Diamond iO: Indistinguishability Obfuscation at Near Functional-Encryption Cost via Matrix Products

Sora Suegami[1][0009−0008−7107−3676]⋆, Enrico Bottazzi[2][0009−0009−4150−5885]⋆⋆, and Gayeong Park[3]⋆⋆⋆

[1] Machina iO, Ethereum Foundation sora.suegami@ethereum.org
[2] Leku enrico@leku.ink
[3] Independent Researcher piapark314@gmail.com

Abstract. Indistinguishability obfuscation (iO) remains far from practical. A major source of inefficiency in recent iO schemes is the transformation from functional encryption (FE) to iO: evaluating the obfuscated circuit requires recursively invoking the FE encryption algorithm for each input bit. This FE-to-iO overhead creates a large efficiency gap between the underlying FE scheme and the resulting iO scheme.

We propose Diamond iO, a lattice-based iO scheme based on a new FE-to-iO transformation that replaces this recursive FE encryption with simple matrix products. This makes the transformation cost asymptotically and concretely smaller than the cost of the underlying FE scheme, enabling iO at nearly the efficiency of the FE scheme itself. For obfuscated PRFs on L-bit inputs, using the same underlying FE scheme, the transformation-to-decryption cost ratio improves from $\Omega(L^5)$ in the prior transformation to $\Theta(1/L^2)$ in Diamond iO. Our concrete efficiency estimates show at least a 2^{118}-fold improvement in the estimated transformation running time under the same comparison setting, with our transformation step taking less than 10^{-31} % of the estimated total evaluation running time.

Our transformation applies to FE schemes with a special ciphertext structure, formalized as path-factorized functional encryption (PFFE). PFFE yields indistinguishable pseudorandom obfuscation (iPRO), and hence iO for general circuits in the pseudorandom oracle model. Instantiating our transformation with our PFFE construction yields an iPRO scheme from LWE, private-coin evasive LWE, and a new assumption called all-product LWE. We explain why all-product LWE avoids the known rank attack of Chen et al. (TCC'19). Additionally, our noise-growth optimization for PFFE decryption enables concrete end-to-end efficiency estimates with realistic lattice parameters. Although PFFE decryption remains the dominant cost, Diamond iO removes the FE-to-iO transformation bottleneck, advancing the path toward practical iO.

⋆ Corresponding author.
⋆⋆ Work done while at the Ethereum Foundation.
⋆⋆⋆ Work done while affiliated with Korea University and supported by a grant from the Ethereum Foundation.

Keywords: indistinguishability obfuscation · functional encryption · evasive LWE · lattices · key-homomorphic encodings

1 Introduction

Indistinguishability obfuscation (iO) is a general-purpose notion of program obfuscation that guarantees that obfuscations of any two same-size circuits with the same functionality are computationally indistinguishable [11]. This has been seen as a "central hub [75]" for its role in building numerous powerful cryptographic primitives [75, 22, 53, 14, 13, 5].

The dominant route to modern iO schemes proceeds through functional encryption (FE). FE is a generalization of public-key encryption in which a functional secret key associated with a public function f allows its holder to decrypt an encryption of an input x and learn only $f(x)$ [21]. Ananth and Jain [8] and Bitansky and Vaikuntanathan [15] provided black-box transformations from FE having succinct encryption complexity to iO. Since then, the FE-to-iO paradigm has served as the common backbone of much of the modern iO literature, ultimately leading to constructions from well-founded assumptions [65, 66, 73] and plausibly post-quantum candidates from lattice-based assumptions [27, 26, 43, 46, 80, 2, 29, 3, 58].

Despite this theoretical progress, iO schemes within the FE-to-iO paradigm remain far from concretely efficient. A major source of inefficiency is the transformation overhead introduced by existing FE-to-iO transformations themselves. In particular, existing transformations [8, 15, 2] rely on a recursive encryption step: for each input bit, the function evaluated during FE decryption invokes the FE encryption algorithm to produce the next ciphertext. Consequently, the size and complexity of FE encryption become a lower bound on the complexity of the functions that the underlying FE scheme must support. This overhead is imposed by the transformation, and arises even when the target circuit to be obfuscated is very small. This raises the following natural question for concretely efficient iO within the FE-to-iO paradigm:

Can we obtain iO at nearly the concrete cost of the underlying FE scheme?

1.1 Our Results and Contributions

Our results. Our answer is affirmative. We propose *Diamond iO*, a lattice-based iO scheme based on a new FE-to-iO transformation whose transformation overhead is implemented by simple matrix products rather than recursive FE encryption. More precisely, Diamond iO constructs indistinguishable pseudorandom obfuscation (iPRO), namely iO restricted to pseudorandom functionalities. By the method of [29], this is bootstrapped to iO for general circuits in the pseudorandom oracle model (PROM) [64].

Table 1 shows that, with respect to the input size L of the circuit being obfuscated[4], the prior transformation [2] incurs an FE-to-iO overhead of at least

[4] Here, L denotes the input size of the circuit, not its circuit size.

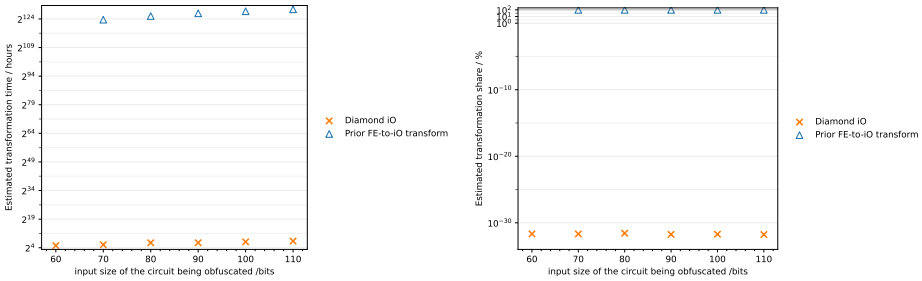
Table 1. Asymptotic comparison of the cost of evaluating obfuscated circuits for PRFs with input size L and one output bit, comparing the prior FE-to-iO transformation for pseudorandom functionalities of [2] with Diamond iO, measured by the number of fixed-width integer multiplications. The prior transformation instantiates FE decryption with the FE scheme of [1], while the Diamond iO row uses the corresponding variant of that FE scheme. The FE scheme is assumed to be instantiated with a PRF whose multiplicative depth grows linearly with L; we do not consider logarithmic-depth PRFs here because, as described in Subsection 5.1, we found it difficult to adopt such a PRF while keeping the lattice parameters in a practical range. This asymptotic efficiency analysis does not take into account the implementation optimizations described from Section 5 onward; it computes the asymptotic efficiency only from the minimal assumptions needed for the concrete constructions. The column $\mathsf{cost}_{\mathsf{FE\text{-}to\text{-}iO}}$ is the cost of the transformation itself. The last column gives the transformation cost normalized by the final FE decryption cost. The details of these asymptotic cost estimates are given in Appendix A.

FE-to-iO Transformation	$\mathsf{cost}_{\mathsf{FE\text{-}to\text{-}iO}}$	$\frac{\mathsf{cost}_{\mathsf{FE\text{-}to\text{-}iO}}}{\mathsf{cost}_{\mathsf{FE.Dec}}}$
Prior	$\Omega(L^{15})$	$\Omega(L^5)$
Diamond iO	$\Theta(L^8)$	$\Theta\left(\frac{1}{L^2}\right)$

$\Omega(L^2)$ relative to the final FE decryption, whereas Diamond iO reduces this relative overhead to $\Theta(1/L^2)$, making the FE-to-iO transformation asymptotically lower-order than the final FE decryption. Moreover, when the prior transformation uses a PRF whose multiplicative depth grows linearly with L, the FE-to-iO transformation cost of Diamond iO is smaller by at least a factor of L^7 compared with that of the prior transformation. Although the prior transformation becomes asymptotically more efficient for sufficiently large L when the FE scheme is instantiated with a PRF whose depth grows logarithmically in L, we found it difficult to implement such a PRF within the range of practical lattice parameters as described in Subsection 5.1.

We implement an estimator for the end-to-end evaluation cost of Diamond iO that accounts for the underlying FE scheme, and report estimated running times for evaluating Diamond iO obfuscations up to $L = 110$ input bits.[5] Our estimates shown in Figure 1 demonstrate that the estimated transformation running time of Diamond iO is at least 2^{118} times smaller than that of the prior transformation, and that the transformation overhead of Diamond iO is negligible within the estimated total iO evaluation running time, whereas the overhead of the prior transformation accounts for almost all of its estimated total evaluation running time.

[5] We choose $L = 110$ input bits as the largest input size in our estimates because it is already sufficient for several applications. For example, at the 100-bit security level, constructing public-key encryption from one-way functions and iO requires at least 101 input bits [75].



(a) Estimated running time of the FE-to-iO transformation.

(b) Estimated FE-to-iO transformation share in total iO evaluation running time.

Fig. 1. Concrete efficiency estimates for evaluating Diamond iO obfuscations, compared with the prior FE-to-iO transformation for pseudorandom functionalities of [2]. The comparison setting is the same as that of Table 1, where both transformations are instantiated with the same underlying FE scheme to evaluate obfuscated PRFs on L-bit inputs. In both panels, **smaller vertical-axis values indicate better efficiency**. For each input size L, the modulus q is set to the value estimated by our implementation to be sufficiently large for correctness, and the lattice dimension is the smallest power of two that the lattice estimator [6] reports as providing 100-bit security for that modulus q. The details of these estimates are given in Section 5.

Our contributions. We organize our contributions into the following three parts.

(i) A new FE-to-iO transformation via matrix products. Our first contribution is a new FE-to-iO transformation that replaces the recursive-encryption step of prior transformations with simple matrix products. This transformation applies to FE schemes whose ciphertexts have a path-factorized structure: by multiplying the matrices selected according to the evaluator's input bits, the evaluator obtains, up to small noise, an FE ciphertext corresponding to its input.

We abstract such FE schemes as *path-factorized functional encryption* (PFFE), and show that any PFFE scheme satisfying the required properties yields iPRO. The security of Diamond iO relies on LWE and private-coin evasive LWE, in addition to the assumptions used to instantiate the underlying PFFE scheme.

(ii) Lattice-based PFFE constructions. Our second contribution is to construct PFFE from lattices. We do so by modifying the lattice-based FE scheme of Agrawal et al. [1] so that its ciphertexts have the path-factorized structure required by the Diamond iO transformation. This construction relies on a new lattice assumption that we introduce, called *all-product LWE*. We also give a variant that avoids all-product LWE, at the cost of an additional functional-secret-key component whose size is proportional to the circuit size.

The all-product LWE assumption states that subexponentially many key-homomorphic encodings of Boneh et al. [16], sharing the same public matrix but using secrets derived by multiplying secret factors along input-dependent paths, are jointly pseudorandom when each encoding has an independently sampled

LWE error. The independent errors mask the exact algebraic relations among the underlying matrix products, preventing the low-rank structure exploited by known matrix-PRF attacks [33] from being directly exposed.

(iii) PFFE-decryption optimizations enabling end-to-end concrete-efficiency estimates with practical lattice parameters. Our third contribution is an optimization of the noise growth in the final PFFE-decryption stage, which is necessary for estimating the concrete end-to-end efficiency of Diamond iO with a realistic modulus q. This matters because our matrix-product FE-to-iO transformation uses the same modulus as the underlying PFFE scheme.

We implement the optimized PFFE scheme and use it as a common backend to compare the prior FE-to-iO transformation with Diamond iO. For several input sizes, we simulate the required lattice parameters and estimate the resulting concrete efficiency.

Remaining challenges. Diamond iO is still not practical iO, because PFFE decryption still requires an unrealistic amount of computation even when it is instantiated with practical lattice parameters. This bottleneck is not introduced by our matrix-product FE-to-iO transformation; it is already inherent in prior FE-to-iO transformations that rely on the same underlying FE-decryption procedure. Thus, making PFFE decryption practical remains an important open challenge.

Position of this work: the role of evasive LWE. The long-term goal is practical iO from solid, preferably standard or at least falsifiable, assumptions. Reaching this goal requires progress on two complementary fronts. The first is foundational: to refine the underlying security assumptions by replacing fragile or insufficiently understood assumptions with better-justified ones. The second is efficiency-oriented: to identify mathematical structures that enable iO constructions with better concrete efficiency, and to formulate the security conditions under which those structures can be used without compromising security. This work pursues the second front.

Concretely, this work shows that, by introducing a path-factorized structure into FE ciphertexts, the FE-to-iO transformation can be made dramatically more efficient. Informally, this efficiency gain comes with a specific security condition: one must justify that exponentially many LWE-like samples whose **secrets and errors are formed as products of path-factorized matrices** are no more useful to adversaries than LWE samples with unstructured secrets and independently sampled error terms.

Our present proof mainly uses private-coin evasive LWE, together with all-product LWE, as candidate assumptions meant to capture this desired security condition. We are aware that private-coin evasive LWE does not hold for arbitrary parameters and samplers as shown in [77, 30, 62, 44, 3, 59], and we do not regard it as a sufficiently mature foundation for iO.[6] Nevertheless, it remains the natural proof vehicle for our present construction: once the corresponding

[6] However, in Appendix D, we review the known attacks relevant to private-coin evasive LWE and explain why they either do not directly apply to Diamond iO under our sampler and parameter choices, or can be avoided by such choices.

precondition is established, this assumption allows us to derive the postcondition security needed to protect the path-factorized structure. Moreover, private-coin evasive LWE has enabled conceptually simple constructions of powerful primitives, including the compact FE scheme of Agrawal et al. [1]. This scheme was our key motivation for viewing constructions based on evasive LWE as a promising starting point in our search for implementable iO. Thus, our use of these assumptions should be read as a current, candidate formalization of the security conditions needed to protect the path-factorized structure.

Even if future cryptanalysis rules out private-coin evasive LWE for the concrete sampler used in our construction, or all-product LWE, the result would still have value as a concrete proposal for an efficiency-enabling mathematical structure in the FE-to-iO paradigm. Such a break would rule out the present instantiation as a secure iO candidate under these assumptions, but it would not erase the efficiency lesson of Diamond iO: products of path-factorized matrices can make the FE-to-iO transformation dramatically more efficient. In this way, this work contributes to the long-term goal of practical iO from solid assumptions by identifying an algebraic structure that future FE-to-iO constructions, under more robust assumptions, would need to protect in order to bring iO evaluation close to the concrete cost of the underlying FE scheme.

