It is noteworthy that multilinear maps were also proposed by Garg et al. [GGH13a]. Subsequently, significant work using program obfuscation(e.g., [BZ17, GGHR14, SW21]) has shown that most interesting cryptographic applications can be realized using iO (and one-way functions).

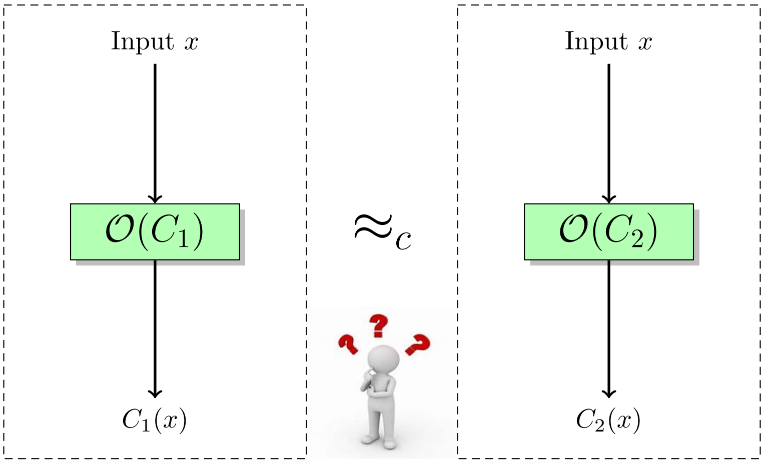

Figure 2: Indistinguishability Obfuscation ($i\mathcal{O}$)

Due to its importance, many scholars have begun to focus on researching how to construct indistinguishable obfuscation. One construction method is based on new multilinear maps, which extends its applicability to a wider range [GGH13a, CLT13, GGH15]. However, in 2016, Hu and Jia [HJ16] broke the indistinguishable obfuscation based on multilinear maps proposed by Garg et al. [GGH13a]. In the same year, Miles, Sahai, and Zhandry [MSZ16] partially broke another indistinguishable obfuscation scheme by Garg et al. [GGH$^+$13b]. Since 2015, the field of obfuscation with multilinear pairings has entered a cycle where proposed schemes are quickly broken, leading to improvements based on the attacks, only to be broken again shortly thereafter.

Recently, Bitansky and Vaikuntanathan [BV18] and Ananth and Jain [AJ15] have independently proven through different methods that when Compact FE (Functional Encryption with compact ciphertexts) exists, then indistinguishable obfuscation can be achieved. Based on these results, the current construction methods for indistinguishable obfuscation mainly fall into two categories, namely:

1. The first approach is to restrict the depth of multilinear maps to achieve indistinguishable obfuscation. For example, in 2016, Lin restricted the depth to 5 layers [Lin17], and later with Tessaro restricted it to 3 layers [LT17]. In 2020, Jain, Lin, and Sahai [JLS21] successfully constructed indistinguishable obfuscation based on bilinear pairings, LWE (Learning With Errors), LPN (Learning Parity with Noise), and sPFG (sub-exponential Pseudorandom Function Generator). This means that we can now achieve indistinguishable obfuscation based on known constructions.

2. The second approach is to achieve indistinguishable obfuscation through splitting fully homomorphic encryption. For example, Brakerski, Dottling, Garg, and Malavolta [BDGM22, BDGM20] combined fully homomorphic encryption (FHE) with leveled homomorphic encryption (LHE) (Damgård-Jurik). By cleverly leveraging circular-security

assumptions, they enable ciphertexts to circulate between the two encryption systems, ultimately constructing indistinguishable obfuscation.

Ideal Obfuscation. In 2023, Jain, Lin, and Luo introduced a new concept called "ideal obfuscation [JLLW23]." This concept is a refinement of Jain's work on indistinguishable obfuscation.

$$\begin{tikzcd}
\text{Collision-resistant hash functions} \arrow[r, "\text{Idealization}"] & \text{Pseudorandom Oracle Model} \\
i\mathcal{O} \arrow[r, "\text{Idealization}"'] & \text{Ideal Model} \arrow[u, "RO(\cdot) = Obf(PRF(k, \cdot))"]
\end{tikzcd}$$

Figure 3: The association between Ideal Obfuscation and $i\mathcal{O}$ (from [Luo23])

The main work of this paper is to propose new lattice-based hardness problems and summarize the research progress of indistinguishable obfuscation in the past five years, particularly focusing on the schemes by Brakerski, Dottling, Garg, and Malavolta [BDGM22, BDGM20]. We introduce two variants of the LWR problem and use one of them to construct an LHE scheme to replace the Damgård-Jurik scheme by Brakerski et al., making it resistant to quantum attacks.

1.1 Our work

First, give the definition of learning with rounding problem [BPR12].

Learning with Rounding. The learning with rounding problem is a variant of the learning with errors problem. In 2012, Banerjee, Peikert, and Rosen first proposed this problem, which is primarily used to construct pseudorandom functions and deterministic encryption [XXZ12]. Let $\lfloor \cdot \rceil_p : \mathbb{Z}_q \to \mathbb{Z}_p$ be the rounding function, where $p < q$. Then the learning with rounding problem involves recovering s from $\lfloor As \rceil_p$. Clearly, Banerjee, Peikert, and Rosen utilized the 'error' introduced by the modulo p in the learning with rounding problem as a replacement for the perturbation e in the learning with errors problem.

Banerjee and others proposed the learning with rounding problem and then provided a very elegant and concise reduction to the case where the modulo is small enough for e and the learning with errors problem. The purpose of this approach is to ensure that $\lfloor \langle a, s \rangle \rceil_p$ and $\lfloor \langle a, s \rangle + e \rceil_p$ are close enough in probability. This deliberate parameter selection aims to make $\Pr[\lfloor \langle a, s \rangle \rceil_p \neq \lfloor \langle a, s \rangle + e \rceil_p] \leq \text{negl}$. Consequently, the amount of information output by the learning with rounding problem is smaller than that of the learning with errors problem, resulting in a reduction in the difficulty of the problem.

In 2013, Alwen and others used information entropy theory to reduce the learning with rounding problem to the learning with errors problem, indirectly leading to a reduction to the closest vector problem in lattices. More specifically, Alwen et al. demonstrated the existence of a ”lossy” sampling algorithm denoted as $A' \leftarrow \text{Lossy}()$, with the form $A' = BC + F$, where $B \leftarrow \mathbb{Z}_q^{m\times n'}$, $C \leftarrow \mathbb{Z}_q^{n'\times n}$, and $F \in \chi^{m\times n}$. This enables:

- Based on the learning with errors problem, $A' = BC + F$ is indistinguishable from $A \leftarrow \mathbb{Z}_q^{m\times n}$.
- For $A' = BC+F$, $\lfloor A's \rfloor_p$ will not lose too much information about s with high probability.

Based on the learning with rounding problem, provide two variants of the indistinguishability theorem, as follows:

Theorem 1 (Informal). *Let p, q be prime numbers, $A \in \mathbb{Z}_q^{m\times n}$, $s \in \mathbb{Z}_q^n$, $u \in \mathbb{Z}_p^m$. If it is difficult to distinguish between $(A, \lfloor As \rfloor_p)$ and $(A, \lfloor u \rfloor_p)$, then for $a \in_R \mathbb{Z}_q$ (or $a \in_R \mathbb{Z}_q^{m\times n}$), we have*

$$(\odot_q(A,a), \lfloor u \rfloor_p) \approx_c \left(\odot_q(A,a), \lfloor \odot_q(A,a)\cdot s \rfloor_p\right).$$

Theorem 2 (Informal). *Let p, q be prime numbers, $A \in \mathbb{Z}_q^{m\times n}$, $s \in \mathbb{Z}_q^n$, $u \in \mathbb{Z}_q^m$. If it is difficult to distinguish between $(A, \lfloor As \rfloor_p)$ and $(A, \lfloor u \rfloor_p)$, then for $a, b \in_R \mathbb{Z}_q$ (or $b \in_R \mathbb{Z}_q$, $a \in_R \mathbb{Z}_q^{m\times n}$), we have*

$$(\odot_q(A,a), \lfloor u \rfloor_p) \approx_c \left(\odot_q(A,a), \lfloor \odot_q(A,ab)\cdot s \rfloor_p\right).$$

A variant problem is used to construct public key encryption, and a second variant problem is used to construct identity-based encryption, demonstrating the semantic security of both. Building on the work of Brakerski, Dottling, Garg, and Malavolta [BDGM22, BDGM20], provide a heuristic notion of indistinguishability obfuscation. Leveraging the second variant problem (referred to for ease of exposition as the delta variant of learning with rounding, LWR.DV), we construct a linear homomorphic scheme. This LWR.DV-based linear homomorphic scheme theoretically possesses properties resistant to quantum attacks.

Theorem 3 (Informal). *Assuming the sub-exponential hardness of the learning with error problem and the learning with rounding problem, there exists a sub-exponentially secure split fully homomorphic encryption scheme. Consequently, there exists an ideal obfuscation that can be applied to any circuit.*

Conceptual Ideal Obfuscation Scheme. Next, present a conceptual split FHE scheme (ideal obfuscation scheme), which is based on three main techniques: (i) linear decryption multiplication in standard FHE schemes (which can be instantiated in almost all LWE structures [BDGM22, BDGM20]), (ii) short decryption gadgets for linear homomorphic encryption schemes (such as the scheme in this paper, based on the LWR.DV problem), and (iii) encrypted hash functions (used for a part of the linear homomorphic encryption scheme). The security of this scheme can be based on a new conjecture regarding the interaction of these primitives, which we believe is a natural strengthening of circular security. In this sense, it is consistent with Gentry's heuristic step in the FHE bootstrap theorem [Gen09].

We aim to instantiate the underlying primitives randomly (or pseudo-randomly) rather than non-randomly, as non-random instantiations of primitives are insecure, and thus would lead to an insecure split FHE scheme. For randomly instantiated primitives, we can speculate about their security.

Security Proof. In order to prove the security of our scheme, demonstrate the existence of an oracle that interacts securely between the underlying primitives and a randomly instantiated scheme. This oracle is defined as $\mathcal{O}_{(\widehat{pk},\overline{pk},q,\tilde{q})}(x)$: given a string $x \in \{0,1\}^*$ and a ciphertext taken from the ciphertext space of the linear homomorphic scheme,

$$c \leftarrow \mathfrak{C},$$

it then calculates

$$\tilde{c} \leftarrow \mathrm{Eval}(\widehat{pk}, -\lfloor \overline{\mathrm{DEC}}(\cdot, c)/\tilde{q} \rfloor \cdot \tilde{q}, \widehat{c}),$$

and returns $(c, \tilde{c})$. In this paper, we use this oracle for the security proof of the scheme.

Quantum Hash Oracle Model. This quantum hash oracle model is used to prove the security of the scheme and to construct the quantum oracle for the scheme. A detailed introduction to this quantum oracle can be found in [DFMS22] and [Zha19]. Let $D = (D_x)_{x\in\mathcal{X}}$ denote a register, where D_x is a set in the Hilbert space $\mathcal{H}_{D_x} = \mathbb{C}[\{0,1\}^n \cup \{\bot\}]$. The Hilbert space $\mathcal{H}_{D_x}$ can be viewed as a space spanned by a set of orthogonal bases $|y\rangle$, where $y \in \{0,1\}^n \cup \{\bot\}$. Let the unitary transformation U be defined as

$$U|\bot\rangle = |\psi_0\rangle, U|\psi_0\rangle = |\bot\rangle \text{ and } U|\psi_y\rangle = |\psi_y\rangle, \forall y \in \{0,1\}^n \setminus \{0\}^n.$$

Within $|\psi_y\rangle := H|y\rangle$, and H is the Hadamard transform on $\mathbb{C}[\{0,1\}^n] = (\mathbb{C}^2)^{\otimes n}$. Let $|y\rangle = 2^{-n/2}\sum_\eta (-1)^{\eta\cdot y}|\psi_\eta\rangle$, one obtain that

$$U|y\rangle = |y\rangle + 2^{-n/2}(|\bot\rangle - |\psi_0\rangle).$$

When the oracle is queried, the unitary transformation O_{XYZ} will act on the query registers X and Y, as well as the database register D, with its specific expression being

$$O_{XYZ} = \sum_x |x\rangle\langle x| \otimes O^x_{YD_x} \text{ and } O^x_{YD_x} = U_{D_x}\mathrm{CNOT}_{YD_x}U_{D_x}.$$

Where $\mathrm{CNOT}|y\rangle|y_x\rangle = |y\rangle|y \oplus y_x\rangle$, $y, y_x \in \{0,1\}^n$ and $\mathrm{CNOT}|y\rangle|\bot\rangle = |y\rangle|\bot\rangle$. With these tools, present the quantum hash oracle model by Don et al. as follows:

$$\varGamma_R := \max_{x\in\mathcal{X}} |\,\{y \in \{0,1\}^n | \langle x, y\rangle \in \mathbb{R}\}\,|.$$

Furthermore, consider the following projectors:

$$\varPi^x_{D_x} := \sum_{\substack{y\ s.t.\\ \langle x,y\rangle\in\mathbb{R}}} |y\rangle\langle y|_{D_x} \text{ and } \varPi^{\emptyset}_{D_x} := \mathbb{1}_D - \sum_{x\in\mathcal{X}} \varPi^x_{D_x} = \bigotimes_{x\in\mathcal{X}} \bar{\varPi}^x_{D_x}.$$

Where $\bar{\Pi}^x_{D_x} := \mathbb{1}_{D_x} - \Pi^x_{D_x}$. Furthermore, define the measurement $\mathcal{M} = \mathcal{M}^R$, and the following projectors

$$\Sigma^x := \bigotimes_{x'<x} \bar{\Pi}^{x'}_{D_{x'}} \otimes \Pi^x_{D_x} \text{ and } \Sigma^{\emptyset} := \mathbb{1} - \sum_{x'} \Sigma^{x'} = \bigotimes_{x'} \bar{\Pi}^{x'}_{D_{x'}} = \Pi^{\emptyset}.$$

Furthermore, define the pure state measurement unitary transformation $M_{DP} = M^R_{DP} \in L(\mathcal{H}_D \otimes \mathcal{H}_R)$, that is

$$M_{DP} := |\varphi\rangle_D |w\rangle_P \mapsto |\varphi\rangle_D |w + x\rangle_P.$$

1.2 Technical Overview

Next, provide a generalized description of the method for constructing split FHE, and readers can refer to relevant literature for a more detailed description.