Paper organization. The remainder of this introduction gives a technical overview of Diamond iO and discusses related work. Section 2 recalls the cryptographic and lattice preliminaries used throughout the paper. Section 3 defines PFFE, constructs iPRO from PFFE, and bootstraps it to iO in the PROM. Section 4 constructs PFFE from lattices, followed by a cryptanalysis of all-product LWE. Section 5 presents the implemented optimizations, including the key-homomorphic PRF via specialized noise refreshing and minor implementation-level optimizations, and reports our concrete-efficiency estimates.

Appendix A gives the detailed asymptotic cost analysis. Appendices B, C, and E contain the missing proofs for Sections 2, 3, and 4, respectively. Appendix D analyzes known attacks on evasive LWE in our setting, Appendix F gives an alternative PFFE construction that avoids all-product LWE, and Appendix G gives the details of the minor optimizations used in our implementation.

1.2 Technical Overview

The FE-to-iO paradigm. Recall that public-key FE generalizes public-key encryption in the following way [21]. A trusted authority holding a master secret key, which corresponds to the obfuscator in the iO setting, releases a public key used for encryption and, for a function f, a functional secret key sk_f. Any party with access to this key, corresponding to an evaluator of obfuscated circuits in the iO setting, can decrypt a ciphertext $\mathsf{Enc}(\mathbf{x})$ of a plaintext $\mathbf{x}$ using sk_f. Correctness guarantees that the result is the output of the function associated with the key:

$$\mathsf{Dec}(\mathsf{sk}_f, \mathsf{Enc}(\mathbf{x})) = f(\mathbf{x}).$$

Informally, security guarantees that, even given sk_f and the ciphertext, the decryptor learns no non-trivial information about $\mathbf{x}$ beyond what can be inferred from f and $f(\mathbf{x})$.

Putting this FE syntax into the iO setting, fix a circuit C to be obfuscated and an evaluator's input $\mathbf{x}$. The obfuscator samples a symmetric key K and computes a ciphertext $\widehat{C} := \mathsf{SKE.Enc}(K, C)$ of the circuit. It then defines a public keyed circuit $F_{\widehat{C}}$ that on input $(K, \mathbf{x})$ decrypts the hardcoded ciphertext $\widehat{C}$ and evaluates the recovered circuit on $\mathbf{x}$ using a universal circuit U:

$$F_{\widehat{C}}(K, \mathbf{x}) := U(\mathsf{SKE.Dec}(K, \widehat{C}), \mathbf{x}) = C(\mathbf{x}).$$

The obfuscator provides a functional secret key $\mathsf{sk}_{F_{\widehat{C}}}$ for this keyed circuit, while the decryption key K for $\widehat{C}$ is kept hidden inside an FE ciphertext. Thus, the evaluator's task is to obtain an FE ciphertext of $(K, \mathbf{x})$, without learning K, and then decrypt it using $\mathsf{sk}_{F_{\widehat{C}}}$.

This gives the following two-phase view of evaluating an obfuscated circuit in the FE-to-iO paradigm [8, 15, 1].

Phase 1: Input insertion into the FE ciphertext. Starting from an FE ciphertext $\mathsf{FE.Enc}((K, \epsilon))$ that contains the hidden symmetric key K with the input slot empty, the evaluator inserts its own input $\mathbf{x}$ and obtains a ciphertext of $(K, \mathbf{x})$, which we denote by $\mathsf{FE.Enc}((K, \mathbf{x}))$.

Phase 2: FE decryption. The evaluator uses the obfuscator-provided functional secret key $\mathsf{sk}_{F_{\widehat{C}}}$ to decrypt the resulting ciphertext:

$$\mathsf{FE.Dec}(\mathsf{sk}_{F_{\widehat{C}}}, \mathsf{FE.Enc}((K, \mathbf{x}))) = F_{\widehat{C}}(K, \mathbf{x}) = C(\mathbf{x}).$$

Thus, the evaluator obtains the circuit output $C(\mathbf{x})$ in the clear.

The Phase 1 bottleneck of prior FE-to-iO transformations. The prior FE-to-iO transformations [8, 15, 1] realize Phase 1 by recursively invoking the FE encryption algorithm for each input bit inside the function evaluated during decryption. Informally, for each index $i \in \{1, \ldots, L-1\}$, the obfuscator provides a functional secret key for a function that on input $(K, x_1, \ldots, x_{i-1})$, outputs FE encryptions of $(K, x_1, \ldots, x_{i-1}, 0)$ and $(K, x_1, \ldots, x_{i-1}, 1)$. The evaluator can obtain the encryption of $(K, x_1, \ldots, x_i)$ by using the supplied functional secret key for index i to decrypt the encryption of $(K, x_1, \ldots, x_{i-1})$, and then chooses the first half or the second half of the resulting ciphertext according to the input bit x_i.

Therefore, when the circuit being evaluated in Phase 2 is simpler than the FE encryption circuit, whose complexity is at least that of the PRF computation, the overhead introduced by Phase 1 becomes larger than the cost of the final FE decryption in Phase 2. More concretely, suppose that the cost of FE decryption is proportional to both the input size and the output size of the function being evaluated. Then the prior FE-to-iO transformations incur cost on the order of L^3 for inserting an L-bit input. This is the bottleneck that motivates our focus on improving the efficiency of Phase 1.

Roadmap of the overview. The rest of this overview focuses on the main new idea of Diamond iO: realizing the input-insertion phase by matrix products. We first introduce the structural property of FE ciphertexts that makes this possible. We abstract this property as PFFE, and explain how GGH15 encodings let the evaluator obtain the FE ciphertext for its input without learning the hidden key K. We then briefly explain how our PFFE construction is obtained from the lattice-based FE scheme of Agrawal et al. [1], and give the security intuition for the resulting Diamond iO construction.

The role of our Phase 2 optimization is mainly to keep FE decryption implementable with a realistic modulus q, which is necessary because Phase 1 uses the same modulus. Concretely, this optimization refreshes the encoding-side noise that arises when homomorphically evaluating the PRF over key-homomorphic encodings; the usual FHE ciphertext noise remains the input-length-dependent noise source. This is important because, under our parameter setting, the encoding-side noise introduced by a single FHE multiplication is roughly eleven times larger than the corresponding noise from the FHE ciphertext itself. We do not discuss this optimization further in the overview; the rest of this section focuses on the idea of matrix-product input-insertion for Phase 1.

Realizing Phase 1: PFFE and matrix-product input insertion. At a high level, Diamond iO brings the path-product intuition behind GGH15-encoding-based iO [48] to Phase 1 of the FE-to-iO paradigm, but uses it only for input insertion rather than for directly recovering the circuit output. We require the relevant FE ciphertext component to have the following path-factorized form:

$$\mathsf{FE.Enc}((K, \mathbf{x})) = \underbrace{\hat{\mathbf{s}}_K \hat{\mathbf{S}}_{1,x_1} \hat{\mathbf{S}}_{2,x_2} \cdots \hat{\mathbf{S}}_{L,x_L}}. \tag{1}$$

Here, $\hat{\mathbf{s}}_K$ depends on the hidden symmetric key K but not on the evaluator's input $\mathbf{x}$, and each factor $\hat{\mathbf{S}}_{i,x_i}$ is selected according only to the input bit x_i. The wavy underline means that the displayed noiseless product should be read as being perturbed by a small noise term, which we omit from the notation.

The obfuscator publishes only GGH15 encodings of these secret factors, keeping the factors themselves hidden. Concretely, the initial vector $\hat{\mathbf{s}}_K$ is encoded as an LWE sample, and each matrix factor $\hat{\mathbf{S}}_{i,b}$ for $b \in \{0,1\}$ is encoded as a short lattice preimage [49, 70, 47]. Given $\mathbf{x} \in \{0,1\}^L$, the evaluator selects the encodings corresponding to $x_1, \ldots, x_L$ and multiplies them. The preimage property ensures that this product evaluates to the path product in Eq. (1) up to small noise. Thus, the evaluator obtains the FE ciphertext needed for Phase 2 using only matrix products over published encodings, while learning neither K nor the secret path factors.

Abstraction as PFFE. We abstract the ciphertext structure in Eq. (1) as PFFE. A PFFE scheme is a partially hiding FE scheme whose ciphertexts admit such a path-factorized representation: the private input $\mathbf{z}$ determines an input-independent initial secret vector $\hat{\mathbf{s}}_{\mathbf{z}}$, while each public input bit selects the matrix factors $\hat{\mathbf{S}}_{i,b}$. In the Diamond iO application, this private input $\mathbf{z}$ contains the hidden symmetric key K. Once an FE scheme satisfies the PFFE properties,

Diamond iO realizes Phase 1 by publishing GGH15 encodings of the path factors and letting the evaluator multiply the selected encodings.

More formally, the function class for which PFFE is defined consists of functions $f(\mathbf{z}, \mathbf{x})$ that take a private input $\mathbf{z} \in \{0,1\}^\ell$ and a public input $\mathbf{x} \in \{0,1\}^L$ and returns an ouput in $\{0,1\}^M$. The difference of the syntax from standard FE is as follows:

- The encryption algorithm Enc takes both the private input $\mathbf{z}$ and the public input $\mathbf{x}$ as input, and its internal logic is decomposed into two algorithms: one is an $\mathsf{EncPriv}$ algorithm that takes only the private input $\mathbf{z}$ and outputs the $\mathbf{x}$-independent part of the ciphertext $\mathsf{ct}_{\mathbf{z}}$ and an auxiliary state st, and the other is an EncPub algorithm that takes the public input $\mathbf{x}$ and the auxiliary state st and outputs the rest of the ciphertext $\mathsf{ct}_{\mathbf{x}}$.
- An additional extraction algorithm $\mathsf{Extract}$ is defined. It takes as input the auxiliary state st and outputs the secret path factors $\left(\hat{\mathbf{s}}_{\mathbf{z}}, \{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],b\in\{0,1\}}\right)$.

Its security says, informally, that if the joint distribution of function outputs $\{f(\mathbf{z},\mathbf{x})\}_{\mathbf{x}\in\{0,1\}^L}$ over all public inputs $\mathbf{x}$ is pseudorandom even given the auxiliary information $\mathsf{aux}_{\mathbf{z}}$ sampled together with $\mathbf{z}$, then the corresponding ciphertexts $\left(\mathsf{ct}_{\mathbf{z}}, \{\mathsf{ct}_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}\right)$ are also pseudorandom even given the functional secret key. In addition, the postcondition adversary is given extra LWE samples whose secrets are intermediate products of the secret factors, namely

$$\left\{ \underbrace{\left(\hat{\mathbf{s}}_{\mathbf{z}} \hat{\mathbf{S}}_{1,x_1} \cdots \hat{\mathbf{S}}_{i,x_i}\right) \mathbf{B}_i}_{} \right\}_{i\in\{1,\ldots,L\},\,(x_1,\ldots,x_i)\in\{0,1\}^i},$$

where $\mathbf{B}_i$ is an uniformly random matrix sampled independently for each i, and each sample is perturbed by an independent error term. The postcondition of the PFFE security also guarantees that these LWE samples are pseudorandom.

Construction of PFFE based on the lattice-based FE of Agrawal et al. We construct PFFE by modifying the lattice-based FE scheme of Agrawal et al. [1]. In this scheme, the relevant ciphertext component is a key-homomorphic encoding introduced by Boneh et al. [16], commonly called a BGG+ encoding:

$$\mathbf{c}_{\mathbf{x}} = \underbrace{\mathbf{s}_{\mathbf{x}} \left(\mathbf{A} - \mathbf{m}_{\mathbf{z},\mathbf{x}} \otimes \mathbf{G}\right)}. \tag{2}$$

Here, a low-norm vector $\mathbf{m}_{\mathbf{z},\mathbf{x}}$ contains the bit string $\hat{\mathbf{z}}$ of an FHE ciphertext encrypting the private input $\mathbf{z}$ and the low-norm FHE decryption key $\mathbf{t}$, together with the public input $\mathbf{x}$. The goal is to make these ciphertexts admit the path-factorized form required by PFFE as shown in Eq. (1).

The main difficulty is the choice of the secret $\mathbf{s}_{\mathbf{x}}$. If $\mathbf{s}_{\mathbf{x}}$ is sampled independently for every public input, as in the original FE scheme, the ciphertexts do not have an obvious path structure. If the same secret is reused for all public inputs, path factorization is straightforward, but the scheme is insecure: subtracting ciphertexts for two inputs that differ in one bit cancels the common

random part and exposes a structured secret term $\underset{\sim\sim}{\mathbf{s}_{\mathbf{x}}\mathbf{G}}$. Since $\mathbf{G}$ is a fixed public gadget matrix, this term can be decoded efficiently, allowing the adversary to recover the secret.

Our solution is to make the secret itself path-factorized:

$$\mathbf{s}_{\mathbf{x}} := \mathbf{s}_{\epsilon}\mathbf{S}_{1,x_1} \cdots \mathbf{S}_{L,x_L}. \tag{3}$$

This gives different secrets for different public inputs while preserving the matrix-product structure needed for our matrix-product input insertion.

Specifically, we define the secret factors as follows.

$$\begin{aligned}
&\hat{\mathbf{s}}_{\mathbf{z}} := \mathbf{m}_{\mathbf{z}} \otimes \mathbf{s}_{\epsilon},\\
&\forall i \in \{1,\ldots,L-1\}, b \in \{0,1\}, \hat{\mathbf{S}}_{i,b} := \mathbf{M}_{i,b} \otimes \mathbf{S}_{i,b},\\
&\forall b \in \{0,1\}, \hat{\mathbf{S}}_{L,b} := (\mathbf{M}_{L,b} \otimes \mathbf{S}_{L,b})\mathbf{B}_L,
\end{aligned}$$

where $\mathbf{s}_{\epsilon}$ and $\mathbf{S}_{i,b}$ are sampled independently over ternary matrices for every $i \in \{1,\ldots,L\}$ and $b \in \{0,1\}$, $\mathbf{B}_L$ is a uniformly random matrix contained in a public key, and $\mathbf{M}_{i,b}$ is a public binary matrix that depends only on i and b. For every $\mathbf{x} \in \{0,1\}^L$, these binary matrices satisfy

$$\mathbf{m}_{\mathbf{z}}\mathbf{M}_{1,x_1} \cdots \mathbf{M}_{L,x_L} = \mathbf{m}_{\mathbf{z},\mathbf{x}}.$$

Then the ciphertext $\mathsf{ct}_{\mathbf{x}}$ corresponding to $\mathbf{z}$ and $\mathbf{x}$ is defined as follows:

$$\begin{aligned}
\mathsf{ct}_{\mathbf{x}} &\approx \hat{\mathbf{s}}_{\mathbf{z}}\hat{\mathbf{S}}_{1,x_1} \cdots \hat{\mathbf{S}}_{L,x_L}\\
&= \left(\left(\mathbf{m}_{\mathbf{z}} \otimes \mathbf{s}_{\epsilon}\right)\prod_{i=1}^{L}\left(\mathbf{M}_{i,x_i} \otimes \mathbf{S}_{i,x_i}\right)\right)\mathbf{B}_L\\
&= \left(\left(\mathbf{m}_{\mathbf{z}}\prod_{i=1}^{L}\mathbf{M}_{i,x_i}\right) \otimes \left(\mathbf{s}_{\epsilon}\prod_{i=1}^{L}\mathbf{S}_{i,x_i}\right)\right)\mathbf{B}_L\\
&= \left(\mathbf{m}_{\mathbf{z},\mathbf{x}} \otimes \mathbf{s}_{\mathbf{x}}\right)\mathbf{B}_L.
\end{aligned}$$

The obfuscator includes a preimage $\mathbf{K}_{\mathsf{att}}$ that derives the BGG+ encoding in Eq. (2), sampled with a trapdoor $\mathbf{B}_L^{-1}$, in the functional secret key, so that the evaluator can compute

$$\mathsf{ct}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}} \approx \mathbf{s}_{\mathbf{x}}\left(\mathbf{A} - \mathbf{m}_{\mathbf{z},\mathbf{x}} \otimes \mathbf{G}\right).$$

Security intuition for Diamond iO. The obfuscated circuit produced by Diamond iO consists of three parts: the published GGH15 encodings used for input insertion, the functional secret key of the underlying PFFE scheme for $F_{\widehat{C}}$, and the part of the FE ciphertext that depends on the hidden key K but not on the evaluator's input $\mathbf{x}$; in our concrete PFFE construction, this input-independent part includes an FHE ciphertext of K. The security argument of Diamond iO is based on two complementary claims.

First, the published GGH15 encodings—both the LWE samples and the preimages—should be computationally indistinguishable from random matrices. Although the evaluator receives preimages encoding the secret factors, the target matrices opened by these preimages remain hidden. We justify this step using private-coin evasive LWE, in a manner similar to [77]. Roughly, assuming that the joint distribution of all possible path products is pseudorandom over all public inputs $\mathbf{x} \in \{0,1\}^L$ when the individual secret factors are hidden, the assumption lets us replace the published GGH15 encodings by random matrices in the security hybrids.

Second, conditioned on the individual secret factors remaining hidden, the joint distribution of the path products for all public inputs should be pseudorandom. In the modular construction of Diamond iO, where the PFFE scheme is used as a black box, we guarantee this by invoking the PFFE security property, which says that the ciphertexts are pseudorandom jointly over all public inputs $\mathbf{x} \in \{0,1\}^L$, even given the functional secret key and the auxiliary information of the secret factors. For our concrete PFFE construction, we prove this property under our new lattice assumption, which we call *all-product LWE*.

Combining these two claims, we show that the resulting construction satisfies the iPRO security definition, which requires that the obfuscations of two functionally equivalent pseudorandom circuits are indistinguishable. By the lifting theorem of [29], this yields iO for general circuits in the PROM.

Cryptanalysis of our new assumption: all-product LWE. At a high level, the all-product LWE assumption says the following. Suppose that the secret $\mathbf{s}_\mathbf{x}$ is defined in the path-factorized form of Eq. (3). Then the joint distribution of BGG+ encodings

$$\mathbf{c}_\mathbf{x} := \mathbf{s}_\mathbf{x} \left(\mathbf{A} - \mathbf{x} \otimes \mathbf{G}\right) + \mathbf{e}_\mathbf{x}$$

for all $\mathbf{x} \in \{0,1\}^L$, together with the shared public matrix $\mathbf{A}$, is pseudorandom. Notably, the noise term $\mathbf{e}_\mathbf{x}$ is sampled independently for each $\mathbf{x} \in \{0,1\}^L$. Such a fresh noise is acceptable in our application, because the assumption is invoked only in the precondition of the evasive LWE.

This assumption avoids a known attack on a matrix PRF [33]. A matrix PRF also outputs a product of secret matrices selected by the input bits and claims that the resulting products are pseudorandom. However, known rank attacks exploit the fact that such products can expose low-rank algebraic structure when the input is split into two parts. In contrast, all-product LWE is not a noiseless matrix-product distribution because every path product is masked by the independent noise term $\mathbf{e}_\mathbf{x}$. These fresh noises hide the exact algebraic relations among different path products, so the low-rank structure used in known matrix-PRF attacks is not directly exposed.

One might object that this does not match the actual Diamond iO construction, or equivalently the postcondition of evasive LWE: there, the noises are not sampled independently, but each output noise is obtained from the noise attached to the LWE sample encoding the initial secret $\hat{\mathbf{s}}_K = \hat{\mathbf{s}}_\mathbf{z}$, multiplied by the preimages each of which is selected according only to a single input bit x_i. Thus,

one might expect the attacks against matrix PRFs to remain applicable. This observation is correct in the sense that the actual BGG+ encodings obtained after multiplying all preimages are no longer pseudorandom.

However, the pseudorandomness of these final BGG+ encodings is not what we ultimately need from the postcondition. Rather, we need the GGH15 encodings to keep the secret factors hidden, and thereby hide both the FHE decryption key $\mathbf{t}$ contained in $\mathbf{m}_{\mathbf{z}}$ and the private input $\mathbf{z}$ encrypted under FHE. Therefore, the fact that the BGG+ encodings in the postcondition are not pseudorandom does not by itself imply a break of Diamond iO. The same logic is used in the security proof of the witness encryption scheme in [77]: if every product of vectors and preimages from GGH encodings included in the ciphertext, perturbed by independently sampled noises, is pseudorandom, then those GGH encodings hide the encoded secret matrices under the evasive LWE assumption.

1.3 Related Works

Lattice-based iO. Existing iO schemes from standard assumptions [65, 66, 73] rely on bilinear maps and are therefore not post-quantum secure. A parallel line of work has pursued plausibly post-quantum iO from lattices [26, 43, 46, 80, 25, 2, 29]. These schemes are based on LWE together with new lattice assumptions, often interpretable as "LWE-with-hints [59]" assumptions. However, subsequent works [57, 63, 59, 3] showed that some of these new assumptions are fragile by giving polynomial-time attacks against them.

More recently, Agrawal et al. [2] and Branco et al. [29] initiated the study of pseudorandom obfuscation, namely iO for pseudorandom functionalities, from LWE and private-coin evasive LWE [79]. Our work follows this line: we construct PFFE from LWE, private-coin evasive LWE, and our new lattice assumption, all-product LWE, and use it to obtain pseudorandom obfuscation.[7] Although recent attacks against private-coin evasive LWE have been proposed [59, 3], our construction remains secure against all known attacks, as discussed in Appendix D.

To the best of our knowledge, all existing lattice-based iO and pseudorandom-obfuscation constructions still rely on the conventional FE-to-iO transformations [8, 15, 2]. As a result, they inherit the overhead caused by recursive FE encryption in the input-insertion phase. Our construction removes this overhead by replacing recursive FE encryption with a matrix-product input-insertion procedure. Thus, our work gives the first lattice-based pseudorandom obfuscation construction that avoids the recursive-encryption overhead of prior FE-to-iO transformations.

iO candidates based on multilinear maps. We also compare Diamond iO with iO candidates based on multilinear maps, especially those based on GGH15 encodings [48], which we refer to as GGH15-based iO candidates. This

[7] More precisely, we construct only the weakest notion of pseudorandom obfuscation, namely indistinguishable pseudorandom obfuscation (iPRO), since stronger notions are known to be impossible for general circuits [29].

comparison is natural because Diamond iO also uses GGH15 encodings in its FE-to-iO transformation, although only for the input-insertion phase.

The first candidate construction of iO by Garg et al. [45] initiated a long line of work based on multilinear maps [40, 72, 48]. This line produced several conceptually simple iO candidates, but concrete multilinear-map instantiations have had a fragile security history, with many proposed constructions and iO variants later broken by algebraic and zeroizing attacks [37, 38, 36, 39, 32, 34, 35, 33].

Several works have also implemented or estimated the concrete cost of multilinear-map-based iO candidates [9, 67, 55, 18]. However, these works either target restricted functionalities or analyze simplified variants; they do not demonstrate a full implementation of a general-purpose iO candidate with all safeguards intact.

Efficiency comparison with GGH15-based iO candidates. From the perspective of efficiency, Diamond iO and GGH15-based iO candidates use GGH15 encodings in qualitatively different ways. In GGH15-based iO candidates, GGH15 encodings serve as the main evaluation medium: the obfuscated evaluation itself is carried out by composing encoded elements. In Diamond iO, by contrast, GGH15 encodings are used only for the FE-to-iO transformation in Phase 1.

The resulting efficiency difference can be explained more concretely by the length of the preimage-multiplication chain. Let ℓ denote the number of successive preimage multiplications performed over GGH15 encodings. This parameter is crucial because the modulus q required for correctness grows roughly exponentially with ℓ. In Diamond iO, for a circuit with input length L, the evaluator multiplies exactly one encoded preimage per input bit; therefore $\ell = L$.

In GGH15-based iO candidates, by contrast, the GGH15 encodings are used to evaluate the obfuscated program itself. For a circuit of depth d, Barrington's theorem [12] converts the circuit into a constant-width branching program of length at most 4^d, implying $\ell \leq 4^d$. For $\mathbf{NC}^1$ circuits, this bound is polynomial in the input length L, and it becomes $O(L)$ under depth normalizations where $4^d = O(L)$. Even in such favorable linear cases, however, the concrete value of ℓ can be significantly larger than L.

This distinction is particularly visible for PRF computations. Chen et al. [33] showed that if a matrix PRF is computable by a read-c, width-w branching program, then it can be attacked in time $\mathsf{poly}(w^c)$. Consequently, when the width is constant, achieving λ-bit security requires reading each input bit $\omega(\log \lambda)$ times. For a PRF on L input bits, this means that the length ℓ of a constant-width GGH15-based obfuscation must grow as $\ell = L \cdot \omega(\log \lambda)$. For example, at the 128-bit security level, this length is at least seven times longer than that in Diamond iO for the same input size L.

This comparison at the GGH15-encoding level, however, should not be interpreted as an end-to-end efficiency advantage. The current end-to-end cost of Diamond iO is dominated by Phase 2, namely the final PFFE decryption. This step requires homomorphic evaluation of relatively complex circuits over key-homomorphic BGG+ encodings. Consequently, although Diamond iO reduces the cost of the input-insertion component, its current end-to-end cost is likely

larger than existing estimates for simplified GGH15-based iO candidates, such as those of [55].

We view this Phase 2 bottleneck as orthogonal to our Phase 1 improvement and as an important target for further optimization. Our implementation already incorporates two recent advances for richer computation over BGG+ encodings: we adapt the noise-refreshing technique introduced in [61] in a form different from their original construction, and we use the lookup-table evaluation technique introduced in [76] as an optimization of our PFFE scheme as described in Appendix G.2. These optimizations make the PFFE scheme implementable with realistic lattice parameters; further improvements for BGG+ encodings may reduce the remaining concrete cost of the final PFFE-decryption phase without changing our matrix-product input-insertion method.

2 Preliminaries

2.1 Notations

For non-negative integers $i, j \in \mathbb{Z}$ such that $i \leq j$ holds, $[i, j]$ denotes the set $\{i, i+1, \ldots, j\}$. For simplicity, $[n]$ and $[0]$ represent the sets $\{1, \ldots, n\}$ and $\emptyset$, respectively.

A vector and a matrix are denoted by bold letters, such as $\mathbf{a}$ and $\mathbf{A}$. Unless otherwise specified, we define vectors as row vectors. For any vector $\mathbf{a}$, the size of $\mathbf{a}$ is denoted by $|\mathbf{a}|$, and the i-th element of $\mathbf{a}$ is denoted by a_i for every $i \in [|\mathbf{a}|]$. For any matrix $\mathbf{A}$ with n rows and m columns, let $a_{i,j}$ denote the (i, j)-th entry of $\mathbf{A}$, for every $i \in [n]$ and $j \in [m]$. The ℓ_2-norm of the n-sized vector $\mathbf{a}$ is defined by $\|a\|_2 := \sqrt{\Sigma_{i \in [|\mathbf{a}|]} a_i^2}$. For a matrix $\mathbf{A}$ with n rows and m columns, $\|A\|_\infty$ denotes the entry-wise infinity norm, defined by $\|A\|_\infty := \max_{i \in [n], j \in [m]} \{|a_{i,j}|\}$. For any multiplication between two matrices $\mathbf{A}$ and $\mathbf{B}$, where the column size of $\mathbf{A}$ and the row size of $\mathbf{B}$ are m, it holds that $\|AB\|_\infty \leq m \|A\|_\infty \|B\|_\infty$. For any $n \in \mathbb{N}$ and $i \in [n]$, $\mathbf{u}_i \in \mathbb{Z}_q^n$ is a unit vector whose i-th entry is 1 and the others are 0.

Let $\mathbb{Z}_q$ denote the ring of integers modulo q. For an integer $x \in \mathbb{Z}_q$, define $\mathsf{bits}(x) := (x_1, \ldots, x_{\lceil \log_2 q \rceil}) \in \{0,1\}^{1 \times \lceil \log_2 q \rceil}$ as the vector of bits corresponding to the binary representation of x in little-endian form—i.e., $x = \Sigma_{i \in [\lceil \log_2 q \rceil]} x_i 2^{i-1}$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathsf{bits}(\mathbf{A}) \in \{0,1\}^{nm\lceil \log_2 q \rceil}$ is similarly defined as $\mathsf{bits}(\mathbf{A}) := (\mathsf{bits}(a_{1,1}), \ldots, \mathsf{bits}(a_{n,1}), \ldots, \mathsf{bits}(a_{1,m}), \ldots, \mathsf{bits}(a_{n,m}))$.

For any distribution $\mathcal{X}$, $x \leftarrow\!\!\$\ \mathcal{X}$ indicates that x is sampled from $\mathcal{X}$. Let $x \leftarrow\!\!\$\ \mathbb{Z}_q$ denote that x is sampled from a uniform distribution over $\mathbb{Z}_q$.

Let λ be a security parameter. A function $\mathsf{negl}(\lambda) : \mathbb{N} \to \mathbb{R}$ is called negligible if for all constants $c > 0$, there exists $n \in \mathbb{N}$ such that $\mathsf{negl}(\lambda) < \lambda^{-c}$ holds for all $\lambda > n$. For any two distributions $\mathcal{X}$ and $\mathcal{Y}$, we write $\mathcal{X} \overset{\mathrm{c}}{\approx} \mathcal{Y}$ (resp., $\mathcal{X} \overset{\mathrm{s}}{\approx} \mathcal{Y}$) to denote that $\mathcal{X}$ and $\mathcal{Y}$ are computationally (resp., statistically) indistinguishable for all probabilistic polynomial-time (PPT) adversaries.