Split FHE. In 2019, Brakerski et al. [BDGM19] introduced the concept of a split FHE scheme. Asymptotically, they aimed to design an efficient FHE scheme by eliminating linear noise in previous LWE-based FHE schemes. More specifically, given an FHE ciphertext c and an LWE key $(s_1, \ldots, s_n)$, we can denote the decryption operator as a linear function $\mathcal{L}_c(\cdot)$, that is

$$\mathcal{L}_c(s_1, \ldots, s_n) = \mathrm{ECC}(m) + e.$$

Here, e is a noise term bounded by B, and ECC is the encoding operator for the text. Then, this paper introduces the construction of a linear homomorphic scheme using LWR.DV, and encrypts the key $(s_1, \ldots, s_n)$ with this homomorphic encryption scheme, allowing the compression of FHE ciphertexts through the computation of $\mathcal{L}_c(\cdot)$. The public key of this scheme is $(r \in_R \{0,1\}^n, \odot_q(A, l))$, and it computes the encryption of a message m as

$$c = \lfloor \odot_q(A, lu)(m + k) \rceil_p.$$

Here, $u = H(r)$, where $H : \{0,1\}^n \to \mathbb{Z}_q$ and $k \in_R \{0, \ell+1\}$. Furthermore, this scheme possesses an additional property, which refer to as split decryption. If the decryption algorithm can be divided into a private subroutine and a public subroutine, then the scheme has split decryption:

- The private process takes a ciphertext c and key $(\odot_q(A, lu), T_{sk})$ as input, outputs $\tilde{m} = \mathrm{LWRInvert}(T_{sk}, \odot_q(A, lu), c)$. For each component $\tilde{m}_i$ of $\tilde{m}$,

$$\begin{cases} \tilde{k}_i = 0, & \text{if } \tilde{m}_i \in \{0,1\}, \\ \tilde{k}_i = \tilde{m}_i, & \text{if } \tilde{m}_i \in \{(\ell+1), \ldots, n(\ell+1)\}, \\ \tilde{k}_i = \tilde{m}_i - 1, & \text{if } \tilde{m}_i \notin \{0, 1, (\ell+1), \ldots, n(\ell+1)\}. \end{cases}$$

  It returns the decryption primer $\rho = \left(sk, \tilde{k} = (\tilde{k}_i)_{i \in \{1,\ldots,\ell\}}\right)$.

- The public process takes the ciphertext c and decryption primer ρ as inputs, outputs $\tilde{m} = \mathrm{LWRInvert}(T_{sk}, \odot_q(A, lu), c)$, decrypts $m' = \tilde{m} - \tilde{k}$.

In summary, m can be fully recovered by passing a fixed-size decryption primer, especially independent of the norm of m. As we will discuss later, this property will be the main feature in constructing universal obfuscation.

The Security of Split FHE. We now discuss the security of the split FHE scheme. Our primary concern is ensuring that the decryption primer does not carry any information about the plaintext; otherwise, the simplicity of the split encryption process and straightforward output of keys in every scheme would be moot. We propose a more profound indistinguishability definition, meaning that for all plaintext pairs (m_0, m_1) and any set of circuits $(C_1, \ldots, C_\beta)$, we have $C_i(m_0) = C_i(m_1)$. Even if an adversary knows the decryption primer ρ_i, they cannot distinguish between the encryptions of (m_0, m_1) as (c_0, c_1). The condition $C_i(m_0) = C_i(m_1)$ eliminates some other attacks, where the adversary only needs to check the obfuscator's output. Here, $\beta = \beta(\lambda)$ is a priori bounded polynomial of a security parameter.

Theorem 4 (Informal)**.** *Assuming the sub-exponential hardness of the LWE problem and the LWR problem, there exists a split FHE scheme secure under the $\mathcal{O}$-hybrid security model.*

From Split FHE Scheme to Ideal Obfuscation. Utilize the split FHE scheme presented in this paper to construct ideal obfuscation. Building on the work of Lin et al. [Lin17], we achieve an obfuscated circuit C with input domain $\{0,1\}^\eta$ whose length does not exceed $\mathrm{poly}(\lambda, |C|) \cdot 2\eta \cdot (1 - \varepsilon)$, where $\varepsilon > 0$. This implies that split FHE signifies the existence of an obfuscator with non-trivial efficiency (for circuits with polynomial-size input domains).

2 Preliminary

We define a function negl$(\cdot)$, which is an infinitesimal of any polynomial function poly, and we refer to it as "negligible". Given a set S, $s \in_R S$ means randomly selecting an element s from the set S. When an algorithm can be computed within a polynomial function poly, we say that this algorithm is "computable in polynomial time".

Lemma 1 ([AJLA+12], Smudging)**.** *Let $B_1 = B_1(n)$, $B_2 = B_2(n)$ be positive integers, and $e_1 \in [B_1]$. Let $e_2 \in_R [B_2]$. If $B_1/B_2 = \mathrm{negl}(n)$, then the distribution of e_2 is computationally indistinguishable from the distribution of $e_2 + e_1$.*

2.1 LWR Trapdoor Algorithm

Algorithm 1 LWR Trapdoor Algorithm [AKPW13]

GenTrap($\boldsymbol{n, m, q}$): A method that outputs $A \in \mathbb{Z}_q^{m \times n}$ and a trapdoor T in polynomial time, where the input to this algorithm is an integer n, q, and a sufficiently large integer m. The matrix A is uniformly distributed in $\mathbb{Z}_q^{m \times n}$.

Invert($\boldsymbol{T, A, c}$): A method that outputs $s \in \mathbb{Z}_q^n$ from $c = As + e \in \mathbb{Z}_q^m$ in polynomial time, with $\|e\|_2 \leq \gamma$. The input to this algorithm is the output A and trapdoor T from the **GenTrap($\boldsymbol{n, m, q}$)** algorithm.

LWRInvert($\boldsymbol{T, A, c}$): A method that outputs $s \in \mathbb{Z}_q^n$ from $c = \lfloor As \rceil_p \in \mathbb{Z}_p^m$ in polynomial time. The input to this algorithm is the output A and trapdoor T from the **GenTrap($\boldsymbol{n, m, q}$)** algorithm.

Lemma 2 (Existence Lemma of LWR Trapdoor Algorithm, [AKPW13])**.** *The LWR trapdoor algorithm definitely exists, that is, for integers n, q, sufficiently large integer $m \geq O(n \log q)$, and sufficiently large integer $p \geq O(\sqrt{n \log q})$, there exist algorithms* GenTrap$(\boldsymbol{n, m, q})$ *and* LWRInvert$(\boldsymbol{T, A, c})$ *that output results in polynomial time.*

2.2 Variants of LWR and Their Applications

Lemma 3. *If $a \in_R \mathbb{Z}_q$, then for $r \in_R \mathbb{Z}_p$, $a^r \bmod q \in_R \mathbb{Z}_q$.*

Proof. First, prove that the function $f : a \to a^r \bmod q, a \in_R \mathbb{Z}_q$ is a bijection. If $(a^r - b^r) \bmod q = 0$, it implies that $(a^r - b^r)$ is a multiple of q, which means $a^r = b^r (\text{mod} q)$. According to Fermat's Little Theorem, if p is a prime and a is a multiple of p, then $a^p = a(\bmod p)$. Therefore, $a^q - 1 = 1(\text{mod} q)$. If $a \neq b$, represent a and b as powers of some primitive root g modulo q, i.e., $a = g^x(\text{mod} q)$, $b = g^y(\text{mod} q)$, then $a^r - b^r = g^{xr} - g^{yr}(\text{mod} q)$. Since g is a primitive root modulo q, the order of g is $q - 1$. According to Euler's theorem, if a and n are coprime, then $a^{\varphi(n)} = 1(\text{mod} n)$. Therefore,

$$g^{xr} - g^{yr} = g^{xr \bmod (q-1)} - g^{yr \bmod (q-1)} (\text{mod} q).$$

Since $a \neq b$, we have $x \neq y$. Thus, $xr \bmod (q-1) \neq yr \bmod (q-1)$, which means $g^{xr \bmod (q-1)} - g^{yr \bmod (q-1)}$ is not equal to 0. Therefore, $(a^r - b^r) \bmod q = 0$ only when $a = b$. Hence, $f : a \to a^r \bmod q, a \in_R \mathbb{Z}_q$ is injective. Moreover, since $\mathbb{Z}_q$ is a finite set, f is surjective, thus f is a bijection. Therefore, if $a \in_R \mathbb{Z}_q$, then for $r \in_R \mathbb{Z}_p$, $a^r \bmod q \in_R \mathbb{Z}_q$. □

Theorem 5. *If $A \in_R \mathbb{Z}_q^{m\times n}$, then $\odot_q(A,r) \in_R \mathbb{Z}_q^{m\times n}$. Where the operation $\odot_q(A,r)$ is defined as follows:*

$$\odot_q(A,r) = \begin{cases} \tilde{A}\left(a_{ij} \in A, a_{ij}^{r} \in \tilde{A}, i \in \{1,\dots,m\}, j \in \{1,\dots,n\}\right) \bmod q, & forr \in \mathbb{Z}_q, \\ \tilde{A}\left(a_{ij} \in A, a_{ij}^{r_{ij}} \in \tilde{A}, i \in \{1,\dots,m\}, j \in \{1,\dots,n\}\right) \bmod q, & forr \in \mathbb{Z}_q^{m\times n}. \end{cases}$$

Proof. From Lemma 3, if $a_{ij} \in_R \mathbb{Z}_q$, then $a_{ij}^{r} \bmod q \in_R \mathbb{Z}_q$ (or $a_{ij}^{r_{ij}} \bmod q \in_R \mathbb{Z}_q$). Therefore, $\odot_q(A,r) \in_R \mathbb{Z}_q^{m\times n}$. □

Theorem 6. *If $(A, \lfloor As \rfloor_p)$ and $(A, \lfloor u \rfloor_p)$ are indistinguishable, then for $r \in_R \mathbb{Z}_p$ (or $r \in \mathbb{Z}_q^{m\times n}$), $(\odot_q(A,r), \lfloor \odot_q(A,r)\cdot s \rfloor_p)$ and $(\odot_q(A,r), \lfloor u \rfloor_p)$ are also indistinguishable.*

Proof. According to the form of LWR, when $A \in \mathbb{Z}_q^{m\times n}$, we have $\tilde{A} = \odot_q(A,r) \in \mathbb{Z}_q^{m\times n}$. Therefore, $(\tilde{A}, \lfloor \tilde{A}s \rfloor_p)$ and $(\tilde{A}, \lfloor u \rfloor_p)$ still maintain indistinguishability. □

Definition 1 (Variant LWR Problem). *Let $A \in \mathbb{Z}_q^{m\times n}$, $s \in \mathbb{Z}_q^{n}$, $u \in \mathbb{Z}_q^{m}$, and for $r \in_R \mathbb{Z}_p$ (or $r \in \mathbb{Z}_q^{m\times n}$), the Variant LWR Problem is to distinguish whether $(\odot_q(A,r), b)$ comes from $(\odot_q(A,r), \lfloor u \rfloor_p)$ or from $(\odot_q(A,r), \lfloor \odot_q(A,r)\cdot s \rfloor_p)$, where $b \in \mathbb{Z}_p^{m}$.*

Corollary 1. *If there exists an algorithm $\mathcal{O}$ to solve the LWR problem, then there also exists an algorithm $\mathcal{O}'$ to solve the Variant LWR problem, and vice versa.*

Proof. According to Theorem 5, the sufficiency of the proposition is established. Now, to prove the necessity, since f is a bijection, there exists f^{-1} such that $f^{-1}\cdot f = f\cdot f^{-1} = Id$. It can be easily shown that f^{-1} is also a bijection. Hence, when $\odot_q(A,r) \in_R \mathbb{Z}_q^{m\times n}$, it implies $A \in_R \mathbb{Z}_q^{m\times n}$, thus the necessity is proved. □

Definition 2 (IND-CPA). *The in distinguishability under chosen-plaintext attack (IND-CPA) game for public key encryption scheme is as follows:*

(1) **Setup**. *The simulator $\mathcal{B}$ generates the system Π, and the adversary $\mathcal{A}$ receives the public key of the system.*

(2) **Training**. *The adversary $\mathcal{A}$ generates plaintext messages and obtains the ciphertext after encryption by the system.*

(3) **Challenge**. *The adversary $\mathcal{A}$ outputs two plaintext messages M_0 and M_1 of the same length. The simulator $\mathcal{B}$ randomly chooses $\beta \leftarrow_R \{0,1\}$, encrypts M_β, and sends the resulting ciphertext C^* to the adversary.*

(4) **Guess**. *The adversary $\mathcal{A}$ outputs β'. If $\beta' = \beta$, then the adversary $\mathcal{A}$ succeeds in the attack.*

The advantage of adversary $\mathcal{A}$ can be defined as a function of the parameter n:

$$Adv_{\Pi,\mathcal{A}}^{\mathrm{IND-CPA}}(n) = \left|\Pr[\beta' = \beta] - \frac{1}{2}\right|.$$

Definition 3 (IND-ID-CPA)**.** *The indistinguishability under identity-based chosen-plaintext attack (IND-ID-CPA) game for Identity-Based Encryption (IBE) scheme is as follows:*

(1) **Setup**. *The simulator $\mathcal{B}$ generates the system Π, creates public parameters and a master secret key msk.*

(2) **Training 1**. *The adversary $\mathcal{A}$ issues queries for private keys corresponding to id. The simulator $\mathcal{B}$ uses the key generation algorithm to generate the private key sk corresponding to id and sends it to the adversary $\mathcal{A}$. This process can be repeated a polynomial number of times.*

(3) **Challenge**. *The adversary $\mathcal{A}$ outputs two plaintext messages M_0 and M_1 of the same length, and a public identity id^* that has not been queried in* **Training 1** *stage. The simulator $\mathcal{B}$ randomly chooses $\beta \leftarrow_R \{0,1\}$, encrypts M_β, and sends the ciphertext $C^* = \varepsilon_{id^*}(M_\beta)$ to the adversary $\mathcal{A}$.*

(4) **Training 2**. *The adversary $\mathcal{A}$ issues queries for private keys corresponding to another id, $id \neq id^*$. The simulator $\mathcal{B}$ uses the key generation algorithm to generate the private key sk corresponding to id and sends it to the adversary $\mathcal{A}$. This process can be repeated a polynomial number of times.*

(5) **Guess**. *The adversary $\mathcal{A}$ outputs β'. If $\beta' = \beta$, then the adversary $\mathcal{A}$ succeeds in the attack.*

The advantage of adversary $\mathcal{A}$ can be defined as a function of the parameter n:

$$Adv_{\Pi,\mathcal{A}}^{\mathrm{IND-ID-CPA}}(n) = \left| \Pr[\beta' = \beta] - \frac{1}{2} \right|.$$

Scheme 2

Setup. Choose a random matrix $l \in_R \mathbb{Z}_q^{m\times n}$ and $A \in_R \mathbb{Z}_q^{m\times n}$. Generate sample $(\tilde{A} = \odot_q(A,l), T) \leftarrow \mathrm{GenTrap}(n,m,q)$. Set the public key $pk = \tilde{A}$ and the trapdoor secret key $sk = (\tilde{A}, T)$.

Enc$_{\boldsymbol{pk}}$($\boldsymbol{s}$). For a plaintext $s \in \{0,1\}^n$, choose a random vector $k \in \{0,2\}^n$ and output $c = \lfloor \tilde{A}(s+k) \rfloor_p$.

Dec$_{\boldsymbol{sk}}$($\boldsymbol{c}$). For a ciphertext $c \in \mathbb{Z}_p^n$, output $\overline{s} = \mathrm{LWRInvert}(T, \tilde{A}, c)$. For each component $\overline{s}_i$ of $\overline{s}$, if $\overline{s}_i \geq 2$, set $s'_i = \overline{s}_i - 2$, otherwise set $s'_i = \overline{s}_i$, obtaining the plaintext $s' = (s'_i), i \in \{1, \dots, n\}$.

Theorem 7. *Assume that Variant LWR problem is hard, then Scheme 2 is IND-CPA secure.*

Proof. The following is the simulation process of the IND game.

(1) **Initialization**. The simulator $\mathcal{B}$ selects random matrices $r, A \in_R \mathbb{Z}_q^{m\times n}$ and outputs the sample $(\odot_q(A,r),T) \leftarrow GenTrap(n,m,q)$. Set the public key $pk = \odot_q(A,r)$ to the adversary $\mathcal{A}$ and the trapdoor secret key $sk = (\odot_q(A,r),T)$.

Enc$_{\boldsymbol{pk}}$($\boldsymbol{s}$). For a plaintext $s \in \{0,1\}^n$, choose a random vector $k \in \{0,2\}^n$ and output $c = \lfloor \odot_q(A,r) \cdot (s+k) \rfloor_p$.

Dec$_{\boldsymbol{sk}}$($\boldsymbol{c}$). For a ciphertext $c \in \mathbb{Z}_p^n$, output $\overline{s} = \text{LWRInvert}(T, \odot_q(A,r), c)$. For each component $\overline{s}_i$ of $\overline{s}$, if $\overline{s}_i \geq 2$, set $s'_i = \overline{s}_i - 2$, otherwise set $s'_i = \overline{s}_i$, obtaining the plaintext $s' = (s'_i), i \in \{1,\ldots,n\}$.

(2) **Training**. The adversary $\mathcal{A}$ generates plaintext messages $s_1, s_2, \ldots, s_t$ and obtains the corresponding ciphertexts $c_1 = \lfloor \odot_q(A,r)(s_1+k_1)\rfloor_p$, $c_2 = \lfloor \odot_q(A,r)(s_2+k_2)\rfloor_p$, …, $c_t = \lfloor \odot_q(A,r)(s_t+k_t)\rfloor_p$.

(3) **Challenge**. The adversary outputs two messages M_0 and M_1. The simulator $\mathcal{B}$ randomly chooses $\beta \leftarrow_R \{0,1\}$, encrypts M_β, and sends the resulting ciphertext $C^* = \lfloor \odot_q(A,r)(M_\beta + k_\beta)\rfloor_p$ to the adversary.

(4) **Guess**. The adversary $\mathcal{A}$ outputs β'. If $\beta' = \beta$, the adversary succeeds in the attack.

Assuming that the adversary $\mathcal{A}$ has a non-negligible advantage in outputting $\beta' = \beta$, then the simulator $\mathcal{B}$ would also have the means to know that $(\odot_q(A,r), \lfloor \odot_q(A,r)\cdot s\rfloor_p)$ is not random, and hence have a non-negligible advantage in distinguishing between $(\odot_q(A,r), \lfloor u \rfloor_p)$ and $(\odot_q(A,r), \lfloor \odot_q(A,r)\cdot s\rfloor_p)$, which contradicts "Variant LWR problem is hard". Therefore, the adversary $\mathcal{A}$ also has a negligible advantage in outputting $\beta' = \beta$. Thus, Scheme 2 is IND-CPA secure. □

According to the idea of Scheme 2, we can construct an IBE scheme as follows:

Scheme 3

IBESetup(($\boldsymbol{n,m,q}$) $\rightarrow$ ($\boldsymbol{pk, msk}$)): Choose random matrices $l, A \in_R \mathbb{Z}_q^{m\times n}$, output samples $(\odot_q(A,l),T) \leftarrow \text{GenTrap}(n,m,q)$. Let $pk = \odot_q(A,l)$ and trapdoor key $msk = (l, A, T_{msk})$.

IBEExtract(($\boldsymbol{id, pk, msk}$) $\rightarrow$ $\boldsymbol{sk}$): Given identity $id \in \{0,1\}^n$, let $u = H(id) \in \mathbb{Z}_q$, output sample $(\odot_q(A,lu),T_{sk}) \leftarrow \text{GenTrap}(n,m,q)$, let $sk = (\odot_q(A,lu),T_{sk})$.

IBEEnc$_{\boldsymbol{pk}}$(($\boldsymbol{id, pk, s}$) $\rightarrow$ $\boldsymbol{c}$): Given identity $id \in \{0,1\}^n$, let $u = H(id) \in \mathbb{Z}_q$. For each element a_{ij}^l of $\odot_q(A,l)$, calculate $(a_{ij}^l)^u \bmod q = a_{ij}^{lu}$, thus obtaining $\odot_q(A,lu)$. For plaintext $s \in \{0,1\}^n$, choose a random vector $k \in \{0,2\}^n$, output $c = \lfloor \odot_q(A,lu)(s+k)\rfloor_p$.

IBEDec$_{\boldsymbol{sk}}$(($\boldsymbol{c, sk}$) $\rightarrow$ $\boldsymbol{s}$): For ciphertext $c \in \mathbb{Z}_p^n$, output $\overline{s} = \text{LWRInvert}(T_{sk}, \odot_q(A,lu), c)$, for each component $\overline{s}_i$ of $\overline{s}$, if $\overline{s}_i \geq 2$ then let $s'_i = \overline{s}_i - 2$, otherwise let $s'_i = \overline{s}_i$, obtaining plaintext $s' = (s'_i), i \in \{1,\ldots,n\}$.

Theorem 8. *If $(A, \lfloor As \rfloor_p)$ is indistinguishable from $(A, \lfloor u \rfloor_p)$, then for randomly chosen $b \in_R \mathbb{Z}_p$ and $a \in_R \mathbb{Z}_p^{m\times n}$, we have*

$$(\odot_q(A,a), \lfloor u \rfloor_p) \approx_c \left(\odot_q(A,a), \lfloor \odot_q(A,ab) \cdot s \rfloor_p\right).$$

Proof.

$$\begin{aligned}(\odot_q(A,a), \lfloor u \rfloor_p) &\approx_c \left(\odot_q(A,ab), \lfloor u \rfloor_p\right)\\ &\approx_c \left(\odot_q(A,ab), \lfloor \odot_q(A,ab) \cdot s \rfloor_p\right)\\ &\approx_c \left(\odot_q(A,a), \lfloor \odot_q(A,ab) \cdot s \rfloor_p\right).\end{aligned}$$

□

Remark 1. *This implies that Theorem 8 is also a variant of LWR. From the proof of Theorem 8, we can see that the underlying algorithms of Scheme 3 and Scheme 2 are indistinguishable, therefore the underlying algorithm of Scheme 3 is IND-CPA secure.*

Theorem 9. *Scheme 3 is secure under the IND-ID-CPA security definition.*

Proof. The following is the complete process of simulating the IND game.

(1) **Initialization**. The simulator $\mathcal{B}$ selects random matrices $a, A \in_R \mathbb{Z}_q^{m\times n}$ and outputs samples $(\odot_q(A,a), T) \leftarrow \text{GenTrap}(n,m,q)$. Let the public key be $pk = \odot_q(A,a)$ and the trapdoor key be $msk = (a, A, T_{msk})$.

Extract. Given an identity id, let

$$b = \max_{x \in \mathbb{Z}_q^n} |\{id \in \{0,1\}^n | \langle x, id\rangle \in \mathbb{Z}\}| \bmod q.$$

Output sample $(\odot_q(A,ab), T_{sk}) \leftarrow \text{GenTrap}(n,m,q)$, and let $sk = (\odot_q(A,ab), T_{sk})$.

Enc$_{pk}(s)$. Given an identity $id \in \{0,1\}^n$, let

$$b = \max_{x \in \mathbb{Z}_q^n} |\{id \in \{0,1\}^n | \langle x, id\rangle \in \mathbb{Z}\}| \bmod q.$$

For each element d_{ij}^a of $\odot_q(A,a)$, compute $(d_{ij}^a)^b = d_{ij}^{ab} \bmod q$ to obtain $\odot_q(A,ab)$. For a plaintext $s \in \{0,1\}^n$, choose a random vector $k \in \{0,2\}^n$, and output $c = \lfloor \odot_q(A,ab)(s+k) \rfloor_p$.