2.2 Lattice and Gaussian Preliminaries

We recall some facts on lattices and Gaussian distributions.

Lattices. Fix some integers $n, m, q \in \mathbb{N}$. For a full-rank integer matrix $\mathbf{B} \in \mathbb{Z}^{n \times m}$, an integer lattice is defined by $\Lambda(\mathbf{B}) := \{\mathbf{B}\mathbf{x}^T \mid \mathbf{x} \in \mathbb{Z}^m\}$. For any matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, a q-ary orthogonal lattice is defined by $\Lambda^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x}^T = \mathbf{0}^T \mod q\}$. For any vector $\mathbf{u} \in \mathbb{Z}_q^n$, its coset is defined by $\Lambda_\mathbf{u}^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x}^T = \mathbf{u}^T \mod q\}$.

We adopt the leftover hash lemma from [1].

Lemma 1 (Leftover Hash Lemma). Fix some $n, m, q \in \mathbb{N}$. The leftover hash lemma states that if $m \geq 2n \log q$, then for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{x} \leftarrow \{0,1\}^m$ and $\mathbf{y} \leftarrow \mathbb{Z}_q^n$, the statistical distance between $(\mathbf{A}, \mathbf{A}\mathbf{x}^T)$ and $(\mathbf{A}, \mathbf{y}^T)$ is negligible. More concretely, it is bounded by $q^n\sqrt{2^{1-m}}$.

Gadget vector and matrix. Let $m \geq n\lceil \log_2 q \rceil$. A gadget vector $\mathbf{g}$ and a gadget matrix $\mathbf{G}$ are defined as $\mathbf{g} := (2^0, 2^1, \ldots, 2^{\lceil \log_2 q \rceil - 1}) \in \mathbb{Z}_q^{\lceil \log_2 q \rceil}$ and $\mathbf{G} := \mathbf{I}_n \otimes \mathbf{g} \in \mathbb{Z}_q^{n \times m}$, respectively. For any vector $\mathbf{a} \in \mathbb{Z}_q^n$, $\mathbf{G}^{-1}(\mathbf{a}) \in \Lambda_\mathbf{a}^\perp(\mathbf{G})$ is defined as $\mathbf{G}^{-1}(\mathbf{a}) := (\mathsf{bits}(a_1), \ldots, \mathsf{bits}(a_n)) \in \mathbb{Z}_q^m$. This is straightforwardly extended to a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times t}$ as $\mathbf{G}^{-1}(\mathbf{A}) := (\mathbf{G}^{-1}(\mathbf{a}_1)^T, \ldots, \mathbf{G}^{-1}(\mathbf{a}_t)^T) \in \mathbb{Z}_q^{m \times t}$, which satisfies $\mathbf{G}\mathbf{G}^{-1}(\mathbf{A}) = \mathbf{A}$.

Gaussian distributions. For any lattice $\Lambda \subseteq \mathbb{R}^b$ and any Gaussian parameter $\sigma > 0$, let $\mathcal{D}_{\Lambda,\sigma}$ be a discrete Gaussian distribution. The following lemma bounds the tail probability of the discrete Gaussian distribution [70]. In the same manner as in [60], we truncate $\mathcal{D}_{\mathbb{Z},\sigma}$ to simplify a noise analysis.

Lemma 2 (Tail and Truncation of the Discrete Gaussian Distribution [70, 60]). For any Gaussian parameter $\sigma > 1$, $\lambda \in \mathbb{N}$, and $e \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma}$, there exists $B_0 \in \Theta(\sqrt{\lambda})$ such that

$$\Pr[e \geq \sigma B_0(\lambda)] \leq 2^{-\lambda}.$$

For $B \geq 0$, the distribution $\mathcal{D}_{\mathbb{Z},\sigma \leq B}$ is sampled by first sampling $x \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma}$, then returning x if $|x| \leq B$ and 0 otherwise. If $\sigma \geq 1$ and $B = \sigma \cdot \Theta(\sqrt{\lambda})$, then $\mathcal{D}_{\mathbb{Z},\sigma \leq B}$ is $2^{-\Omega(\lambda)}$-close to $\mathcal{D}_{\mathbb{Z},\sigma}$.

In the sequel, we simply write $\mathcal{D}_{\mathbb{Z},\sigma}$ for the truncated distribution $\mathcal{D}_{\mathbb{Z},\sigma \leq B}$, where $B = \sigma \cdot \Theta(\sqrt{\lambda})$. Thus, whenever we write $e \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma}$, it is understood that $|e| \leq \sigma \cdot \Theta(\sqrt{\lambda})$ holds deterministically.

We adopt the smudging lemma from [78, 1].

Lemma 3 (Smudging Lemma [78, 1]). Let λ be a security parameter. Take any $a \in \mathbb{Z}$ where $|a| \leq B$. Suppose $\sigma \geq B\lambda^{\omega(1)}$. Then the statistical distance between the distributions $\{e \mid e \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma}\}$ and $\{e + a \mid e \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma}\}$ is $\mathsf{negl}(\lambda)$.

Trapdoor and preimage sampling. We adopt the definitions and syntax of trapdoor and preimage sampling algorithms from [1]. For integers $n, q \in \mathbb{N}$, let

$m := \mathcal{O}(n \log q)$ and $\gamma := \omega(\sqrt{n \log q \log m})$. Previous works [49, 70, 47] propose an algorithm $\mathsf{TrapGen}(1^n, 1^m, q)$ that outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ which is 2^{-n} close to uniform, and a γ-trapdoor for $\mathbf{A}$ denoted by $\mathbf{A}_\gamma^{-1}$.

The trapdoor allows efficient sampling of a preimage for a target vector $\mathbf{p} \in \mathbb{Z}_q^n$ from the discrete Gaussian distribution over $\Lambda_{\mathbf{p}}^{\perp}(\mathbf{A})$, namely $\mathcal{D}_{\Lambda_{\mathbf{p}}^{\perp}(\mathbf{A}),\gamma}$. To sample a preimage for a matrix $\mathbf{P} \in \mathbb{Z}_q^{n \times l}$, we sample l preimages, one for each column of $\mathbf{P}$—that is, $\mathbf{A}_\gamma^{-1}(\mathbf{P}) := (\mathbf{A}_\gamma^{-1}(\mathbf{p}_1), \ldots, \mathbf{A}_\gamma^{-1}(\mathbf{p}_l))$, where $\mathbf{p}_i$ denotes the i-th column of $\mathbf{P}$.

Hardness assumptions. We rely on the following hardness assumptions on lattices. The subexponential hardness of the learning with errors (LWE) assumption defined in Assumption 1 is adopted from Assumption A.5 of [2] with necessary modifications.

Assumption 1 ((Subexponential) Hardness of Learning With Errors (LWE) [74]). Let $n = n(\lambda), m = m(\lambda), q = q(\lambda) > 2$ be integers, and $\chi_s = \chi_s(\lambda)$ and $\chi_e = \chi_e(\lambda)$ be distributions over $\mathbb{Z}_q$. The LWE assumption is said to hold if for every PPT adversary $\mathcal{A}$, it holds that

$$|\Pr[\mathcal{A}(\mathbf{A}, \mathbf{sA} + \mathbf{e}) = 1] \ - \ \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u}) = 1]| \ \leq \ \mathrm{negl}(\lambda),$$

where $\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow\!\$\ \chi_s^n$, $\mathbf{e} \leftarrow\!\$\ \chi_e^m$, $\mathbf{u} \leftarrow\!\$\ \mathbb{Z}_q^m$, and the probability is taken over the randomness of $\mathcal{A}$.

Additionally, the LWE assumption is said to be subexponentially hard if there exists some constant $\delta \in (0, 1)$ such that for every non-uniform adversary $\mathcal{A}$ of size at most 2^{n^δ}, it holds that

$$|\Pr[\mathcal{A}(\mathbf{A}, \mathbf{sA} + \mathbf{e}) = 1] \ - \ \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u}) = 1]| \ \leq \ 2^{-n^\delta},$$

where $\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow\!\$\ \chi_s^n$, $\mathbf{e} \leftarrow\!\$\ \chi_e^m$, $\mathbf{u} \leftarrow\!\$\ \mathbb{Z}_q^m$, and the probability is taken over the randomness of $\mathcal{A}$.

In what follows, we instantiate the secret distribution χ_s with the uniform distribution over $\{0, \pm 1\}^n$, and the error distribution χ_e with a discrete Gaussian distribution over $\mathbb{Z}_q$ with a Gaussian parameter σ.

We adopt a variant of the evasive LWE assumption defined in [2], called "non-uniform κ-evasive LWE assumption [2]". However, unlike their definition, our definition assumes that an adversary can access a public matrix $\mathbf{P}$. This is because Brzuska et al. [30] show that the adversary who can see only another public matrix $\mathbf{B}$ can break the security of the evasive LWE with a non-negligible advantage in the private-coin setting. To fix that vulnerability, they introduced a definition, called "private-coin binding evasive LWE [30]", which includes $\mathbf{P}$ in the adversary's view.

Assumption 2 (Private-coin Binding Non-Uniform κ-Evasive LWE [2, 30]). Let $n, m, t, m', q \in \mathbb{N}$ be parameters and λ be a security parameter. Let χ and χ' be Gaussian parameters. Let Samp be a PPT algorithm that outputs

$$\mathbf{S} \in \mathbb{Z}_q^{m' \times n}, \mathbf{P} \in \mathbb{Z}_q^{n \times t}, \mathsf{aux} \in \{0, 1\}^*$$

on input 1^λ. For non-uniform adversaries $\mathcal{A}_0 = \{\mathcal{A}_{0,\lambda}\}_\lambda$, $\mathcal{A}_1 = \{\mathcal{A}_{1,\lambda}\}_\lambda$, we define the following advantage functions:

$$\begin{aligned}\mathrm{Adv}^{\mathrm{PRE}}_{\mathcal{A}_0}(\lambda) &:= \Pr\big[\mathcal{A}_0\big(\mathbf{B},\, \mathbf{P},\, \mathbf{SB}+\mathbf{E},\, \mathbf{SP}+\mathbf{E}',\, \mathsf{aux}\big)=1\big] \\ &\quad -\ \Pr\big[\mathcal{A}_0\big(\mathbf{B},\, \mathbf{P},\, \mathbf{C}_0,\, \mathbf{C}',\, \mathsf{aux}\big)=1\big], \\ \mathrm{Adv}^{\mathrm{POST}}_{\mathcal{A}_1}(\lambda) &:= \Pr\big[\mathcal{A}_1\big(\mathbf{B},\, \mathbf{P},\, \mathbf{SB}+\mathbf{E},\, \mathbf{K},\, \mathsf{aux}\big)=1\big] \\ &\quad -\ \Pr\big[\mathcal{A}_1\big(\mathbf{B},\, \mathbf{P},\, \mathbf{C}_0,\, \mathbf{K},\, \mathsf{aux}\big)=1\big],\end{aligned}$$

where

$$\begin{aligned}&(\mathbf{S}, \mathbf{P}, \mathsf{aux}) \leftarrow\!\$\ \mathsf{Samp}(1^\lambda), \mathbf{B} \leftarrow\!\$\ \mathbf{Z}_q^{n\times m}, \\ &\mathbf{C}_0 \leftarrow\!\$\ \mathbf{Z}_q^{m'\times m}, \mathbf{C}' \leftarrow\!\$\ \mathbf{Z}_q^{m'\times t}, \\ &\mathbf{E} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi}^{m'\times m}, \mathbf{E}' \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi'}^{m'\times t}, \\ &\mathbf{K} \leftarrow\!\$\ \mathbf{B}_\gamma^{-1}(\mathbf{P}),\end{aligned}$$

and $\gamma = \mathcal{O}(\sqrt{m \log q})$.

For a function $\kappa = \kappa(\lambda)$ of the security parameter λ, the private-coin binding non-uniform κ-evasive LWE assumption is said to hold, if for every non-uniform sampler Samp and every non-uniform adversary $\mathcal{A}_1$ such that $\mathrm{Size}(\mathsf{Samp}) \le \mathsf{poly}(\lambda')$ and $\mathrm{Size}(\mathcal{A}_1) \le \mathsf{poly}(\kappa)$ for $\lambda'(\lambda) \le \kappa(\lambda)$, there exists another non-uniform adversary $\mathcal{A}_0$ and a polynomial $Q(\cdot)$ such that the following relations hold:

$$\begin{aligned}\mathrm{Adv}^{\mathrm{PRE}}_{\mathcal{A}_0}(\lambda) &\ge \mathrm{Adv}^{\mathrm{POST}}_{\mathcal{A}_1}(\lambda)/Q(\lambda') - \mathrm{negl}(\kappa), \\ \mathrm{Size}(\mathcal{A}_0) &\le Q(\lambda') \cdot \mathrm{Size}(\mathcal{A}_1).\end{aligned}$$

Rather than directly relying on Assumption 2, we adopt Lemma 3.4 from [4] with necessary modifications. This distinguishes between the portion of the auxiliary information that depends on the secret $\mathbf{S}$ and the portion that does not, and claims that if the former portion in the precondition is pseudorandom, the same holds in the postcondition.