Dec$_{sk}(c)$. Given a ciphertext $c \in \mathbb{Z}_p^n$, output $\overline{s} = \text{LWRInvert}(T_{sk}, \odot_q(A,ab), c)$, for each component $\overline{s}_i$ of $\overline{s}$, if $\overline{s}_i \geq 2$ then let $s'_i = \overline{s}_i - 2$, otherwise let $s'_i = \overline{s}_i$, obtaining plaintext $s' = (s'_i), i \in \{1, \ldots, n\}$.

(2) **Training Phase 1**. Let t be a polynomially bounded number of queries.

- **Hash Query.** Build a list of hash query values for each of the adversary's queries $id_i, i \in \{1, \ldots, t\}$. Check if id_i has been queried before. If not, compute

$$b_i = \max_{x \in \mathbb{Z}_q^n} |\{id_i \in \{0,1\}^n | \langle x, id_i \rangle \in \mathbb{Z}\}| \bmod q,$$

and store (id_i, b_i, i) in the list; if already queried, look up and return the result to the adversary $\mathcal{A}$.

- **Private Key Generation Query.** For each of the adversary's queries $id_i, i \in \{1, \ldots, t\}$ for the key sk_i, use Extract to generate $sk_i = (\odot_q(A, a \cdot b_i), T_{sk_i})$ and provide it to the adversary $\mathcal{A}$.

(3) **Challenge Phase**. The adversary outputs two messages M_0 and M_1 of the same length and a public id^* that has not been queried in the hash or private key generation queries. Calculate

$$b^* = \max_{x \in \mathbb{Z}_q^n} |\{id^* \in \{0,1\}^n | \langle x, id^* \rangle \in \mathbb{Z}\}| \bmod q.$$

The simulator $\mathcal{B}$ randomly chooses $\beta \leftarrow_R \{0,1\}$, encrypts M_β, and gives the ciphertext $C^* = \lfloor \odot_q(A, a \cdot b^*)(M_\beta + k_\beta) \rfloor_p$ to the adversary $\mathcal{A}$.

(4) **Training Phase 2**. Similar to **Training Phase 1**, the adversary $\mathcal{A}$ will make hash queries and private key generation queries for another set of identities, where $id \neq id^*$.

(5) **Guess Phase**. The adversary $\mathcal{A}$ outputs β', and if $\beta' = \beta$, the adversary succeeds in the attack.

Assuming that the adversary $\mathcal{A}$ has a non-negligible advantage in outputting $\beta' = \beta$, then the simulator $\mathcal{B}$ would also be able to determine that $(\odot_q(A, a), \lfloor \odot_q(A, a \cdot b^*) \cdot s \rfloor_p)$ is not random, thus having a non-negligible advantage in distinguishing $(\odot_q(A, a), \lfloor u \rfloor_p)$ and $(\odot_q(A, a), \lfloor \odot_q(A, a \cdot b^*) \cdot s \rfloor_p)$, which contradicts Theorem 8. Therefore, the adversary $\mathcal{A}$ also has a negligible advantage in outputting $\beta' = \beta$. Thus, Scheme 3 is secure under the IND-ID-CPA definition. □

2.3 Homomorphic Encryption and Ideal Obfuscation

Homomorphic encryption is defined as follows:

Definition 4. *A homomorphic encryption scheme consists of the following components:*

- **KeyGen(n):** *Given a security parameter n, the key generation part returns a key pair (sk, pk).*
- **Enc(pk, m):** *Given the public key pk and the plaintext message m, the encryption part returns the encrypted ciphertext c.*
- **Eval(pk, C, $(c_1, \ldots, c_\ell)$):** *Given the public key pk, a circuit C of depth ℓ, and a vector of ciphertexts $(c_1, \ldots, c_\ell)$, the homomorphic operation part returns the ciphertext after homomorphic computation.*

- **Dec(sk, c):** *Given the private key sk and the ciphertext c, the decryption part returns the decrypted plaintext message m.*

Definition 5 (Correctness)**.** *Let $n \in \mathbb{N}$, and C be a circuit of depth ℓ. For an encryption scheme* $(\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Eval}, \mathrm{Dec})$ *with inputs $(m_1, \ldots, m_\ell)$, key pair* $(\mathrm{pk}, \mathrm{sk})$ *generated by* $\mathrm{KeyGen}(n)$, *and ciphertexts c_i generated by* $\mathrm{Enc}(\mathrm{pk}, m_i)$ *according to the scheme, we have*

$$\Pr[\mathrm{Dec}(\mathrm{sk}, \mathrm{Eval}(\mathrm{pk}, C, (c_1, \ldots, c_\ell))) = C(c_1, \ldots, c_\ell)] = 1.$$

Refer to such an encryption scheme as a homomorphic encryption scheme. We desire that the length of ciphertexts in the scheme does not increase due to the depth ℓ of circuit C, a property referred to as "compactness" (distinct from the concept of "compactness" in functional analysis).

Definition 6 (Compactness)**.** *Let $n \in \mathbb{N}$, C be a circuit of depth ℓ, and* $\mathrm{poly}(\cdot)$ *be a polynomial function. For a homomorphic encryption scheme* $(\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Eval}, \mathrm{Dec})$ *with inputs $(m_1, \ldots, m_\ell)$, key pair (pk, sk) generated by* $\mathrm{KeyGen}(n)$, *and ciphertexts c_i generated by* $\mathrm{Enc}(pk, m_i)$, *if*

$$|\mathrm{Eval}(\mathrm{pk}, C, (c_1, \ldots, c_\ell))| = \mathrm{poly}(n) \cdot |C(m_1, \ldots, m_\ell)|,$$

then one called the homomorphic encryption scheme compact. Define a weak security notion (implied by standard semantic security [38]) for convenience.

Definition 7 (Semantic Security)**.** *Let $n \in \mathbb{N}$, C be a circuit of depth ℓ, and* $\mathrm{negl}(\cdot)$ *be a negligible function. For a homomorphic encryption scheme* $(\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Eval}, \mathrm{Dec})$ *with inputs (m_0, m_1), key pair (pk, sk) generated by* $\mathrm{KeyGen}(n)$, *ciphertexts c_i generated by* $\mathrm{Enc}(pk, m_i)$, *and all polynomial-time distinguishers $\mathcal{D}$, if*

$$|\Pr[1 = \mathcal{D}(pk, \mathrm{Enc}(pk, m_0))] - \Pr[1 = \mathcal{D}(pk, \mathrm{Enc}(pk, m_1))]| = \mathrm{negl}(n),$$

then one called the homomorphic encryption scheme semantically secure. Here, the key pair (pk, sk) is generated by $\mathrm{KeyGen}(n)$ *of the scheme.*

Definition 8 (ϵ-Indistinguishability)**.** *Consider two distributions $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$, and $\epsilon : \mathbb{N} \to [0, 1]$. If for every sufficiently large $\lambda \in \mathbb{N}$, it holds that*

$$\left| \Pr_{x \leftarrow \mathcal{X}_\lambda}[\mathcal{A}(1^\lambda, x) = 1] - \Pr_{y \leftarrow \mathcal{Y}_\lambda}[\mathcal{A}(1^\lambda, y) = 1] \right| \leq \epsilon(\lambda),$$

one said that the two distributions $\mathcal{X}$ and $\mathcal{Y}$ are indistinguishable. Here, $\mathcal{A}$ is a probabilistic polynomial-time adversary. Specifically, when $\epsilon(\lambda) = \mathrm{negl}(\lambda)$, *one called $\mathcal{X}$ and $\mathcal{Y}$ indistinguishable with respect to ϵ; when $\epsilon(\lambda) = 2^{-\lambda^c}$, one called $\mathcal{X}$ and $\mathcal{Y}$ sub-exponentially indistinguishable.*

Definition 9 (Circuit Obfuscation)**.** *A circuit obfuscation scheme under the ideal model with an oracle $\mathcal{O}$ is said to be efficient* $\mathrm{Obf}^{\mathcal{O}}(\lambda, C)$ *if, for a given input circuit C, it outputs an obfuscated circuit $\widehat{C}^\bullet$. The scheme is required to be correct, meaning that for all $\lambda \in \mathbb{N}$, where the circuit $C : \{0,1\}^D \to \{0,1\}^*$ and input $x \in \{0,1\}^D$, the following relation holds:*

$$\Pr[\widehat{C}^\bullet \leftarrow \mathrm{Obf}^{\mathcal{O}}(\lambda, C) : \widehat{C}^{\mathcal{O}} = C(x)] = 1.$$

Definition 10 (Ideal Obfuscation)**.** *A circuit obfuscation scheme* $\mathrm{Obf}^{\mathcal{O}}(\lambda, C)$ *is said to be ideal if there exists an efficient simulator* $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$ *such that for all adversaries* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, the adversary's advantage is negligible, i.e.,*

$$\Pr\left[\begin{array}{c} C \leftarrow \mathcal{A}_1^{\mathcal{O}}(\lambda) \\ \widehat{C}^{\bullet} \leftarrow \mathrm{Obf}^{\mathcal{O}}(\lambda, C) \end{array} : \mathcal{A}_2^{\mathcal{O}}(\widehat{C}^{\bullet}) = 1\right] - \Pr\left[\begin{array}{c} C \leftarrow \mathcal{A}_1^{\mathcal{S}_1}(\lambda) \\ \widetilde{C}^{\bullet} \leftarrow \mathcal{S}_2^{C}(\lambda, D, S) \end{array} : \mathcal{A}_2^{\mathcal{S}_3^{C}}(\widetilde{C}^{\bullet}) = 1\right].$$

Here, $D = |x|$ *is the length of the input circuit* C*, and* $S = |C|$ *is the size of the circuit* C*.*

3 Linear Homomorphic Encryption Scheme based on LWR Variant Problems

Scheme 4 LHE Scheme based on LWR Problem

LWR.DV.KeyGen$(\boldsymbol{n, m, q})$. Choose a random vector $r \in \{0,1\}^n$ and matrices $A, l \in_R \mathbb{Z}_q^{m\times n}$. Let $u = H(r) \in \mathbb{Z}_q$, output sample $(\odot_q(A, lu), T_{sk}) \leftarrow \mathrm{GenTrap}(n, m, q)$, let $pk = (r, \odot_q(A, l))$ and $sk = (\odot_q(A, lu), T_{sk})$.

LWR.DV.Enc$(\boldsymbol{pk, s, q, p})$. Let $u = H(r) \in \mathbb{Z}_q$. For each element a_{ij}^{l} of $\odot_q(A, l)$, compute $(a_{ij}^{l})^u \bmod q = a_{ij}^{lu}$, thus obtaining $\odot_q(A, lu)$. For plaintext $s \in \{0,1\}^n$, choose a random vector $k \in \{0, \ell+1\}^n$, output $c = \lfloor \odot_q(A, lu)(s + k)\rceil_p$.

LWR.DV.Eval$(\boldsymbol{pk, q, p, f, (c_1, \ldots, c_\ell)})$. Input ciphertext vector $(c_1, \ldots, c_\ell)$ and linear function $g = (\alpha_1, \ldots, \alpha_\ell) \in \{0,1\}^\ell$, compute

$$c = \sum_{i=1}^{\ell} \alpha_i c_i \bmod q.$$

LWR.DV.PDec$(\boldsymbol{sk, c})$. For ciphertext $c \in \mathbb{Z}_p^n$, output $\tilde{s} = \mathrm{LWRInvert}(T_{sk}, \odot_q(A, lu), c)$, for each component $\tilde{s}_i$ of $\tilde{s}$,

$$\begin{cases} \tilde{k}_i = 0, & \text{when } \tilde{s}_i \in \{0,1\}, \\ \tilde{k}_i = \tilde{s}_i, & \text{when } \tilde{s}_i \in \{(\ell+1), \ldots, n(\ell+1)\}, \\ \tilde{k}_i = \tilde{s}_i - 1, & \text{when } \tilde{s}_i \notin \{0, 1, (\ell+1), \ldots, n(\ell+1)\}. \end{cases}$$

Return $\rho = \left(sk, \tilde{k} = (\tilde{k}_i)_{i\in\{1,\ldots,\ell\}}\right)$.

LWR.DV.Rec$(\boldsymbol{\rho, c})$. For ciphertext $c \in \mathbb{Z}_p^n$, output $\tilde{s} = \mathrm{LWRInvert}(T_{sk}, \odot_q(A, lu), c)$, decrypt $s' = \tilde{s} - \tilde{k}$.

Simulatable Decryption Hint. For given ciphertext c and plaintext message $\tilde{s}$ (where c and $\tilde{s}$ are unrelated), choose $\tilde{k} \in_R \{0, \ell+1, \ldots, n(\ell+1)\}^n$, $\tilde{u} \in_R \mathbb{Z}_q$. Let $\tilde{sk} \leftarrow \mathrm{GenTrap}(n, m, q)$, compute simulated ciphertext $\tilde{c}$ and

$$\tilde{c}_i = \left|\left(c - \lfloor \odot_q(A, l\tilde{u})(\tilde{s} + \tilde{k})\rceil_p\right)_i\right|, i \in \{1, \ldots, n\}.$$

Then output $\tilde{\rho} = (\tilde{sk}, \tilde{k})$.

4 Splitting Fully Homomorphic Encryption Scheme

Next, we will introduce an instantiation of FHE with split decryption. First propose a scheme based on standard assumptions, which assumes the existence of a structured version of a random oracle, and then present a trusted candidate scheme for this oracle.

4.1 Defining a Special Oracle for Constructing Splitting Fully Homomorphic Schemes

Before presenting the split fully homomorphic scheme, define a special oracle. The parameters of this oracle are $(\widehat{pk}, \overline{pk}, q, \tilde{q})$, where the input is a string $x \in \{0,1\}^*$, and it uniformly outputs encrypted values for LHE and FHE. The oracle is deterministic and accessible to all parties, so when given the same input x, the oracle always outputs the same pair of ciphertexts. The formal definition of this oracle is as follows.

Definition 11 ([BDGM20])**.** $\mathcal{O}_{(\widehat{pk},\overline{pk},q,\tilde{q})}$*: Given input string* $x \in \{0,1\}^*$*, outputs two ciphertexts that are uniformly distributed:*

$$\overline{\mathrm{Enc}}(\overline{pk}, s) \text{ and } \widehat{\mathrm{Enc}}(\widehat{pk}, -\lfloor s/\tilde{q} \rfloor \cdot \tilde{q})$$

where $s \leftarrow \mathbb{Z}_q$*.*

The oracle $\mathcal{O}_{(\widehat{pk},\overline{pk},q,\tilde{q})}$ can encrypt the private key of FHE using LHE scheme, and the resulting ciphertexts follow a uniform distribution. This is because we use the decryption and multiplication algorithms DEC&Mult in the FHE scheme to compute $\overline{\mathrm{Enc}}(\overline{pk}, s - \lfloor s/\tilde{q} \rfloor \cdot \tilde{q} + \text{noise})$, where the noise is the decryption noise of the FHE scheme. By choosing appropriate parameters $\tilde{q}$, we can achieve

$$\begin{aligned} \overline{\mathrm{Enc}}(\overline{pk}, s - \lfloor s/\tilde{q} \rfloor \cdot \tilde{q} + \text{noise}) &= \overline{\mathrm{Enc}}(\overline{pk}, (s \bmod \tilde{q}) + \text{ noise}) \\ &\approx_s \overline{\mathrm{Enc}}(\overline{pk}, (s \bmod \tilde{q})). \end{aligned}$$

Thus, one obtained ciphertexts that are statistically indistinguishable through the two encryption systems.

Description. Now, provide a formal description of our scheme. We assume the existence of the following primitives:

- **FHE** = $(\widehat{\mathbf{KeyGen}}, \widehat{\mathbf{Enc}}, \widehat{\mathbf{Eval}}, \widehat{\mathbf{Dec}})$ with linear decryption-multiplication and noise constraint B, then we refer to **FHE** as fully homomorphic encryption;
- **LHE** = $(\overline{\mathbf{KeyGen}}, \overline{\mathbf{Enc}}, \overline{\mathbf{Eval}}, \overline{\mathbf{PDec}}, \overline{\mathbf{Rec}})$ with small decryption hints and simulatable decryption hints, then we refer to **LHE** as linear homomorphic encryption.

If the underlying FHE scheme is leveled out, then it will result in split FHE. Conversely, if the FHE scheme supports evaluation of unbounded circuits, then the resultant split FHE construction will also do so. The formal description of this scheme is as follows.

Scheme 5 Split Homomorphic Encryption Scheme

KeyGen(n, m, q). Given security parameter n, output sample $(\overline{sk}, \overline{pk}) \leftarrow \overline{\text{KeyGen}}(n)$. Let $\mathbb{Z}_q$ be the plaintext space under LHE definition, output sample $(\widehat{sk}, \widehat{pk}) \leftarrow \widehat{\text{KeyGen}}(n, m, q)$. Let $\widehat{sk} = (T_1, \dots, T_n) \in \{0,1\}^{n \times n}$, then return

$$sk = \overline{sk} \text{ and } pk = (\widehat{pk}, \overline{pk}, \bar{c}_1, \dots, \bar{c}_n).$$

where, for any $i \in [n]$, define $\bar{c}_i \leftarrow \overline{\text{Enc}}(\overline{pk}, T_i)$.

Enc$(\boldsymbol{pk}, \boldsymbol{s})$. Return the ciphertext

$$c \leftarrow \widehat{\text{Enc}}(\widehat{pk}, s).$$

Eval$(\boldsymbol{pk}, \boldsymbol{f}, (\boldsymbol{c_1}, \dots, \boldsymbol{c_\ell})$. Given a circuit $\mathcal{C}$ of ℓ bits and ciphertexts of length k bits $(c_1, \dots, c_\ell))$. For any $j \in [k]$, $\mathcal{C}_j$ is the j-th component of circuit $\mathcal{C}$, calculate

$$d_j \leftarrow \widehat{\text{Eval}}(\widehat{pk}, C_j, (c_1, \dots, c_\ell)).$$

Define the linear function over $\mathbb{Z}_q$ as

$$g(x_1, \dots, x_n) = \sum_{j=1}^{k} \text{DEC\&Mult}\left((x_1, \dots, x_n), d_j, 2^{\lceil \log(\tilde{q}+(k+1)B) \rceil + j}\right).$$

Compute $d \leftarrow \overline{\text{Eval}}(\overline{pk}, g, (\bar{c}_1, \dots, \bar{c}_n))$, then query $(a, \tilde{a}) \leftarrow \mathcal{O}_{(\widehat{pk}, \overline{pk}, q, \tilde{q})}(d)$ and define the following linear function

$$\tilde{g}(x_1, \dots, x_n, x_{n+1}, x_{n+2}) = \text{DEC\&Mult}((x_1, \dots, x_n), \tilde{a}, 1) + x_{n+1} + x_{n+2}.$$

Return

$$c \leftarrow \overline{\text{Eval}}(\overline{pk}, \tilde{g}, (\bar{c}_1, \dots, \bar{c}_n), d, a).$$

PDec$(\boldsymbol{sk}, \boldsymbol{c})$. Given an evaluable ciphertext c, return

$$\rho \leftarrow \overline{\text{PDec}}(\overline{sk}, c).$$

Rec$(\boldsymbol{\rho}, \boldsymbol{c})$. Given an evaluable ciphertext c, return

$$\tilde{s} \leftarrow \overline{\text{Rec}}(\rho, c),$$

and return the binary representation of $\tilde{s}$ without the $\lceil \log(\tilde{q} + (k+1)B) \rceil$ least significant bits.

Analysis: During the analysis, set parameters as needed to ensure the scheme can decrypt correctly. Subsequently, demonstrate that our choices lead to a set of satisfiable constraints. These constraints satisfy the conditions of the underlying hard problems, thus the hardness problem assumptions still hold. The following theorem establishes correctness.

Theorem 10 (Correctness of Split Homomorphic Encryption Scheme)**.** *Let* $q \geq 2^k + 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil}$. *Assuming that* FHE *and* LHE *are correct, then* **Scheme 5** *satisfies the correctness of split homomorphism.*

Proof. We rewrite

$$\tilde{s} = \overline{\text{Rec}}(\rho, c) = \overline{\text{Rec}}(\overline{\text{PDec}}(\overline{sk}, c), c),$$

where $c = \overline{\text{Eval}}(\overline{pk}, \tilde{g}, (\bar{c}_1, \ldots, \bar{c}_n), d, a))$. By the correctness of the LHE scheme, we can rewrite d as

$$\begin{aligned} d &= \overline{\text{Eval}}(\overline{pk}, g, (\bar{c}_1, \ldots, \bar{c}_n)) \\ &= \overline{\text{Eval}}(\overline{pk}, g, (\overline{\text{Enc}}(\overline{pk}, T_1), \ldots, \overline{\text{Enc}}(\overline{pk}, T_n))) \\ &= \overline{\text{Enc}}\left(\overline{pk}, \sum_{j=1}^{k} \text{DEC\&Mult}\left((T_1, \ldots, T_n), d_j, 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j}\right)\right). \end{aligned}$$

Where

$$d_j = \widehat{\text{Eval}}(\widehat{pk}, C_j, (c_1, \ldots, c_\ell))$$

and $c_i = \widehat{\text{Enc}}(\widehat{pk}, s_i)$. Therefore, by the correctness of the FHE scheme for decryption-multiplication, we can rewrite as

$$\begin{aligned} d &= \overline{\text{Enc}}\left(\overline{pk}, \sum_{j=1}^{k} \text{DEC\&Mult}\left((T_1, \ldots, T_n), d_j, 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j}\right)\right) \\ &= \overline{\text{Enc}}\left(\overline{pk}, \sum_{j=1}^{k} 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j} \cdot C_j(s_1, \cdots, s_\ell) + \underbrace{\sum_{j=1}^{k} e_j}_{\tilde{e}}\right). \end{aligned}$$

Let $r \leftarrow \mathbb{Z}_q$ and define the oracle $\mathcal{O}_{(\widehat{pk}, \overline{pk}, q, \tilde{q})}$ such that $a = \overline{\text{Enc}}(\overline{pk}, r)$ and

$$\tilde{g}(x_1, \ldots, x_n, x_{n+1}, x_{n+2}) = \text{DEC\&Mult}((x_1, \ldots, x_n), \tilde{a}, 1) + x_{n+1} + x_{n+2}.$$

Where, $\tilde{a} = \widehat{\text{Enc}}(\widehat{pk}, -\lfloor r/\tilde{q} \rfloor \cdot \tilde{q})$. Then by the correctness of the FHE scheme, and $c = \overline{\text{Enc}}(\overline{pk}, \tilde{s})$, where $\tilde{s}$ is

$$\begin{aligned} \tilde{s} &= \text{DEC\&Mult}\left((T_1, \ldots, T_n), \tilde{a}, 1\right) + \sum_{j=1}^{k} 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j} \cdot C_j(s_1, \cdots, s_\ell) + \tilde{e} + r \\ &= -\lfloor r/\tilde{q} \rfloor \cdot \tilde{q} + e + \sum_{j=1}^{k} 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j} \cdot C_j(s_1, \cdots, s_\ell) + \tilde{e} + r \\ &= \sum_{j=1}^{k} 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j} \cdot C_j(s_1, \cdots, s_\ell) + \tilde{e} + e + \underbrace{r \bmod \tilde{q}}_{\tilde{r}}. \end{aligned}$$

Note that an upper bound for $\tilde{e}+e$ is $(k+1)\cdot B$, and $\tilde{r}$ is a small perturbation due to the modulo $\tilde{q}$. This means that the output of the circuit is encoded as a high-order bit $\tilde{s}$ with probability 1 when q is sufficiently large. □

Theorem 11 (Security of Split Homomorphic Encryption Scheme)**.** *Let* $q \geq 2^k+2^{\lceil \log(\tilde{q}+(k+1)B)\rceil}$. *Assuming that the FHE scheme and the LHE scheme are secure schemes, then* **Scheme 5** *satisfies the security model* $\mathcal{O}_{(\widehat{pk},\overline{pk},q,\tilde{q})}$ *for split homomorphism.*

Proof. Assume $(s_0, s_1, C_1, \ldots, C_\beta)$ is the adversary's input chosen at the beginning of the generation of system π.

Hybrid $\mathcal{H}_0$: Define the following original system. The challenger generates a distribution using a random coin toss as follows:

$$(pk, c = \widehat{\mathrm{Enc}}(\widehat{pk}, s_\delta), \rho_1, \ldots, \rho_\beta).$$

Where

$$pk = (\widehat{pk}, \overline{pk}, \overline{\mathrm{Enc}}(\overline{pk}, T_1), \ldots, \overline{\mathrm{Enc}}(\overline{pk}, T_n)),$$

and ρ_i is obtained from $\mathrm{PDec}(sk, \mathrm{Eval}(pk, C_i, c))$.

Hybrids $\mathcal{H}_1, \ldots, \mathcal{H}_\beta$: Let $\mathrm{Eval}(pk, C_i, c)$ generate $d^{(i)}$. The ith *Hybrids* $\mathcal{H}_i$ is defined the same as *Hybrids* $\mathcal{H}_{i-1}$ except for the input $d^{(i)}$ and the output a (or $\tilde{a}$) such that

$$c = \overline{\mathrm{Enc}}\left(\overline{pk}, \mathrm{ECC}(C_i(s_\delta)) + \tilde{e} + e + r - \lfloor r/\tilde{q} \rfloor \cdot \tilde{q}\right),$$

where ECC is the high-order bit encoding defined in the homomorphic encryption part, $\tilde{e}+e$ is the decryption noise after homomorphic computation $(d^{(1)}, \ldots, d^{(k)}, \tilde{a})$, $r \leftarrow \mathbb{Z}_q$, $\tilde{\rho}_i$ is the "decryption tweak" obtained using random coin toss a, which can be used to decrypt the ciphertext c.