Lemma 4. Let $n, m, t, m', q \in \mathbb{N}$ be parameters and λ be a security parameter. Let χ and χ' be Gaussian parameters. Let Samp be a PPT algorithm that takes as input 1^λ and outputs

$$\mathbf{S} \in \mathbb{Z}_q^{m'\times n}, \mathbf{P} \in \mathbb{Z}_q^{n\times t}, \mathsf{aux} = (\mathsf{aux}_1, \mathsf{aux}_2) \in \mathcal{S} \times \{0,1\}^*$$

for some set $\mathcal{S}$. Furthermore, we assume that there exists a public deterministic polynomial-time algorithm $\mathsf{Reconstruct}$ that allows one to derive $\mathbf{P}$ from aux_2, i.e., $\mathbf{P} = \mathsf{Reconstruct}(\mathsf{aux}_2)$. For a non-uniform adversary $\mathcal{A}$, we define the following advantage functions:

$$\begin{aligned}\mathrm{Adv}^{\mathrm{PRE}}_{\mathcal{A}}(\lambda) &:= \Pr\big[\mathcal{A}\big(\mathbf{B},\, \mathbf{P},\, \mathbf{SB}+\mathbf{E},\, \mathbf{SP}+\mathbf{E}',\, \mathsf{aux}_1,\, \mathsf{aux}_2\big)=1\big] \\ &\quad -\ \Pr\big[\mathcal{A}\big(\mathbf{B},\, \mathbf{P},\, \mathbf{C}_0,\, \mathbf{C}',\, \mathbf{c},\, \mathsf{aux}_2\big)=1\big], \\ \mathrm{Adv}^{\mathrm{POST}}_{\mathcal{A}}(\lambda) &:= \Pr\big[\mathcal{A}\big(\mathbf{B},\, \mathbf{P},\, \mathbf{SB}+\mathbf{E},\, \mathbf{K},\, \mathsf{aux}_1,\, \mathsf{aux}_2\big)=1\big] \\ &\quad -\ \Pr\big[\mathcal{A}\big(\mathbf{B},\, \mathbf{P},\, \mathbf{C}_0,\, \mathbf{K},\, \mathbf{c},\, \mathsf{aux}_2\big)=1\big],\end{aligned}$$

where

$$(\mathbf{S}, \mathbf{P}, \mathsf{aux} = (\mathsf{aux}_1, \mathsf{aux}_2)) \leftarrow\!\$\, \mathsf{Samp}(1^\lambda), \mathbf{B} \leftarrow\!\$\, \mathbf{Z}_q^{n\times m},$$
$$\mathbf{C}_0 \leftarrow\!\$\, \mathbf{Z}_q^{m'\times m}, \mathbf{C}' \leftarrow\!\$\, \mathbf{Z}_q^{m'\times t}, \mathbf{c} \leftarrow\!\$\, \mathcal{S},$$
$$\mathbf{E} \leftarrow\!\$\, \mathcal{D}_{\mathbb{Z},\chi}^{m'\times m}, \mathbf{E}' \leftarrow\!\$\, \mathcal{D}_{\mathbb{Z},\chi'}^{m'\times t}, \mathbf{K} \leftarrow\!\$\, \mathbf{B}_\gamma^{-1}(\mathbf{P}),$$

and $\gamma = \mathcal{O}(\sqrt{m \log q})$.

Assume that the private-coin binding non-uniform κ-evasive LWE assumption (Assumption 2) holds. Then, for a function $\kappa := \kappa(\lambda)$ of the security parameter λ, if $\mathrm{Size}(\mathsf{Samp}) \le \mathsf{poly}(\lambda')$ and $\mathrm{Size}(\mathcal{A}_1) \le \mathsf{poly}(\kappa)$ for $\lambda' \le \kappa(\lambda)$, there exists another non-uniform adversary $\mathcal{A}_0$ and a polynomial $Q(\cdot)$ such that the following relations hold:

$$\begin{aligned} \mathrm{Adv}_{\mathcal{A}_0}^{\mathrm{PRE}}(\lambda) &\ge \mathrm{Adv}_{\mathcal{A}_1}^{\mathrm{POST}}(\lambda)/Q(\lambda') - \mathrm{negl}(\kappa), \\ \mathrm{Size}(\mathcal{A}_0) &\le Q(\lambda') \cdot \mathrm{Size}(\mathcal{A}_1). \end{aligned}$$

Proof. The only difference between our lemma and the original lemma in [4] is that $\mathbf{P}$ is added to the adversaries' views. Since the proof of the original lemma does not depend on the presence of $\mathbf{P}$, their proof can be directly applied to the proof of our lemma. □

Additionally, we employ a private-coin hiding evasive LWE assumption introduced in [30]. In this variant, the adversary is not given the public matrices $\mathbf{B}$ and $\mathbf{P}$. This assumption requires $\mathbf{P}$ to be hidden from the adversary. Formally, for a parameter β, the auxiliary information aux should not help the adversary distinguish $\mathbf{P}$ from $\mathbf{P} + \mathbf{R}$, where $\mathbf{R}$ is a random matrix whose entries are sampled uniformly from $\{0, 1, \ldots, \beta\}$. The assumption becomes weaker as β grows. As with the previous assumption, we assume non-uniform adversaries.

Assumption 3 (Private-coin Hiding Non-Uniform κ-Evasive LWE [30]). Let $n, m, t, m', q, \beta \in \mathbb{N}$ be parameters and λ be a security parameter. Let χ and χ' be Gaussian parameters. Let Samp be a PPT algorithm that outputs

$$\mathbf{S} \in \mathbb{Z}_q^{m'\times n}, \mathbf{P} \in \mathbb{Z}_q^{n\times t}, \mathsf{aux} \in \{0,1\}^*$$

on input 1^λ. For non-uniform adversaries $\mathcal{A}_0 = \{\mathcal{A}_{0,\lambda}\}_\lambda$, $\mathcal{A}_1 = \{\mathcal{A}_{1,\lambda}\}_\lambda$, we define the following advantage functions:

$$\begin{aligned} \mathrm{Adv}_{\mathcal{A}_0}^{\mathrm{PRE1}}(\lambda) &:= \Pr\big[\mathcal{A}_0\big(\mathbf{SB} + \mathbf{E},\, \mathbf{SP} + \mathbf{E}',\, \mathsf{aux}\big) = 1\big] \\ &\quad - \Pr\big[\mathcal{A}_0\big(\mathbf{C}_0,\, \mathbf{C}',\, \mathsf{aux}\big) = 1\big], \\ \mathrm{Adv}_{\mathcal{A}_0}^{\mathrm{PRE2}}(\lambda) &:= \Pr\big[\mathcal{A}_0\big(\mathbf{P},\, \mathsf{aux}\big) = 1\big] \\ &\quad - \Pr\big[\mathcal{A}_0\big(\mathbf{P} + \mathbf{R},\, \mathsf{aux}\big) = 1\big], \\ \mathrm{Adv}_{\mathcal{A}_1}^{\mathrm{POST}}(\lambda) &:= \Pr\big[\mathcal{A}_1\big(\mathbf{SB} + \mathbf{E},\, \mathbf{K},\, \mathsf{aux}\big) = 1\big] \\ &\quad - \Pr\big[\mathcal{A}_1\big(\mathbf{C}_0,\, \mathbf{K},\, \mathsf{aux}\big) = 1\big], \end{aligned}$$

where

$$(\mathbf{S}, \mathbf{P}, \mathsf{aux}) \leftarrow\!\$\ \mathsf{Samp}(1^\lambda), \mathbf{B} \leftarrow\!\$\ \mathbf{Z}_q^{n\times m},$$
$$\mathbf{C}_0 \leftarrow\!\$\ \mathbf{Z}_q^{m'\times m}, \mathbf{C}' \leftarrow\!\$\ \mathbf{Z}_q^{m'\times t},$$
$$\mathbf{E} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi}^{m'\times m}, \mathbf{E}' \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi'}^{m'\times t},$$
$$\mathbf{R} \leftarrow\!\$\ \{0,\ldots,\beta\}[n\times t], \mathbf{K} \leftarrow\!\$\ \mathbf{B}_\gamma^{-1}(\mathbf{P}),$$

and $\gamma = \mathcal{O}(\sqrt{m \log q})$.

For a function $\kappa = \kappa(\lambda)$ of the security parameter λ, the private-coin hiding non-uniform κ-evasive LWE assumption is said to hold, if for every non-uniform sampler Samp and every non-uniform adversary $\mathcal{A}_1$ such that $\mathrm{Size}(\mathsf{Samp}) \le \mathsf{poly}(\lambda')$ and $\mathrm{Size}(\mathcal{A}_1) \le \mathsf{poly}(\kappa)$ for $\lambda'(\lambda) \le \kappa(\lambda)$, there exists another non-uniform adversary $\mathcal{A}_0$ and a polynomial $Q(\cdot)$ such that the following relations hold:

$$\begin{aligned}\mathrm{Adv}_{\mathcal{A}_0}^{\mathrm{PRE1}}(\lambda) + \mathrm{Adv}_{\mathcal{A}_0}^{\mathrm{PRE2}}(\lambda) \;&\ge\; \mathrm{Adv}_{\mathcal{A}_1}^{\mathrm{POST}}(\lambda)/Q(\lambda') \;-\; \mathrm{negl}(\kappa),\\ \mathrm{Size}(\mathcal{A}_0) \;&\le\; Q(\lambda') \cdot \mathrm{Size}(\mathcal{A}_1).\end{aligned}$$

In our use of the above assumption, we always set $\mathbf{P} := \bar{\mathbf{P}} + \mathbf{E}$, where $\mathbf{E} \in \mathcal{D}_{\mathbb{Z},\sigma}^{n\times t}$. Under the LWE assumption, this allows us to claim security while distinguishing between the portion of the auxiliary information that depends on the secret $\mathbf{S}$ and the portion that does not, as well as replacing the preimage $\mathbf{K}$ with a random Gaussian matrix in the random case.

Lemma 5. Let $n, m, t, m', q \in \mathbb{N}$ be parameters and λ be a security parameter. Let χ, χ', σ be Gaussian parameters. Let Samp be a PPT algorithm that takes as input 1^λ and outputs

$$\mathbf{S} \in \mathbb{Z}_q^{m'\times n}, \bar{\mathbf{P}} \in \mathbb{Z}_q^{n\times t}, \mathsf{aux} = (\mathsf{aux}_1, \mathsf{aux}_2) \in \mathcal{S} \times \{0,1\}^*$$

for some set $\mathcal{S}$. For a non-uniform adversary $\mathcal{A}$, we define the following advantage functions:

$$\begin{aligned}\mathrm{Adv}_{\mathcal{A}}^{\mathrm{PRE}}(\lambda) :=\ &\Pr\big[\mathcal{A}\big(\mathbf{SB}+\mathbf{E},\, \mathbf{SP}+\mathbf{E}',\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) = 1\big]\\ &-\ \Pr\big[\mathcal{A}\big(\mathbf{C}_0,\, \mathbf{C}',\, \mathbf{c},\, \mathsf{aux}_2\big) = 1\big],\\ \mathrm{Adv}_{\mathcal{A}}^{\mathrm{POST}}(\lambda) :=\ &\Pr\big[\mathcal{A}\big(\mathbf{SB}+\mathbf{E},\, \mathbf{K},\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) = 1\big]\\ &-\ \Pr\big[\mathcal{A}\big(\mathbf{C}_0,\, \mathbf{K}',\, \mathbf{c},\, \mathsf{aux}_2\big) = 1\big],\end{aligned}$$

where

$$(\mathbf{S}, \bar{\mathbf{P}}, \mathsf{aux} = (\mathsf{aux}_1, \mathsf{aux}_2)) \leftarrow\!\$\ \mathsf{Samp}(1^\lambda), \mathbf{B} \leftarrow\!\$\ \mathbf{Z}_q^{n\times m},$$
$$\mathbf{C}_0 \leftarrow\!\$\ \mathbf{Z}_q^{m'\times m}, \mathbf{C}' \leftarrow\!\$\ \mathbf{Z}_q^{m'\times t}, \mathbf{c} \leftarrow\!\$\ \mathcal{S},$$
$$\mathbf{E} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi}^{m'\times m}, \quad \mathbf{E}' \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\chi'}^{m'\times t},$$
$$\mathbf{P} := \bar{\mathbf{P}} + \bar{\mathbf{E}}, \quad \bar{\mathbf{E}} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\sigma}^{n\times t},$$
$$\mathbf{K} \leftarrow\!\$\ \mathbf{B}_\gamma^{-1}(\mathbf{P}), \qquad \mathbf{K}' \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\gamma}^{n\times t},$$

and $\gamma = \mathcal{O}(\sqrt{m \log q})$.

Assume that the private-coin hiding non-uniform κ-evasive LWE assumption (Assumption 3) holds with a parameter β such that $\sigma \geq \beta\lambda^{\omega(1)}$. Then, for a function $\kappa := \kappa(\lambda)$ of the security parameter λ, if $\mathrm{Size}(\mathsf{Samp}) \leq \mathsf{poly}(\lambda')$ and $\mathrm{Size}(\mathcal{A}_1) \leq \mathsf{poly}(\kappa)$ for $\lambda' \leq \kappa(\lambda)$, there exists another non-uniform adversary $\mathcal{A}_0$ and a polynomial $Q(\cdot)$ such that the following relations hold:

$$\begin{aligned} \mathrm{Adv}^{\mathrm{PRE}}_{\mathcal{A}_0}(\lambda) \;&\geq\; \mathrm{Adv}^{\mathrm{POST}}_{\mathcal{A}_1}(\lambda)/Q(\lambda') \;-\; \mathrm{negl}(\kappa), \\ \mathrm{Size}(\mathcal{A}_0) \;&\leq\; Q(\lambda') \;\cdot\; \mathrm{Size}(\mathcal{A}_1). \end{aligned}$$

Proof. Our proof is similar to the proof of Lemma 3.4 in [4], except that we discuss the case where $\mathbf{B}$ and $\mathbf{P}$ remain hidden from the adversary. By the precondition of this lemma, it holds that $\big(\mathbf{SB} + \mathbf{E},\, \mathbf{SP} + \mathbf{E}',\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) \stackrel{\mathrm{c}}{\approx} \big(\mathbf{C}_0,\, \mathbf{C}',\, \mathbf{c},\, \mathsf{aux}_2\big)$. This implies that $\big(\mathsf{aux}_1,\, \mathsf{aux}_2\big) \stackrel{\mathrm{c}}{\approx} \big(\mathbf{c},\, \mathsf{aux}_2\big)$. This additionally implies that

$$\big(\mathbf{C}_0,\, \mathbf{C}',\, \mathbf{c},\, \mathsf{aux}_2\big) \stackrel{\mathrm{c}}{\approx} \big(\mathbf{C}_0,\, \mathbf{C}',\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) \tag{4}$$

because the other samples $\mathbf{C}_0$ and $\mathbf{C}'$ are sampled independently from aux_1. Additionally, we have

$$\begin{aligned} \big(\mathbf{P}, \mathsf{aux}_1, \mathsf{aux}_2\big) =& \big(\bar{\mathbf{P}} + \bar{\mathbf{E}}, \mathsf{aux}_1, \mathsf{aux}_2\big) \\ \stackrel{\mathrm{s}}{\approx}& \big(\bar{\mathbf{P}} + \bar{\mathbf{E}} + \mathbf{R}, \mathsf{aux}_1, \mathsf{aux}_2\big) \\ =& \big(\mathbf{P} + \mathbf{R}, \mathsf{aux}_1, \mathsf{aux}_2\big), \end{aligned} \tag{5}$$

where the statistical indistinguishability follows from the noise flooding (Lemma 3).