Note that the decryption noise $\tilde{e}+e$ can be efficiently calculated using the FHE scheme key, therefore $\tilde{\rho}_i$ can also be computed in polynomial time. The ciphertext distributions of *Hybrids* $\mathcal{H}_1, \ldots, \mathcal{H}_\beta$ are consistent, with the only difference being the specific form of $\tilde{\rho}_i$. This is because the LHE scheme has simulatable decryption tweaks, so the distribution of $\mathcal{H}_i$ is consistent with the distribution of $\mathcal{H}_{i-1}$, i.e.,

$$\begin{aligned}&(pk, \widehat{\mathrm{Enc}}(\widehat{pk}, s_\delta), \tilde{\rho}_1, \ldots, \tilde{\rho}_{i-1}, \rho_i, \rho_{i+1}, \ldots, \rho_\beta)\\ &= (pk, \widehat{\mathrm{Enc}}(\widehat{pk}, s_\delta), \tilde{\rho}_1, \ldots, \tilde{\rho}_{i-1}, \tilde{\rho}_i, \rho_{i+1}, \ldots, \rho_\beta).\end{aligned}$$

Hybrids $\mathcal{H}_{\beta+1}, \ldots, \mathcal{H}_{2\beta}$: The $\beta+i$th Hybrids and the previous β Hybrids are different mainly in a, i.e.,

$$\begin{aligned}c &= \overline{\mathrm{Enc}}\left(\overline{pk}, \mathrm{ECC}(C_i(s_\delta)) + \tilde{e} + e + \lfloor r/\tilde{q} \rfloor \cdot \tilde{q} + \tilde{r} - \lfloor r/\tilde{q} \rfloor \cdot \tilde{q}\right)\\ &= \overline{\mathrm{Enc}}\left(\overline{pk}, \mathrm{ECC}(C_i(s_\delta)) + \tilde{e} + e + \tilde{r}\right).\end{aligned}$$

Where, $\tilde{r} \leftarrow \mathbb{Z}_{\tilde{q}}$. Note that the distributions caused by these two Hybrids are different only when $r \in R$, where $R := \{q - (q \bmod \tilde{q}), \ldots, q\}$. Because $\tilde{q}/q \leq 2^{-\lambda}$, these two distributions to be statistically close.

Hybrids $\mathcal{H}_{2\beta+1}, \ldots, \mathcal{H}_{3\beta}$: The $2\beta + i$th Hybrids are defined the same as the previous ones, except for the value of a, i.e.,

$$c = \overline{\mathrm{Enc}}(\overline{pk}, \mathrm{ECC}(C_i(s_\delta)) + \tilde{y}).$$

Where the noise $\tilde{e}$ can be neglected in the calculation, therefore it is not reflected in the above equation. The difference between this and the previous Hybrids lies in whether the ciphertext contains $\tilde{e} + e$. Since an upper bound of the noise $\tilde{e} + e$ is $(k+1) \cdot B$, and $\tilde{q} \geq 2^\lambda \cdot (k+1) \cdot B$, according to Lemma 1, the distribution caused by this Hybrids is statistically indistinguishable from the previous one.

Hybrids $\mathcal{H}_{3\beta+1}, \ldots, \mathcal{H}_{3\beta+n}$: The $3\beta + i$th Hybrids are defined the same as the previous ones, except that the ciphertext $c_{(\mathrm{LHE},i)}$ is derived from encrypting 0 with the public key. At this point, the LHE scheme key no longer contributes to $(\tilde{\rho}_1, \ldots, \tilde{\rho}_\beta)$, so use indistinguishability to demonstrate the semantic security of these Hybrids.

$$\begin{pmatrix} \overline{\mathrm{Enc}}(\overline{pk}, 0), \ldots, \overline{\mathrm{Enc}}(\overline{pk}, 0), \overline{\mathrm{Enc}}(\overline{pk}, T_i), \\ \overline{\mathrm{Enc}}(\overline{pk}, T_{i+1}), \ldots, \overline{\mathrm{Enc}}(\overline{pk}, T_n) \end{pmatrix}$$
$$\approx_c \begin{pmatrix} \overline{\mathrm{Enc}}(\overline{pk}, 0), \ldots, \overline{\mathrm{Enc}}(\overline{pk}, 0), \overline{\mathrm{Enc}}(\overline{pk}, 0), \\ \overline{\mathrm{Enc}}(\overline{pk}, T_{i+1}), \ldots, \overline{\mathrm{Enc}}(\overline{pk}, T_n) \end{pmatrix}.$$

Hybrids $\mathcal{H}^{(0)}_{3\beta+n}, \ldots, \mathcal{H}^{(b)}_{3\beta+n}$: Fix the length of the challenge plaintext to i, and use the symbol $\mathcal{H}^{(i)}_{3\beta+n}$ to represent the Hybrids at this point. The distribution of this Hybrids is

$$(pk, c = \widehat{\mathrm{Enc}}(\widehat{pk}, s_i), \tilde{\rho}_1, \ldots, \tilde{\rho}_\beta),$$

where

$$pk = (\widehat{pk}, \overline{pk}, \overline{\mathrm{Enc}}(\overline{pk}, 0), \ldots, \overline{\mathrm{Enc}}(\overline{pk}, 0)).$$

Because the FHE scheme key is no longer encoded in the public parameters, there is no need to compute $(\tilde{\rho}_1, \ldots, \tilde{\rho}_\beta)$. Therefore, any advantage that the adversary has in distinguishing $\mathcal{H}^{(0)}_{3\beta+n}$ and $\mathcal{H}^{(1)}_{3\beta+n}$ cannot be greater than distinguishing $\widehat{\mathrm{Enc}}(\widehat{pk}, s_0)$ and $\widehat{\mathrm{Enc}}(\widehat{pk}, s_1)$. Therefore, the FHE scheme is computationally indistinguishable, thus proving the semantic security of the sFHE scheme. □

4.2 Instantiation of Oracle Model

To complete the description of our scheme, we discuss some candidate instantiations $\mathcal{O}_{(\widehat{pk}, \overline{pk}, q, \tilde{q})}$ of the oracle. We require the underlying LHE scheme to have a dense ciphertext space. We introduced the cyclic assumption introduced by Brakerski et al. [BDGM20] bridging the gap between FHE and LHE schemes. The oracle machine shown in Theorem 11 is just one of them, which is a special program obfuscation that enables the realization of split fully homomorphic schemes. Next, we introduce another oracle constructed by Brakerski et al. [BDGM20].

Simple Candidate Quantum Oracle. Let $\mathfrak{C}$ be the ciphertext space of LHE. The first instantiation is to take the encryption algorithm in FHE and encrypt the key in LHE, $\widehat{c} \leftarrow \widehat{\mathrm{Enc}}(\widehat{pk}, \overline{sk})$. Extract the ciphertext hash value of the homomorphic operation obtained through a hash function, which is used to fix the random coin in the algorithm. LHE ciphertext is sampled without knowing the underlying plaintext (which is why we need dense ciphertext), while FHE terms are calculated by homomorphically evaluating the decryption circuit and rounding the resulting message to the nearest multiple of $\tilde{q}$.

Let $D = (D_a)_{a \in \mathfrak{C}}$, where D_a is a set in the Hilbert space $\mathcal{H}_{D_a} = \mathbb{C}[\{0,1\}^n \cup \{\perp\}]$. The Hilbert space $\mathcal{H}_{D_a}$ can be seen as a space spanned by a set of orthogonal bases $|b\rangle$, where $b \in \{0,1\}^n \cup \{\perp\}$. Let the unitary transformation U be defined as

$$U|\perp\rangle = |\psi_0\rangle, U|\psi_0\rangle = |\perp\rangle \text{ and } U|\psi_b\rangle = |\psi_b\rangle, \forall b \in \{0,1\}^n \setminus \{0\}^n.$$

where $|\psi_b\rangle := H|b\rangle$, and H is the Hadamard transform on $\mathbb{C}[\{0,1\}^n] = (\mathbb{C}^2)^{\otimes n}$. Let $|b\rangle = 2^{-n/2} \sum_\eta (-1)^{\eta \cdot b} |\psi_\eta\rangle$, then we have

$$U|b\rangle = |b\rangle + 2^{-n/2}(|\perp\rangle - |\psi_0\rangle).$$

When the oracle is queried, the unitary transformation O_{XYZ} acts on the query register X and Y, and the database register D, with the specific expression

$$O_{XYZ} = \sum_a |a\rangle\langle a| \otimes O^a_{YD_a} \text{ and } O^a_{YD_a} = U_{D_a} \mathrm{CNOT}_{YD_a} U_{D_a}.$$

where $\mathrm{CNOT}|b\rangle|b_a\rangle = |b\rangle|b \oplus b_a\rangle$, $b, b_a \in \{0,1\}^n$ and $\mathrm{CNOT}|b\rangle|\perp\rangle = |b\rangle|\perp\rangle$. With these tools, present Don et al.'s quantum hash oracle model as follows:

$$y := \max_{a \in \mathfrak{C}} |\, \{b \in \{0,1\}^n | \langle a, b\rangle \in \mathbb{R}\} \,|, \quad \tilde{y} \leftarrow \widehat{\mathrm{Eval}}(\widehat{pk}, -\lfloor \overline{\mathrm{Dec}}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, \widehat{c})$$

Additionally, consider the following projector:

$$\mathit{\Pi}^a_{D_a} := \sum_{\substack{b \ s.t. \\ \langle a,b\rangle \in \mathbb{R}}} |b\rangle\langle b|_{D_a} \text{ and } \mathit{\Pi}^{\emptyset}_{D_a} := \mathbb{1}_D - \sum_{a \in \mathcal{X}} \mathit{\Pi}^a_{D_a} = \bigotimes_{a \in \mathcal{X}} \bar{\mathit{\Pi}}^a_{D_a}.$$

where $\bar{\mathit{\Pi}}^a_{D_a} := \mathbb{1}_{D_a} - \mathit{\Pi}^a_{D_a}$. Furthermore, define the measurement $\mathcal{M} = \mathcal{M}^R$, and the following projector

$$\Sigma^a := \bigotimes_{a' < a} \bar{\mathit{\Pi}}^{a'}_{D_{a'}} \otimes \mathit{\Pi}^a_{D_a} \text{ and } \Sigma^{\emptyset} := \mathbb{1} - \sum_{a'} \Sigma^{a'} = \bigotimes_{a'} \bar{\mathit{\Pi}}^{a'}_{D_{a'}} = \mathit{\Pi}^{\emptyset}.$$

In addition, define the pure state measurement unitary transformation $M_{DP} = M^R_{DP} \in L(\mathcal{H}_D \otimes \mathcal{H}_R)$, i.e.,

$$M_{DP} := |\varphi\rangle_D |w\rangle_P \mapsto |\varphi\rangle_D |w + a\rangle_P.$$

Note that y is an element in the ciphertext domain of LHE, and its form is $y = \overline{\mathrm{Enc}}(\overline{pk}, s)$. For some $s \in \mathbb{Z}_q$, because LHE has a dense ciphertext domain. Furthermore, through the correctness of the FHE and LHE schemes, we have

$$\begin{aligned}\tilde{y} &= \widehat{\mathrm{Eval}}(\widehat{pk}, -\lfloor \overline{\mathrm{Dec}}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, \widehat{c}) \\ &= \widehat{\mathrm{Eval}}(\widehat{pk}, -\lfloor \overline{\mathrm{Dec}}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, \widehat{\mathrm{Enc}}(\widehat{pk}, \overline{sk})) \\ &= \widehat{\mathrm{Enc}}(\widehat{pk}, -\lfloor \overline{\mathrm{Dec}}(\overline{sk}, y)/\tilde{q} \rfloor \cdot \tilde{q}) \\ &= \widehat{\mathrm{Enc}}(\widehat{pk}, -\lfloor s/\tilde{q} \rfloor \cdot \tilde{q}).\end{aligned}$$

Therefore, it can be seen that the formation of $(y, \tilde{y})$ is based on the following assumptions.

Alternating Encryption Security. The cyclic dependency introduced by $\widehat{c} = \widehat{\mathrm{Enc}}(\widehat{pk}, \overline{sk})$ in the security of LHE and FHE schemes (e.g., the split FHE construction in this paper includes the encryption of $\widehat{sk}$ under $\overline{pk}$ in the public key) is considered a very mild assumption. Currently, it is the only known method to construct FHE from the LWE problem through bootstrapping theorems [Gen09].

Perturbation. In the case of $y := \max_{a \in \mathfrak{C}} |\{b \in \{0,1\}^n | \langle a, b \rangle \in \mathbb{R}\}|$, although $\tilde{y}$ is an FHE encryption of the correct value, it is not necessarily uniformly distributed. In particular, the randomness of $\tilde{y}$ may depend on the low-order bits of s in a complex way. In the specific case of LWE-based schemes, the noise term may carry information about s modulo $\tilde{q}$, which may introduce perturbation that interferes with decryption. However, the noise function is usually highly nonlinear, making it difficult to exploit. Therefore, we only consider the FHE.Eval algorithm.

Perturbation Elimination. Regarding the methods for eliminating the perturbation in LHE and FHE ciphertexts, we naturally think of ciphertext reprocessing techniques [DS16]: it can be expected that repeating bootstraping operations on FHE ciphertexts can eliminate the perturbation from LHE ciphertext noise. Unfortunately, our setting is different from the typical settings considered in the literature, as the ciphertext perturbation reprocessing algorithm must be executed by the distinguisher and cannot use private random coins. Although it seems difficult to formally analyze the effectiveness of these methods in our setting, we hope that these techniques may (at least heuristically) help mitigate the perturbation that interferes with decryption. This paper takes a different approach and provides a simple heuristic to alleviate perturbation. In short, the idea is to sample a set of random plaintexts and define a random string as the sum of a uniform subset $\mathfrak{S}$ of these plaintexts. For the construction described earlier, Brakerski et al.'s instantiation includes a ciphertext $\widehat{c} = \widehat{\mathrm{Enc}}(\widehat{pk}, \overline{sk})$. The parameter $\sigma \in \mathrm{poly}(n, m, q, p)$ of the scheme is determined by the length of the set $\mathfrak{S}$. The algorithm is presented randomly below, although this simplification can be easily bypassed using standard techniques (e.g., computing random coins using encrypted $\mathrm{Hash}(x)$).

$\mathcal{O}(\widehat{pk}, \overline{pk}, q, \tilde{q})(x)$: Input string $x \in \{0,1\}^*$ and a random set $\mathfrak{S} \leftarrow \{0,1\}^\sigma$. For all $i \in [\sigma]$, when $\mathfrak{S}_i = 1$, uniformly output sample $y_i := \max_{a \in \mathfrak{C}} |\{b \in \{0,1\}^n | \langle a, b \rangle \in \mathbb{R}\}|$; when $\mathfrak{S}_i = 0$, uniformly output sample $y_i \leftarrow \overline{\mathrm{Enc}}(\overline{pk}, s_i)$, where s_i is any known plaintext message. Then compute

$$\tilde{y} \leftarrow \widehat{\mathrm{Eval}}\left(\widehat{pk}, -\sum_{i=1}^{\sigma} \lfloor \overline{\mathrm{Dec}}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, \widehat{c}\right).$$

Let g be a linear function defined as follows

$$g(x_1, \ldots, x_{\mathfrak{S}}) = \sum_{i \in \mathfrak{S}} x_i + \sum_{i \notin \mathfrak{S}} \lfloor x_i/\tilde{q} \rfloor \cdot \tilde{q}.$$

Then compute $\tilde{y} \leftarrow \overline{\mathrm{Eval}}(\overline{pk}, g, \{y_i\}_{i \in \mathfrak{S}})$ and return $(y, \tilde{y})$. By the correctness of homomorphic operations in the FHE scheme, it shown that

$$\begin{aligned}
\tilde{y} &= \widehat{\mathrm{Eval}}\left(\widehat{pk}, -\sum_{i=1}^{\sigma} \lfloor \overline{\mathrm{Dec}}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, \widehat{c}\right) \\
&= \widehat{\mathrm{Eval}}\left(\widehat{pk}, -\sum_{i=1}^{\sigma} \lfloor \overline{\mathrm{Dec}}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, \widehat{\mathrm{Enc}}(\widehat{pk}, \overline{sk})\right) \\
&= \widehat{\mathrm{Enc}}\left(\widehat{pk}, -\sum_{i=1}^{\sigma} \lfloor \overline{\mathrm{Dec}}(\overline{sk}, y)/\tilde{q} \rfloor \cdot \tilde{q}\right) \\
&= \widehat{\mathrm{Enc}}\left(\widehat{pk}, -\sum_{i=1}^{\sigma} \lfloor s/\tilde{q} \rfloor \cdot \tilde{q}\right).
\end{aligned}$$

Combining with the correctness of the LHE scheme, one obtain

$$\begin{aligned}
y &= \overline{\mathrm{Eval}}(\overline{pk}, g, \{y_i\}_{i \in \mathfrak{S}}) \\
&= \overline{\mathrm{Eval}}(\overline{pk}, g, \{\overline{\mathrm{Enc}}(\overline{pk}, s_i)\}_{i \in \mathfrak{S}}) \\
&= \overline{\mathrm{Enc}}\left(\overline{pk}, \sum_{i \in \mathfrak{S}} s_i + \sum_{i \notin \mathfrak{S}} \lfloor s_i/\tilde{q} \rfloor \cdot \tilde{q}\right) \\
&= \overline{\mathrm{Enc}}\left(\overline{pk}, \underbrace{\sum_{i \in \mathfrak{S}} (s_i \bmod \tilde{q})}_{\tilde{s}} + \sum_{i \notin \mathfrak{S}} \lfloor s_i/\tilde{q} \rfloor \cdot \tilde{q}\right).
\end{aligned}$$

5 Constructing Ideal Obfuscation using Homomorphic Splitting Encryption Scheme

5.1 Ideal Obfuscation

Scheme 6 Ideal Obfuscation Scheme

KeyGen(n, m, q). For $i \in [0, D)$, $j \in [0, B]$, randomly sample $k_{i,j} \leftarrow \{0,1\}^{\lambda}$ and compute

$$h_{i,j} = \mathrm{Pr}\mathcal{O}(k_{i.j}, x).$$

Randomly sample $s_\varepsilon \leftarrow \{0,1\}^\lambda$. For $d \in [0, D]$, input security parameter n, output sample $(\overline{sk}_d, \overline{pk}_d) \leftarrow \overline{\text{KeyGen}}(n)$. Let $\mathbb{Z}_q$ be the plaintext space under LHE definition, output sample $(\widehat{sk}_d, \widehat{pk}_d) \leftarrow \widehat{\text{KeyGen}}(n, m, q)$. Let $\widehat{sk}_d = (T_1, \ldots, T_n) \in \{0,1\}^{n \times n}$, then return

$$sk_d = \overline{sk}_d \text{ and } pk_d = (\widehat{pk}_d, \overline{pk}_d, \bar{c}_1, \ldots, \bar{c}_n).$$

where, for any $i \in [n]$, we define $\bar{c}_i \leftarrow \overline{\text{Enc}}(\overline{pk}_d, T_i)$.

$\mathbf{Enc}(\boldsymbol{pk_d}, \mathbf{info}_{\boldsymbol{\varepsilon}})$. For input $\text{info}_\varepsilon = (\text{normal}, \varepsilon, \{k_{i,j}\}_{i \in [0,D), j \in [1,B]}, s_\varepsilon)$, return

$$ct_\varepsilon \leftarrow \widehat{\text{Enc}}(\widehat{pk}_d, \text{info}_\varepsilon).$$

$\mathbf{Eval}(\boldsymbol{pk_d}, \boldsymbol{f_d}, (\boldsymbol{c_1}, \ldots, \boldsymbol{c_\ell}))$. f_d is provided later. Input circuit $\mathcal{C}$ of ℓ bits and ciphertext of length k bits $(c_1, \ldots, c_\ell)$. For any $j \in [k]$, where $\mathcal{C}_j$ is the j-th component of circuit $\mathcal{C}$, compute

$$\dot{d}_j \leftarrow \widehat{\text{Eval}}(\widehat{pk}_d, C_j, (c_1, \ldots, c_\ell)).$$

Define linear function over $\mathbb{Z}_q$ as

$$g(x_1, \ldots, x_n) = \sum_{j=1}^{k} \text{DEC\&Mult}\left((x_1, \ldots, x_n), \dot{d}_j, 2^{\lceil \log(\tilde{q} + (k+1)B) \rceil + j}\right).$$

Compute $\dot{d} \leftarrow \overline{\text{Eval}}(\overline{pk}_d, g, (\bar{c}_1, \ldots, \bar{c}_n))$, then query $(a, \tilde{a}) \leftarrow \mathcal{O}_{(\widehat{pk}_d, \overline{pk}_d, q, \tilde{q})}(\dot{d})$ and define the following linear function

$$\tilde{g}(x_1, \ldots, x_n, x_{n+1}, x_{n+2}) = \text{DEC\&Mult}((x_1, \ldots, x_n), \tilde{a}, 1) + x_{n+1} + x_{n+2}.$$

Output

$$ct_\varepsilon \leftarrow \overline{\text{Eval}}(\overline{pk}_d, \tilde{g}, (\bar{c}_1, \ldots, \bar{c}_n), \dot{d}, a).$$

Return the obfuscated circuit

$$\widehat{C} = (\{h_{i,j}\}_{i \in [0,D), j \in [1,B]}, ct_\varepsilon, \{sk_d\}_{d \in [0,D]}).$$

Eval&Expand. (normal mode)

- For $d \in [0, D)$, Eval&Expand encrypts $f_d(\text{normal}, \chi, \{k_{i,j}\}_{i \in [0,D), j \in [1,B]}, s_\chi)$
  1. Compute $s_{\chi\|0} \| r_{\chi\|0} \| s_{\chi\|1} \| r_{\chi\|1} \leftarrow G(s_\chi)$.
  2. For $b \in \{0,1\}$, run $ct_{\chi\|b} \leftarrow \widehat{\text{Enc}}(\widehat{pk}_{d+1}, \text{info}_{\chi\|b}; r_{\chi\|b})$. where,

     $$\text{info}_{\chi\|b} = (\text{normal}, C, \chi\|b, \{k_{i,j}\}_{i \in [d+1,D), j \in [1,B]}, s_\chi\|b),$$

     C is the circuit to be obfuscated. Output

     $$(H(k_{d,1}, \chi) \| \cdots \| H(k_{d,B}, \chi)) \oplus (ct_{\chi\|0} \| ct_{\chi\|1}).$$

- For $d = D$, $f_D(\text{normal}, C, x, s_x)$, output $C(x)$.

$$\widehat{C}^{\mathcal{O}}[ct_\varepsilon, \{sk_d\}_{d\in[0,D]}, \{h_{i,j}\}_{i\in[0,D),j\in[0,B]}](x)$$

Hardwired. ct_ε, initial ciphertext.

sk_d, secret key.

$h_{i,j}$, handles generated by Pr$\mathcal{O}$M.

Input. $x \in \{0,1\}^D$, input circuit.

Output. Compute as follows.

```
**For** $d = 0, \dots, D-1$:
    $\chi_d \leftarrow x_{\leq d}$
    $\nu_{\chi_d} \leftarrow \overline{\mathrm{Rec}}(\rho_{\chi_d}, ct_{\chi_d})$, $\rho_{\chi_d} \leftarrow \overline{\mathrm{PDec}}(sk_d, ct_{\chi_d})$
    $\mathrm{otp}_{\chi_d} \leftarrow \mathcal{O}(\mathrm{hEval}, h_{d,1}, \chi_d \| 0^{D-d}) \| \cdots \| \mathcal{O}(\mathrm{hEval}, h_{d,B}, \chi_d \| 0^{D-d})$
    $ct_{\chi_d\|0} \| ct_{\chi_d\|1} \leftarrow \nu_{\chi_d} \oplus \mathrm{otp}_{\chi_d}$
Output $\mathrm{Dec}(sk_D, ct_x)$
```

Figure 4: Obfuscated Circuit $(\widehat{C}^{\mathcal{O}}) \to \widehat{C}^{\bullet}[ct_x, \{sk_d\}_{d\in[0,D]}, \{h_{i,j}\}_{i\in[0,D),j\in[0,B]}]$

Correctness Analysis. According to the obfuscation form $\widehat{C}^{\mathcal{O}}$ in Figure 5 and the tree structure in Figure 6.

$$\begin{aligned} &\vec{H}(k_{d,1}, \chi_d \| 0^{D-d}) \| \cdots \| H(k_{d,B}, \chi_d \| 0^{D-d}) \\ =\ &\mathcal{O}(\mathrm{hEval}, h_{d,1}, \chi_d \| 0^{D-d}) \| \cdots \| \mathcal{O}(\mathrm{hEval}, h_{d,B}, \chi_d \| 0^{D-d}). \end{aligned}$$

5.2 Security Analysis

Lemma 4. *Assuming H is a pseudo-random function, G_{sr}, G_v are pseudo-random generators, and* (Gen, Enc, Enc) *is adaptively secure, with appropriate parameters L and B, then Construction 1 in [JLLW23] is an ideal obfuscation under* Pr$\mathcal{O}$M.