By indistinguishabilities in Equations (4) and (5), we can invoke the private-coin hiding evasive-LWE assumption to claim that

$$\big(\mathbf{SB} + \mathbf{E},\, \mathbf{K},\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) \stackrel{\mathrm{c}}{\approx} \big(\mathbf{C}_0,\, \mathbf{K},\, \mathsf{aux}_1,\, \mathsf{aux}_2\big).$$

Recall that by Lemma 4.4 in [34], under the LWE assumption, for arbitrary matrix $\bar{\mathbf{P}} \in \mathbb{Z}_q^{n \times t}$, it holds that

$$(\mathbf{K} \leftarrow\!\$\; \mathbf{B}_\gamma^{-1}(\bar{\mathbf{P}} + \bar{\mathbf{E}})) \stackrel{\mathrm{c}}{\approx} (\mathbf{K}' \leftarrow\!\$\; \mathcal{D}^{n \times t}_{\mathbb{Z},\gamma}).$$

Since aux_1 and aux_2 do not depend on $\mathbf{B}$, we can apply the above indistinguishability to the previous indistinguishability to claim that

$$\big(\mathbf{C}_0,\, \mathbf{K},\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) \stackrel{\mathrm{c}}{\approx} \big(\mathbf{C}_0,\, \mathbf{K}',\, \mathsf{aux}_1,\, \mathsf{aux}_2\big).$$

Since the RHS distribution of the above indistinguishability does not depend on $\mathbf{S}$, we can further claim that

$$\big(\mathbf{C}_0,\, \mathbf{K}',\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) \stackrel{\mathrm{c}}{\approx} \big(\mathbf{C}_0,\, \mathbf{K}',\, \mathbf{c},\, \mathsf{aux}_2\big).$$

Hence, we have

$$\big(\mathbf{SB} + \mathbf{E},\, \mathbf{K},\, \mathsf{aux}_1,\, \mathsf{aux}_2\big) \stackrel{\mathrm{c}}{\approx} \big(\mathbf{C}_0,\, \mathbf{K}',\, \mathbf{c},\, \mathsf{aux}_2\big).$$

□

2.3 Fully Homomorphic Encryption

We adopt a fully homomorphic encryption (FHE) scheme with the following syntax:

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$: it takes as input the security parameter 1^λ and outputs a secret key sk and a public key pk.
- $\mathsf{Enc}(\mathsf{pk}, \mathbf{z}) \to \mathsf{ct}$: it takes as input the public key pk and a message $\mathbf{z}$ and outputs a ciphertext ct.
- $\mathsf{Eval}(\mathsf{pk}, C, \mathsf{ct}) \to \mathsf{ct}'$: it takes as input the public key pk, a circuit C, and a ciphertext ct, and outputs a ciphertext ct'.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to \mathbf{z}'$: it takes as input the secret key sk and a ciphertext ct and outputs a message $\mathbf{z}'$.

For simplicity, the $\mathsf{FHE.Eval}$ algorithm takes a single encrypted message as input.

An FHE scheme is required to satisfy subexponential semantic security, defined as follows.

Definition 1 ((Subexponential) Semantic Security of FHE [25]). An FHE scheme is said to be semantically secure if for all polynomial-sized adversaries $\mathcal{A}$ and pairs of messages $(\mathbf{z}_0, \mathbf{z}_1)$, it holds that

$$|\Pr[\mathcal{A}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathbf{z}_0)) = 1] - \Pr[\mathcal{A}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathbf{z}_1)) = 1]| \leq \mathsf{negl}(\lambda)(\lambda),$$

where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$. The FHE scheme is said to satisfy subexponential semantic security if there exists some constant $\delta \in (0, 1)$ such that for all non-uniform adversaries of size at most 2^{λ^δ}, the advantage defined above is bounded by $2^{-\lambda^\delta}$ for sufficiently large λ.

Additionally, the following lemma requires that (1) a ciphertext can be decrypted by multiplying it with the low-norm secret key vector, and (2) the evaluation algorithm is shallow.

Lemma 6. There exists an FHE scheme parameterized by $q, n, m \geq 2n\lceil \log_2 q \rceil$, $\ell, L, M, \mathsf{dep} \in \mathbb{N}$ such that in addition to subexponential semantic security (Definition 1), the following properties hold:

- **Correctness with Linear Decryption.** The FHE scheme supports the evaluation of vector-valued circuits $C : \{0,1\}^\ell \to \mathbb{Z}_q^{1 \times M}$ of depth at most dep. For every such circuit C and every input $\mathbf{z} \in \{0,1\}^\ell$, let

  $$(\mathbf{t}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda), \quad \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathbf{z}), \quad \mathbf{Y} \leftarrow \mathsf{Eval}(\mathsf{pk}, C, \mathsf{ct}),$$

  where $\mathbf{t} \in \{0, \pm 1\}^{n+1}$ and $\mathbf{Y} \in \mathbb{Z}_q^{(n+1) \times M}$. Then there exists an error vector $\mathbf{e} \in \mathbb{Z}_q^M$ such that

  $$\mathbf{t}\mathbf{Y} = C(\mathbf{z}) + \mathbf{e} \quad \text{and} \quad \|\mathbf{e}\|_\infty \leq \mathsf{poly}(\lambda) \cdot m^{\mathcal{O}(\mathsf{dep})}.$$

– **Shallowness of the FHE Evaluation Algorithm.** For every public key pk, every circuit $C[\mathbf{x}] : \{0,1\}^\ell \to \mathbb{Z}_q^{1\times M}$ of depth at most dep, in which a value $\mathbf{x} \in \{0,1\}^L$ is hardcoded, there exists an efficient algorithm

$$\mathsf{MakeVEvalCkt}(n, m, q, C) \to G_C$$

that takes as input n, m, q and the circuit C and outputs a circuit G_C taking as input a ciphertext ct and a hardcoded value $\mathbf{x} \in \{0,1\}^L$ and outputting a new ciphertext

$$\mathbf{Y} = \mathsf{Eval}(\mathsf{pk}, C[\mathbf{x}], \mathsf{ct}).$$

Moreover, the depth of G_C is bounded by

$$\mathsf{dep}(G_C) \leq \mathsf{dep} \cdot \mathsf{poly}(\log m, \log\log q).$$

Proof. The proof is deferred to Appendix B.1.

2.4 Homomorphic Evaluation of BGG+ Encodings

We adopt the syntax of homomorphic evaluation algorithms used for BGG+ encodings [16] from [79]. For parameters $q, n, t, L, M \in \mathbb{N}$ and $m \geq n\lceil \log_2 q \rceil$, a public matrix for the input encoding is denoted by $\mathbf{A} \in \mathbb{Z}_q^{n\times(L+1)m}$.

The following two deterministic algorithms allow homomorphically evaluating a Boolean circuit $C : \{0,1\}^L \to \{0,1\}^M$ with bits $\mathbf{x} \in \{0,1\}^L$ on the input encoding [16].

– $\mathsf{EvalC}(\mathbf{A}, C) \to \mathbf{A}_C$: it takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times(L+1)m}$ and the circuit C and outputs a matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n\times Mm}$.
– $\mathsf{EvalCX}(\mathbf{A}, C, \mathbf{x}) \to \mathbf{H}_{C,x}$: it takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times(L+1)m}$, the circuit C, and the input $\mathbf{x}$, and outputs a matrix $\mathbf{H}_{C,x} \in \mathbb{Z}_q^{(L+1)m\times Mm}$.

The outputs of these algorithms satisfy the following properties:

– For a circuit C of depth dep, $\left\|\mathbf{H}_{C,x}\right\|_\infty \leq m^{\mathcal{O}(\mathsf{dep})}$ holds.
– For any $\mathbf{A}$, C, and $\mathbf{x}$, it holds that

$$(\mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G})\mathbf{H}_{C,x} = \mathbf{A}_C - C(\mathbf{x}) \otimes \mathbf{G},$$

where $\mathbf{A}_C \leftarrow \mathsf{EvalC}(\mathbf{A}, C)$ and $\mathbf{H}_{C,x} \leftarrow \mathsf{EvalCX}(\mathbf{A}, C, \mathbf{x})$.

Additionally, we adopt evaluation algorithms for an inner product proposed in [54]. The original algorithms are defined for an inner product between a hidden vector $\mathbf{t} \in \mathbb{Z}_q^t$ and the output of the Boolean circuit $C(\mathbf{x}) \in \{0,1\}^t$; ours in Lemma 7 extends the output of the circuit to a matrix $\mathbf{Y} \in \mathbb{Z}_q^{t\times M}$ by combining a technique introduced in Section 4.1 of [28].

Lemma 7 (Evaluation of BGG+ encodings for a vector-matrix multiplication). For all parameters $q, n, L, t, M \in \mathbb{N}$ and $m \geq n\lceil \log_2 q \rceil$, the following two deterministic algorithms are defined:

- $\mathsf{VMEvalC}((\mathbf{A}, \mathbf{A}_t), G) \to \mathbf{A}_G$: it takes as input matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times (L+1)m}$ and $\mathbf{A}_t \in \mathbb{Z}_q^{n \times tm}$, and a matrix-valued circuit $G : \{0,1\}^L \to \mathbb{Z}_q^{t \times M}$, and outputs a matrix $\mathbf{A}_G \in \mathbb{Z}_q^{n \times Mm}$.
- $\mathsf{VMEvalCX}((\mathbf{A}, \mathbf{A}_t), G, \mathbf{x}) \to \mathbf{H}_{G,x}$: it takes as input matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times (L+1)m}$ and $\mathbf{A}_t \in \mathbb{Z}_q^{n \times tm}$, a matrix-valued circuit $G : \{0,1\}^L \to \mathbb{Z}_q^{t \times M}$, and $\mathbf{x} \in \{0,1\}^L$, and outputs a matrix $\mathbf{H}_{G,x} \in \mathbb{Z}_q^{(L+1+t)m \times Mm}$.

The outputs of these algorithms satisfy the following properties:

- For a circuit G of depth dep,

$$\left\|\mathbf{H}_{G,x}\right\|_\infty \leq t\lceil \log_2 q \rceil m^{\mathcal{O}(\mathsf{dep})}.$$

  In particular, this bound is independent of the output size M.
- For any $\mathbf{A}$, $\mathbf{A}_t$, G, $\mathbf{x}$, and $\mathbf{t}$, it holds that

$$(\mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G},\ \mathbf{A}_t - \mathbf{t} \otimes \mathbf{G})\, \mathbf{H}_{G,x} = \mathbf{A}_G - (\mathbf{t}\mathbf{Y}) \otimes \mathbf{G}, \tag{6}$$

  where

$$\mathbf{A}_G \leftarrow \mathsf{VMEvalC}((\mathbf{A}, \mathbf{A}_t), G), \qquad \mathbf{H}_{G,x} \leftarrow \mathsf{VMEvalCX}((\mathbf{A}, \mathbf{A}_t), G, \mathbf{x}),$$

  and $\mathbf{Y} := G(\mathbf{x})$.

Proof. The proof is deferred to Appendix B.2.

2.5 Indistinguishability Obfuscation

We recall the definition of iO in [11].

Definition 2 (Indistinguishability Obfuscation [11]). For input and output sizes $L, M \in \mathbb{N}$, an indistinguishability obfuscator (iO) scheme is defined by two PPT algorithms $(\mathsf{Obf}, \mathsf{Eval})$ with the following syntax.

- $\mathsf{Obf}(1^\lambda, C) \to \widetilde{C}$: it takes as input a security parameter 1^λ and a circuit $C : \{0,1\}^L \to \{0,1\}^M$ and outputs an obfuscated circuit $\widetilde{C}$.
- $\mathsf{Eval}(\widetilde{C}, \mathbf{x}) \to \mathbf{y}$: it takes as input an obfuscated circuit $\widetilde{C}$ and an input $\mathbf{x} \in \{0,1\}^L$ and outputs the evaluation result $\mathbf{y} \in \{0,1\}^M$.

These algorithms satisfy the following properties.

- **Correctness.** For all $\lambda \in \mathbb{N}$, circuits $C : \{0,1\}^L \to \{0,1\}^M$, and inputs $\mathbf{x} \in \{0,1\}^L$, it holds that

$$\Pr\left[\mathsf{Eval}(\mathsf{Obf}(1^\lambda, C), \mathbf{x}) = C(\mathbf{x})\right] \geq 1 - \mathsf{negl}(\lambda),$$

  where the probability is taken over the random coins of the Obf algorithm.

– **Polynomial Slowdown.** For all $\lambda \in \mathbb{N}$, circuits $C : \{0,1\}^L \to \{0,1\}^M$, it holds that

$$|\mathsf{Obf}(1^\lambda, C)| \leq \mathsf{poly}(|C|).$$

– **Indistinguishability for Equivalent Functionalities.** For all $\lambda \in \mathbb{N}$, pairs of two circuits $C_0, C_1 : \{0,1\}^L \to \{0,1\}^M$ that compute the same function and are of the same size, it holds that

$$\{C_0, C_1, \mathsf{Obf}(1^\lambda, C_0)\} \stackrel{c}{\approx} \{C_0, C_1, \mathsf{Obf}(1^\lambda, C_1)\}$$

2.6 Pseudorandom Oracle Model

We recall the definition of pseudorandom oracle model (PROM) in [64].

Definition 3 (Pseudorandom Oracle Model [64]). Let PRF be a pseudorandom function with a key domain $\{0,1\}^\lambda$. The pseudorandom oracle model for PRF is the model in which all algorithms, including adversaries, can access the oracle O. This oracle internally uses a random permutation $\mathsf{hMap} : \{0,1\}^\lambda \to \{0,1\}^\lambda$ and responds to the following two types of queries:

– $\mathsf{O}(\mathsf{hGen}, K) = \mathsf{hMap}(K)$,
– $\mathsf{O}(\mathsf{hEval}, h, \mathbf{x}) = \mathsf{PRF}(\mathsf{hMap}^{-1}(h), \mathbf{x})$.

2.7 Indistinguishable Pseudorandom Obfuscation

A notion of pseudorandom obfuscation is introduced in [29] with multiple security definitions. The weakest one is called indistinguishable pseudorandom obfuscation (iPRO), which requires that the obfuscation of a keyed circuit $C(K, \cdot)$ is indistinguishable from the obfuscation of another keyed circuit $C(K', \cdot)$ that is functionally equivalent to $C(K, \cdot)$. While iPRO is weaker than iO, its stronger variants are shown to be impossible for general circuits [29]; therefore, we adopt only the iPRO security definition as follows.

Definition 4. For key, input, and output sizes $\ell, L, M \in \mathbb{N}$, an indistinguishable pseudorandom obfuscation (iPRO) scheme for keyed circuits $C : \{0,1\}^\ell \times \{0,1\}^L \to \{0,1\}^M$ is defined by two PPT algorithms $(\mathsf{Obf}, \mathsf{Eval})$ with the following syntax.

– $\mathsf{Obf}(1^\lambda, C, K) \to \widetilde{C}$: it takes as input a security parameter 1^λ, a circuit C, and a key $K \in \{0,1\}^\ell$, and outputs an obfuscated circuit $\widetilde{C}$.
– $\mathsf{Eval}(C, \widetilde{C}, \mathbf{x}) \to \mathbf{y}$: it takes as input a circuit C, an obfuscated circuit $\widetilde{C}$, and an input $\mathbf{x} \in \{0,1\}^L$, and deterministically returns an output $\mathbf{y} \in \{0,1\}^M$.

These algorithms satisfy the following properties.

– **Correctness.** For all $\lambda \in \mathbb{N}$, circuits $C : \{0,1\}^\ell \times \{0,1\}^L \to \{0,1\}^M$, keys $K \in \{0,1\}^\ell$, and inputs $\mathbf{x} \in \{0,1\}^L$, it holds that

$$\Pr\big[\mathsf{Eval}(C, \mathsf{Obf}(1^\lambda, C, K), \mathbf{x}) = C(K, \mathbf{x})\big] \geq 1 - \mathsf{negl}(\lambda),$$

where the probability is taken over the random coins of the Obf algorithm.

– **Indistinguishable Pseudorandomness.** Let $\mathsf{PROSamp}$ be a sampling algorithm that on input 1^λ, outputs

$$(K \in \{0,1\}^\ell, \mathsf{aux}_K \in \{0,1\}^\star).$$

For all $\lambda \in \mathbb{N}$, a circuit $C : \{0,1\}^\ell \times \{0,1\}^L \to \{0,1\}^M$, if

$$\left\{1^\kappa, C, \{C(K,\mathbf{x})\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}_K\right\} \overset{\mathrm{c}}{\approx} \left\{1^\kappa, C, \{\Delta_{\mathbf{x}} \leftarrow\!\$\, \{0,1\}^M\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}_K\right\},$$

for a randomly sampled $(K, \mathsf{aux}_K) \leftarrow\!\$\, \mathsf{PROSamp}(1^\lambda)$, then we have that

$$\left\{C, \mathsf{Obf}(1^\lambda, C, K), \mathsf{aux}_K\right\} \overset{\mathrm{c}}{\approx} \left\{C, \mathsf{Obf}(1^\lambda, C, K'), \mathsf{aux}_K\right\},$$

where $\kappa \geq 2^L$, $(K, \mathsf{aux}_K) \leftarrow\!\$\, \mathsf{PROSamp}(1^\lambda)$, and $K' \in \{0,1\}^\ell$ is such that $C(K,\cdot)$ and $C(K',\cdot)$ are functionally equivalent.

Remark 1. As in Definition 5.3 of [2], we introduce 1^κ to the precondition of the iPRO security to allow the distinguisher in the precondition to run in time $\mathsf{poly}(\kappa)$. We require $\kappa \geq 2^L$ to ensure that the distinguisher can read the entire truth table of the circuit $C(K,\cdot)$.

3 Diamond iO from PFFE: Main Construction

In this section, we introduce a formal definition of PFFE, a new variant of functional encryption that serves as the main building block for our iO scheme. We next construct an iPRO scheme for a class of Boolean circuits using PFFE as a black box, and then bootstrap this PRO scheme to obtain iO for general circuits in the PROM.

3.1 Definition of PFFE

The PFFE scheme is defined as a special type of partially-hiding FE scheme, where the ciphertext is associated with a private input $\mathbf{z}$ and a public input $\mathbf{x}$ [68, 7]. We additionally require that the ciphertext for $(\mathbf{z}, \mathbf{x})$ is represented as a product of a vector depending only on $\mathbf{z}$ and matrices, each of which depends only on one bit of $\mathbf{x}$, which we call the path-factorized structure. To naturally capture the path-factorized structure, we define an extractor algorithm that, given the state of the encryption algorithm, extracts the path-factorized secret factors $\{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],b\in\{0,1\}}$.

Syntax. Define a function class $\mathcal{F} := \{\mathcal{F}_{\ell,L,M}\}$ parameterized by the private and public input sizes $\ell, L \in \mathsf{n}$ and the output size $M \in \mathsf{n}$, where $\mathcal{F}_{\ell,L,M} := \{f : \{0,1\}^\ell \times \{0,1\}^L \to \{0,1\}^M\}$. A PFFE scheme for the function class $\mathcal{F}$, parameterized by $q, \hat{n}, \hat{m} := \mathsf{poly}(\lambda) \in \mathbb{N}$, consists of the following PPT algorithms:

– $\mathsf{Setup}(1^\lambda, 1^\ell, 1^L, 1^M) \to (\mathsf{mpk}, \mathsf{msk})$. It takes as input the security parameter λ and the parameters ℓ, L, M of the function class $\mathcal{F}$, and outputs a pair of a master public key mpk and a master secret key msk. Without loss of generality, we assume that the following algorithms implicitly take λ, ℓ, L, M as input.

- $\mathsf{KeyGen}(\mathsf{msk}, f) \to \mathsf{sk}_f$. It takes as input the master secret key msk and a function $f \in \mathcal{F}_{\ell,L,M}$, and outputs a functional secret key sk_f.
- $\mathsf{Enc}(\mathsf{mpk}, \mathbf{z}, \mathbf{x}) \to \mathsf{ct}$. It takes as input the master public key mpk, a private input $\mathbf{z} \in \{0,1\}^\ell$, and a public input $\mathbf{x} \in \{0,1\}^L$, and outputs a ciphertext $\mathsf{ct} \in \mathcal{CT}_{\lambda,\ell,L,M}$, where $\mathcal{CT}_{\lambda,\ell,L,M}$ is the ciphertext parameterized by λ, ℓ, L, M. The encryption algorithm is decomposed into the following two algorithms:
  - $\mathsf{EncPriv}(\mathsf{mpk}, \mathbf{z}) \to (\mathsf{ct}_\mathbf{z}, \mathsf{st})$. It takes as input the master public key mpk and a private input $\mathbf{z} \in \{0,1\}^\ell$, and outputs a part of the ciphertext $\mathsf{ct}_\mathbf{z} \in \mathcal{CT}^z_{\lambda,\ell,L,M}$ and an auxiliary state st, where $\mathcal{CT}^z_{\lambda,\ell,L,M}$ is the private-input-dependent part of the ciphertext parameterized by λ, ℓ, L, M.
  - $\mathsf{EncPub}(\mathsf{mpk}, \mathbf{x}, \mathsf{st}) \to \mathsf{ct}_\mathbf{x}$. It takes as input the master public key mpk, a public input $\mathbf{x} \in \{0,1\}^L$, and the auxiliary state st, and outputs a part of the ciphertext $\mathsf{ct}_\mathbf{x} \in \mathcal{CT}^x_{\lambda,\ell,L,M}$, where $\mathcal{CT}^x_{\lambda,\ell,L,M}$ is the public-input-dependent part of the ciphertext parameterized by λ, ℓ, L, M.

  Then the ciphertext ct is defined as $\mathsf{ct} := \big(\mathsf{ct}_\mathbf{z}, \mathsf{ct}_\mathbf{x}\big)$.
- $\mathsf{Dec}(\mathsf{mpk}, \mathbf{x}, \mathsf{ct}, f, \mathsf{sk}_f) \to \mathbf{y}$. It takes as input the master public key mpk, a public input $\mathbf{x} \in \{0,1\}^L$, a ciphertext ct, a function $f \in \mathcal{F}_{\ell,L,M}$, and a functional secret key sk_f, and outputs the value $\mathbf{y} \in \{0,1\}^M$.
- $\mathsf{Extract}(\mathsf{st}) \to (\hat{\mathbf{s}}_\mathbf{z}, \{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],b\in\{0,1\}})$. It takes as input the auxiliary state st and outputs the path-factorized secret factors $\hat{\mathbf{s}}_\mathbf{z} \in \mathbb{Z}_q^{\hat{n}}$, $\{\hat{\mathbf{S}}_{1,b} \in \mathbb{Z}_q^{\hat{n}\times\hat{n}}\}_{b\in\{0,1\}}$, $\ldots$, $\{\hat{\mathbf{S}}_{L-1,b} \in \mathbb{Z}_q^{\hat{n}\times\hat{m}}\}_{b\in\{0,1\}}$, $\{\hat{\mathbf{S}}_{L,b} \in \mathbb{Z}_q^{\hat{n}\times\hat{m}}\}_{b\in\{0,1\}}$.

Definition 5 (Correctness of PFFE). A PFFE scheme is said to be correct if for every security parameter $\lambda \in \mathbb{N}$, every set of function parameters $\ell, L, M \in \mathbb{N}$, every function $f \in \mathcal{F}_{\ell,L,M}$, and every private and public input $(\mathbf{z}, \mathbf{x}) \in \{0,1\}^\ell \times \{0,1\}^L$, it holds that

$$\Pr[\mathsf{Dec}(\mathsf{mpk}, \mathbf{x}, \mathsf{Enc}(\mathsf{mpk}, \mathbf{z}, \mathbf{x}), f, \mathsf{sk}_f) = f(\mathbf{z}, \mathbf{x})] = 1,$$

where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^L, 1^M)$ and $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$.

The definition of the path-factorized ciphertext structure specifies the distribution of the ciphertext $\mathsf{ct}_\mathbf{x}$ for every public input $\mathbf{x}$.

Definition 6 (Path-Factorized Ciphertext Structure). A PFFE scheme is said to have the path-factorized ciphertext structure with a Gaussian parameter σ and bound parameters $\beta_s, \beta_e \in \mathbb{N}$, if for every security parameter $\lambda \in \mathbb{N}$, every set of function parameters $\ell, L, M \in \mathbb{N}$, every function $f \in \mathcal{F}_{\ell,L,M}$, and every private and public input $(\mathbf{z}, \mathbf{x}) \in \{0,1\}^\ell \times \{0,1\}^L$, the following conditions hold.

Define

$$\begin{aligned}
&(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^L, 1^M),\\
&(\mathsf{ct}_\mathbf{z}, \mathsf{st}) \leftarrow \mathsf{EncPriv}(\mathsf{mpk}, \mathbf{z}),\\
&\mathsf{ct}_\mathbf{x} \leftarrow \mathsf{EncPub}(\mathsf{mpk}, \mathbf{x}, \mathsf{st}),\\
&(\hat{\mathbf{s}}_\mathbf{z}, \{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],b\in\{0,1\}}) \leftarrow \mathsf{Extract}(\mathsf{st}).
\end{aligned}$$

1. **Path-factorized ciphertext distribution.** It holds that

$$\mathsf{ct}_{\mathbf{x}} \overset{\mathsf{dist}}{=} \hat{\mathbf{s}}_{\mathbf{z}} \left(\prod_{i=1}^{L} \hat{\mathbf{S}}_{i,x_i} \right) + \mathbf{e}_{\mathbf{x}} \pmod{q}, \tag{7}$$

   where $\mathbf{e}_{\mathbf{x}} \leftarrow\!\!\$\ \mathcal{D}^m_{\mathbb{Z},\sigma}$.
2. **Norm bounds on the secret factors.** It holds that

$$\left\| \hat{\mathbf{s}}_{\mathbf{z}} \right\|_\infty \le \beta_s, \qquad \forall i \in [L-1],\, b \in \{0,1\}, \quad \left\| \hat{\mathbf{S}}_{i,b} \right\|_\infty \le \beta_s.$$

   Notably, we do not require a norm bound on the last secret factor $\hat{\mathbf{S}}_{L,b}$ for $b \in \{0,1\}$.
3. **Correctness under bounded errors.** It holds that for every $\mathbf{e}'_{\mathbf{x}} \in \mathbb{Z}^m$ satisfying

$$\left\| \mathbf{e}'_{\mathbf{x}} \right\|_\infty \le \beta_e,$$

   the PFFE scheme is correct as stated in Definition 5 when the ciphertext $\mathsf{ct}_{\mathbf{x}}$ is modified as follows:

$$\widetilde{\mathsf{ct}}_{\mathbf{x}} := \hat{\mathbf{s}}_{\mathbf{z}} \left(\prod_{i=1}^{L} \hat{\mathbf{S}}_{i,x_i} \right) + \mathbf{e}'_{\mathbf{x}} \pmod{q}. \tag{8}$$

The security definition for PFFE requires that ciphertexts for a single private input $\mathbf{z}$ and exponentially many public inputs $\mathbf{x} \in \{0,1\}^L$ remain pseudorandom, even when given a single functional secret key sk_f, provided that the joint distribution of the function outputs $\{f(\mathbf{z},\mathbf{x})\}_{\mathbf{x}\in\{0,1\}^L}$ is pseudorandom. The postcondition adversary also sees the auxiliary LWE samples in which products of path-factorized secrets are used as the LWE secrets.