Theorem 12. *Assuming H is a pseudo-random function, G_{sr}, G_v are pseudo-random generators, algorithm 6 is an ideal obfuscation under* Pr$\mathcal{O}$M.

```
$\text{Expand}_{d,\text{hyb}}[pk_{d+1}](\chi, \text{info}_\chi)$

Hardwired. $pk_{d+1}$, public key at level $(d+1)$.
    Input. $x \in \{0,1\}^d$, input appropriate circuit;
           $\text{info}_\chi = (C, \{k_{i,j}\}_{i\in(d,D),j\in[1,B]}, s_\chi, \beta, \{\sigma_{\chi,j}\}_{j\in[0.\beta)}, w_\chi, \{k_{d,j}\}_{j\in(\sigma,B]})$:
               $C$, circuit to be obfuscated.
               $k_{i,j}$, keys of $H$ at levels $(d+1,\dots,D-1)$.
               $s_\chi$, seed of pseudo-random generator $G_{sr}$, related to $\chi$.
               $\beta$, mixing index.
               $\sigma_{\chi,j}$, seed of pseudo-random generator $G_v$, related to $\chi$.
               $w_\chi$, decryption result of the software module.
               $k_{d,j}$, keys of $H$ at level $(d+1)$.
   Output. Calculated as follows.

           $s_{\chi\|0} \| r_{\chi\|0} \| s_{\chi\|1} \| r_{\chi\|1} \leftarrow G_{sr}(s_\chi)$
           **For** $\eta = 0, 1$:
               $\text{flag}_{\chi\|\eta} \leftarrow \text{normal}$
               $\text{info}_{\chi\|\eta} \leftarrow (C, \{k_{i,j}\}_{i\in[d+1,D),j\in[1,B]}, s_{\chi\|\eta})$
               $ct_\varepsilon \leftarrow \text{Enc}(pk_{d+1}, \text{flag}_{\chi\|\eta}, \chi\|\eta, \text{info}_{\chi\|\eta})$
           Output $\nu_\chi \leftarrow G_\nu(\sigma_\chi, 1) \| \cdots \| G_\nu(\sigma_\chi, \beta-1) \| w_\chi$
               $\|([ct_{\chi\|0} \| ct_{\chi\|1}]_{\beta+1} \oplus H(k_{d,\beta+1}, \chi\|0^{D-d})) \| \cdots$
               $\|([ct_{\chi\|0} \| ct_{\chi\|1}]_B \oplus H(k_{d,B}, \chi\|0^{D-d}))$
```

Figure 5: Obfuscation circuit $(\widehat{C}^{\mathcal{O}}) \to \widehat{C}^{\bullet}[ct_x, \{sk_d\}_{d\in[0,D]}, \{h_{i,j}\}_{i\in[0,D),j\in[0,B]}]$

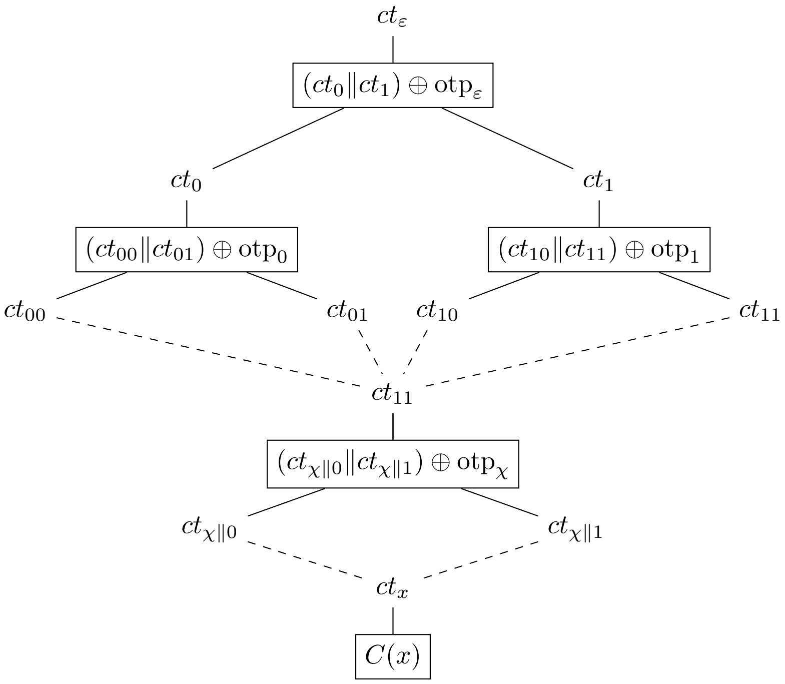
ct_ε
$(ct_0\|ct_1)\oplus \mathrm{otp}_\varepsilon$
ct_0
ct_1
$(ct_{00}\|ct_{01})\oplus \mathrm{otp}_0$
$(ct_{10}\|ct_{11})\oplus \mathrm{otp}_1$
ct_{00}
ct_{01}
ct_{10}
ct_{11}
ct_{11}
$(ct_{\chi\|0}\|ct_{\chi\|1})\oplus \mathrm{otp}_\chi$
$ct_{\chi\|0}$
$ct_{\chi\|1}$
ct_x
$C(x)$

Figure 6: The binary tree of ciphertexts [JLLW23] in Scheme 6

$\text{Expand}_d[pk_{d+1}](\text{flag}_\chi, \chi, \text{info}_\chi)$ – For Level $d \in [0, D)$

Hardwired. pk_{d+1}, $(d+1)$ level public key.

Input. $\text{flag}_\chi \in \{\text{normal}, \text{hyb}, \text{sim}\}$, flag matching with χ.

$\chi \in \{0,1\}^d$, input appropriate prefix for the circuit.

info_χ, information matching with χ, changes with flag_χ.

Output.

$$\begin{cases} \text{Expand}_{d,\text{normal}}[pk_{d+1}](\chi, \text{info}_\chi), & \text{if flag}_\chi = \text{normal};\\ \left.\begin{array}{ll} \text{Expand}_{d,\text{hyb}}[pk_{d+1}](\chi, \text{info}_\chi), & \text{if flag}_\chi = \text{hyb};\\ \text{Expand}_{d,\text{hyb}}(\chi, \text{sim}_\chi), & \text{if flag}_\chi = \text{sim}. \end{array}\right\}\text{Fig.7} \end{cases}$$

$\text{Eval}(\text{flag}_\chi, \chi, \text{info}_\chi)$ – For Level D

Hardwired. pk_{d+1}, $(d+1)$ level public key.

Input. $\text{flag}_\chi \in \{\text{normal}, \text{sim}\}$, flag matching with χ.

$\chi \in \{0,1\}^D$, input circuit.

info_χ, information matching with χ, changes with flag_χ.

Output.

$$\begin{cases} \text{Eval}_{d,\text{normal}}(\chi, \text{info}_\chi), & \text{if flag}_\chi = \text{normal};\\ \text{Eval}_{d,\text{sim}}(\chi, \text{info}_\chi), & \text{if flag}_\chi = \text{sim}. \end{cases}$$

$\text{Expand}_{d,\text{normal}}[pk_{d+1}](\chi, \text{info}_\chi)$

Hardwired. pk_{d+1}, $(d+1)$ level public key.

Input. $\chi \in \{0,1\}^d$, input appropriate prefix for the circuit.

$\text{info}_\chi = (C, \{k_{i,j}\}_{i \in [0,D), j \in [0,B]}, s_\chi)$:

- C, circuit to be obfuscated.
- $k_{i,j}$, keys for hash functions $d, \dots, D-1H$.
- s_χ, seed for pseudo-random generator G_{sr} corresponding to χ.

Output. Perform the following calculations.

```
$s_{\chi\|0} \| r_{\chi\|0} \| s_{\chi\|1} \| r_{\chi\|1} \leftarrow G_{sr}(s_\chi)$.
for $\eta = 0, 1$:
    $\text{flag}_{\chi\|\eta} \leftarrow \text{normal}$
    $\text{info}_{\chi\|\eta} \leftarrow (C, \{k_{i,j}\}_{i \in [d+1,D), j \in [1,B]}, s_{\chi\|\eta})$.
    $ct_{\chi\|\eta} \leftarrow \text{Enc}(pk_{d+1}, \text{flag}_{\chi\|\eta}, \chi\|\eta, \text{info}_{\chi\|\eta})$
$\text{otp}_\chi \leftarrow H(k_{d,1}, \chi\|0^{D-d}\| \cdots \|H(k_{d,B}, \chi\|0^{D-d})$
Output $\nu_\chi \leftarrow (ct_{\chi\|0} \| ct_{\chi\|1}) \oplus \text{otp}_\chi$
```

$\text{Eval}_{\text{normal}}(\chi, \text{info}_\chi)$

Input. $\chi \in \{0,1\}^D$, input circuit.

$\text{info}_\chi = (C, s_\chi)$:

- C, circuit to be obfuscated.
- s_χ, unused seed.

Output. $C(\chi)$, compute the evaluation of a generalized circuit (C, χ).

Figure 7: The circuits Expand&Eval$_d$ in Scheme 6

References

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In *Advances in Cryptology – CRYPTO 2015*, pages 308–326. Springer Berlin Heidelberg, 2015.

[AJLA$^+$12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold fhe. In *Proceedings of the 31st Annual International Conference on Theory and Applications of Cryptographic Techniques*, page 483 - 501. Springer-Verlag, 2012.

[AKPW13] Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited. In *Advances in Cryptology – CRYPTO 2013*, pages 57–74. Springer Berlin Heidelberg, 2013.

[BDGM19] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In *Theory of Cryptography: 17th International Conference*, page 407 - 437. Springer-Verlag, 2019.

[BDGM20] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate io from homomorphic encryption schemes. In *Advances in Cryptology – EUROCRYPT 2020*, pages 79–109. Springer International Publishing, 2020.

[BDGM22] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and Pairings Are Not Necessary for IO: Circular-Secure LWE Suffices. In *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, volume 229, pages 1–20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

[BGI$^+$01] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology — CRYPTO 2001*, pages 1–18. Springer Berlin Heidelberg, 2001.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *Advances in Cryptology – EUROCRYPT 2012*, pages 719–737. Springer Berlin Heidelberg, 2012.

[BV18] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. *Journal of the ACM*, 65(6):1 - 37, 2018.

[BZ17] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. *Algorithmica*, 79:1233–1285, 2017.

[CLT13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *Advances in Cryptology – CRYPTO 2013*, pages 476–493. Springer Berlin Heidelberg, 2013.

[DFMS22] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Online-extractability in the quantum random-oracle model. In *Advances in Cryptology – EUROCRYPT 2022*, pages 677–706. Springer International Publishing, 2022.

[DS16] Léo Ducas and Damien Stehlé. Sanitization of fhe ciphertexts. In *Advances in Cryptology – EUROCRYPT 2016*, pages 294–310. Springer Berlin Heidelberg, 2016.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, page 169 - 178. Association for Computing Machinery, 2009.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology – EUROCRYPT 2013*, pages 1–17. Springer Berlin Heidelberg, 2013.

[GGH$^+$13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 40–49, 2013.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *Theory of Cryptography*, pages 498–527. Springer Berlin Heidelberg, 2015.

[GGHR14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure mpc from indistinguishability obfuscation. In *Theory of Cryptography*, pages 74–94. Springer Berlin Heidelberg, 2014.

[Had00] Satoshi Hada. Zero-knowledge and code obfuscation. In *Advances in Cryptology — ASIACRYPT 2000*, pages 443–457. Springer Berlin Heidelberg, 2000.

[HJ16] Yupu Hu and Huiwen Jia. Cryptanalysis of ggh map. In *Advances in Cryptology – EUROCRYPT 2016*, pages 537–565. Springer Berlin Heidelberg, 2016.

[JLLW23] Aayush Jain, Huijia Lin, Ji Luo, and Daniel Wichs. The pseudorandom oracle model and ideal obfuscation. In *Advances in Cryptology – CRYPTO 2023*, pages 233–262, Cham, 2023. Springer Nature Switzerland.

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, page 60 - 73. Association for Computing Machinery, 2021.

[Lin17] Huijia Lin. Indistinguishability obfuscation from sxdh on 5-linear maps and locality-5 prgs. In *Advances in Cryptology - CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 599–629. Springer, 2017.

[LT17] Huijia Lin and Stefano Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local prgs. In *Advances in Cryptology – CRYPTO 2017*, pages 630–660. Springer International Publishing, 2017.

[Luo23] Ji Luo. The pseudorandom oracle model and ideal obfuscation. Working Paper, https://luoji.bio/assets/slides/JLLW22zh.pdf, 2023.

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over ggh13. In *Advances in Cryptology – CRYPTO 2016*, pages 629–658. Springer Berlin Heidelberg, 2016.

[SW21] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. *SIAM Journal on Computing*, 50(3):857 – 908, 2021.

[XXZ12] Xiang Xie, Rui Xue, and Rui Zhang. Deterministic public key encryption and identity-based encryption from lattices in the auxiliary-input setting. In *Proceedings of the 8th International Conference on Security and Cryptography for Networks*, page 1 – 18. Springer-Verlag, 2012.

[Yue20] Steven Yue. Introduction to io 01: What is indistinguishability obfuscation (io)?, 2020.

[Zha19] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In *Advances in Cryptology – CRYPTO 2019*, page 239 – 268. Springer-Verlag, 2019.