Definition 7 (Security of PFFE). Let λ be the security parameter and $\ell, L, M \in \mathbb{N}$ be the function parameters. Let $\{\chi_{p,i}\}_{i\in[0,L-1]}$ be the Gaussian parameters. Define a PPT algorithm $\mathsf{PFFESamp}_{\mathbf{z}}$ that on input 1^λ, outputs

$$f \in \mathcal{F}_{\ell,L,M}, \mathbf{z} \in \{0,1\}^\ell, \mathsf{aux}_{\mathbf{z}} \in \{0,1\}^\star.$$

Define $\mathsf{AuxPFFEReal}(\mathsf{st})$ as follows. First compute

$$(\hat{\mathbf{s}}_{\mathbf{z}}, \{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],\, b\in\{0,1\}}) \leftarrow \mathsf{Extract}(\mathsf{st}).$$

For every $i \in [0, L-1]$ and $\mathbf{x}_i \in \{0,1\}^i$, compute

$$\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{z}} \left(\prod_{j=1}^{i} \hat{\mathbf{S}}_{j,x_j} \right) \mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i},$$

where $\mathbf{B}_i \leftarrow\!\!\$\ \mathbb{Z}_q^{\hat{n}\times\hat{m}}$ and $\mathbf{e}_{\mathbf{x}_i} \leftarrow\!\!\$\ \mathcal{D}^{\hat{m}}_{\mathbb{Z},\chi_{p,i}}$ are independent for every $i \in [0, L-1]$ and $\mathbf{x}_i \in \{0,1\}^i$. Output the following auxiliary information:

$$\mathsf{aux}_{\mathsf{pffe}} := \left(\{\mathbf{p}_{\mathbf{x}_i}\}_{i\in[0,L-1],\, \mathbf{x}_i\in\{0,1\}^i} \right).$$

Define $\mathsf{AuxPFFERand}(1^\lambda)$ that outputs the following auxiliary information:

$$\mathsf{aux}^\star_{\mathsf{pffe}} := \left(\{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\!\$\, \mathbb{Z}_q^{1\times \hat{m}}\}_{i\in[0,L-1],\, \mathbf{x}_i\in\{0,1\}^i}\right).$$

For non-uniform adversaries $\mathcal{A}_0 = \{\mathcal{A}_{0,\lambda}\}_\lambda$, $\mathcal{A}_1 = \{\mathcal{A}_{1,\lambda}\}_\lambda$, we define the following advantage functions:

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{PRE}}_{\mathcal{A}_0}(\lambda) &:= \Pr\Big[\mathcal{A}_0\big(f, \mathsf{aux}_{\mathbf{z}}, \{\mathbf{x}, f(\mathbf{z},\mathbf{x})\}_{\mathbf{x}\in\{0,1\}^L}\big) = 1\Big] \\
&\quad - \Pr\Big[\mathcal{A}_0\big(f, \mathsf{aux}_{\mathbf{z}}, \{\mathbf{x}, \mathbf{y}^\star_{\mathbf{x}} \leftarrow\!\$\, \{0,1\}^M\}_{\mathbf{x}\in\{0,1\}^L}\big) = 1\Big], \\
\mathrm{Adv}^{\mathrm{POST}}_{\mathcal{A}_1}(\lambda) &:= \Pr\Big[\mathcal{A}_1\big(f, \mathsf{aux}_{\mathbf{z}}, \mathsf{mpk}, \mathsf{sk}_f, \mathsf{ct}_{\mathbf{z}}, \{\mathbf{x}, \mathsf{ct}_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}_{\mathsf{pffe}}\big) = 1\Big] \\
&\quad - \Pr\Big[\mathcal{A}_1\big(f, \mathsf{aux}_{\mathbf{z}}, \mathsf{mpk}, \mathsf{sk}_f, \mathsf{ct}^\star_{\mathbf{z}}, \{\mathbf{x}, \mathsf{ct}^\star_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}^\star_{\mathsf{pffe}}\big) = 1\Big],
\end{aligned}$$

where

$$\begin{aligned}
&(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^L, 1^M), \\
&(f, \mathbf{z}, \mathsf{aux}_{\mathbf{z}}) \leftarrow \mathsf{PFFESamp}_{\mathbf{z}}(1^\lambda),\ \mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f), \\
&(\mathsf{ct}_{\mathbf{z}}, \mathsf{st}) \leftarrow \mathsf{EncPriv}(\mathsf{mpk}, \mathbf{z}),\ \mathsf{ct}^\star_{\mathbf{z}} \leftarrow\!\$\, \mathcal{CT}^{z}_{\lambda,\ell,L,M}, \\
&\mathsf{aux}_{\mathsf{pffe}} \leftarrow \mathsf{AuxPFFEReal}(\mathsf{st}),\ \mathsf{aux}^\star_{\mathsf{pffe}} \leftarrow \mathsf{AuxPFFERand}(1^\lambda) \\
&\forall \mathbf{x} \in \{0,1\}^L, \mathsf{ct}_{\mathbf{x}} \leftarrow \mathsf{EncPub}(\mathsf{mpk}, \mathbf{x}, \mathsf{st}),\ \mathsf{ct}^\star_{\mathbf{x}} \leftarrow\!\$\, \mathcal{CT}^{x}_{\lambda,\ell,L,M}
\end{aligned}$$

For a function $\kappa = \kappa(\lambda)$ of the security parameter λ, a PFFE scheme is said to be secure with respect to a sampler class $\mathcal{SC}$, if for every non-uniform private-input sampler $\mathsf{PFFESamp}_{\mathbf{z}} \in \mathcal{SC}$ and every non-uniform adversary $\mathcal{A}_1$ such that $\mathrm{Size}(\mathsf{PFFESamp}_{\mathbf{z}}) \le \mathsf{poly}(\lambda')$ and $\mathrm{Size}(\mathcal{A}_1) \le \mathsf{poly}(\kappa)$ for $\lambda'(\lambda) \le \kappa(\lambda)$, there exists another non-uniform adversary $\mathcal{A}_0$ and a polynomial $Q(\cdot)$ such that the following relations hold:

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{PRE}}_{\mathcal{A}_0}(\lambda) &\ \ge\ \mathrm{Adv}^{\mathrm{POST}}_{\mathcal{A}_1}(\lambda)/Q(\lambda') \ -\ \mathrm{negl}(\kappa), \\
\mathrm{Size}(\mathcal{A}_0) &\ \le\ Q(\lambda') \cdot \mathrm{Size}(\mathcal{A}_1).
\end{aligned}$$

3.2 Construction

We provide an iPRO scheme for a class of keyed Boolean circuits with ℓ key bits, L input bits, M output bits, and depth bounded by dep.

Parameters. The construction depends on the following parameters:

- A security parameter: $\lambda \in \mathbb{N}$.
- Lattice dimension $n \in \mathbb{N}$ and $m \ge (n+1)\lceil \log_2 q \rceil$.
- Gaussian parameters σ_E and σ_S.

Algorithm 1 iPRO.Obf

Input: $1^\lambda, C, K$.
Output: $\widetilde{C}$.

1: $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{PFFE.Setup}(1^\lambda, 1^\ell, 1^L, 1^M)$.
2: $\mathbf{z} := K \in \{0,1\}^\ell$.
3: $(\mathsf{ct}_\mathbf{z}, \mathsf{st}) \leftarrow \mathsf{PFFE.EncPriv}(\mathsf{mpk}, \mathbf{z})$.
4: $(\hat{\mathbf{s}}_\mathbf{z}, \{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],b\in\{0,1\}}) \leftarrow \mathsf{PFFE.Extract}(\mathsf{st})$.
5: For every $i \in [0, L-1]$, $(\mathbf{B}_i, \mathbf{B}_{i,\gamma_B}^{-1}) \leftarrow \mathsf{TrapGen}(1^{\hat{n}}, 1^{\hat{m}}, q)$.
6: $\mathbf{p}_\epsilon := \hat{\mathbf{s}}_\mathbf{z}\mathbf{B}_0 + \mathbf{e}_\epsilon$, where $\mathbf{e}_\epsilon \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma_E}^{\hat{m}}$.
7: For every $i \in [L-1]$ and $b \in \{0,1\}$, $\mathbf{K}_{i,b} \leftarrow \mathbf{B}_{i-1,\gamma_B}^{-1}(\hat{\mathbf{S}}_{i,b}\mathbf{B}_i + \mathbf{E}_{K,i,b})$, where $\mathbf{E}_{K,i,b} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma_S}^{\hat{n}\times\hat{m}}$.
8: For every $b \in \{0,1\}$, $\mathbf{K}_{L,b} \leftarrow \mathbf{B}_{L-1,\gamma_B}^{-1}(\hat{\mathbf{S}}_{L,b} + \mathbf{E}_{K,L,b})$, where $\mathbf{E}_{K,L,b} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma_S}^{\hat{n}\times\hat{m}}$.
9: $\mathsf{sk}_C \leftarrow \mathsf{PFFE.KeyGen}(\mathsf{msk}, C)$.
10: **return**

$$\widetilde{C} := \left(\mathsf{mpk}, \mathsf{ct}_\mathbf{z}, \mathbf{p}_\epsilon, \{\mathbf{K}_{i,b}\}_{i\in[L],b\in\{0,1\}}, \mathsf{sk}_C\right)$$

Algorithm 2 iPRO.Eval

Input: $C, \widetilde{C}, \mathbf{x} \in \{0,1\}^L$.
Output: $\mathbf{y} \in \{0,1\}^M$.

1: Parse $\widetilde{C}$ as $(\mathsf{mpk}, \mathsf{ct}_\mathbf{z}, \mathbf{p}_\epsilon, \{\mathbf{K}_{i,b}\}_{i\in[L],b\in\{0,1\}}, \mathsf{sk}_C)$.
2: $\widetilde{\mathsf{ct}}_\mathbf{x} := \mathbf{p}_\epsilon \prod_{i\in[L]} \mathbf{K}_{i,x_i}$.
3: $\mathbf{y} \leftarrow \mathsf{PFFE.Dec}(\mathbf{x}, (\mathsf{ct}_\mathbf{z}, \widetilde{\mathsf{ct}}_\mathbf{x}), C, \mathsf{sk}_C)$.
4: **return** $\mathbf{y}$.

- Gaussian parameter for trapdoors and preimages: $\sigma_B = \omega(\sqrt{\hat{n}\lceil\log_2 q\rceil \log \hat{m}})$.
- Parameters for the PFFE scheme: $q, \hat{n}, \hat{m} \in \mathbb{N}$.
- Parameters for the path-factorized structure of the PFFE scheme (Definition 6): σ'_E, β_s, β_e that satisfy

$$\beta_e = \sigma'_E\sqrt{\lambda} > \sigma_E\sqrt{\lambda}\left(\gamma B\sqrt{\lambda}\right)^L + L\beta_s\sigma S\sqrt{\lambda}\cdot(\hat{n}\beta_s)^{L-1} \tag{9}$$

Construction. We rely on the PFFE scheme for the function class $\mathcal{F}_{\ell,L,M}$ with parameters $q, \hat{n}, \hat{m}$. The construction of the iPRO scheme is given in Algorithms 1 and 2.

Lemma 8. Assume that the PFFE scheme used in the iPRO scheme is correct as stated in Definition 5 and has the path-factorized ciphertext structure defined in Definition 6. Then the iPRO scheme defined in Algorithms 1 and 2 is correct as stated in Definition 4.

Proof. The proof is deferred to Appendix C.1. □

Security. The security of the iPRO scheme is stated in the following theorem.

Theorem 1 (Security of the iPRO scheme from PFFE). Let $\kappa = \lambda^{L^2 \log \lambda}$. Assume that the following holds:

- the LWE assumption (Assumption 1).
- the private-coin hiding non-uniform κ-evasive LWE assumption (Assumption 3).
- the PFFE scheme that is correct (Definition 5), has the path-factorized ciphertext structure (Definition 6), and is secure with respect to the private-input sampler class induced by the PRO sampler class (Definition 7).

Then the PRO scheme defined in Algorithms 1 and 2 satisfies iPRO as stated in Definition 4.

Proof. The proof is deferred to Appendix C.2. □

Remark 2 (Safeguards against known attacks on private-coin evasive LWE). Both our iPRO construction and the resulting PFFE-based iO construction rely on private-coin evasive LWE. Recent works [77, 30, 62, 44, 3, 59] have proposed counterexamples and attacks against private-coin evasive LWE. In Appendix D, we explain why our constructions avoid these known attacks.

As shown in [29], iPRO implies iO for general circuits in the PROM.

Theorem 2 (iO implied by our iPRO scheme). Assume that the same assumptions as in Theorem 1 hold. Then, in the PROM (Definition 2.6), there exists an iO scheme with properties as stated in Definition 2.

Proof. By Lemma 8 and Theorem 1, under the given assumptions, Algorithms 1 and 2 instantiate an iPRO scheme. Therefore the bootstrapping method of [29] implies the existence of an iO scheme for general circuits in the PROM under the same assumptions. □

4 Construction of PFFE from All-Product LWE

In this section, we present the concrete construction of the PFFE scheme, proven secure under the all-product LWE assumption, along with the private-coin binding evasive LWE assumption. This is followed by an analysis of the all-product LWE assumption.

4.1 Construction of PFFE

We provide a construction of a PFFE scheme for a function f represented as a Boolean circuit with ℓ private input bits, L public input bits, M output bits, maximum depth dep, and maximum size size_F.

Parameters. The construction depends on the following parameters:

- A security parameter: $\lambda \in \mathbb{N}$.

Algorithm 3 PFFE.Setup

Input: $1^\lambda, 1^\ell, 1^L, 1^M$.
Output: $\mathsf{mpk}, \mathsf{msk}$.
1: $(\mathbf{B}_L, \mathbf{B}_{L,\gamma_B}^{-1}) \leftarrow \mathsf{TrapGen}(1^{n_B}, 1^{m_B}, q)$.
2: $\mathbf{A}_{\mathsf{att}} \leftarrow\!\$\ \mathbb{Z}_q^{(n+1)\times L'm}$ and $\mathbf{A}_t \leftarrow\!\$\ \mathbb{Z}_q^{(n+1)\times(n+1)m}$.
3: $\mathsf{mpk} := (\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t)$.
4: $\mathsf{msk} := \left(\mathbf{B}_L, \mathbf{B}_{L,\gamma_B}^{-1}, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t\right)$.
5: **return** $\mathsf{mpk}, \mathsf{msk}$.

- A modulus $q \in \mathbb{N}$.
- Lattice dimension $n \in \mathbb{N}$ and $m \geq (n+1)\lceil \log_2 q \rceil$.
- Gaussian parameters σ_E.
- A total bit size of the FHE encryptions of $\ell + \lambda$ bits: $\ell' := \Theta(\ell + \lambda) \in \mathbb{N}$.
- $L' := 1 + \ell' + L$.
- Parameters for trapdoors and preimages:

$$n_B = (L' + (n+1))(n+1), m_B = \mathcal{O}(n_B \lceil \log_2 q \rceil),$$
$$\sigma_B = \omega(\sqrt{n_B \lceil \log_2 q \rceil \log m_B}).$$

- A bound $B \in \mathbb{N}$, which is exponentially smaller than $\frac{q}{4}$.

Construction. We rely on a pseudorandom function $\mathsf{PRF}_B : \{0,1\}^\lambda \times \{0,1\}^\lambda \to [-q/4 + B, q/4 - B]^{1\times\ell}$ with depth $\mathsf{dep}_{\mathsf{PRF}_B} = \mathsf{poly}(\lambda)$ and on an FHE scheme satisfying the properties in Lemma 6. The formal construction is parameterized by q, $\hat{n} = n_B$, $\hat{m} = m_B$, and is provided in Algorithms 3, 4, 5, 6, and 7.

Correctness. Lemma 8 states the correctness of the PFFE scheme.

Lemma 8 (Correctness of the PFFE scheme). The PFFE scheme defined in Algorithms 3, 4, 5, 6, and 7 is correct (Definition 5).

Proof. The proof is deferred to Appendix E.1. □

Path-Factorized Structure. Lemma 9 states that the PFFE scheme has the path-factorized ciphertext structure defined in Definition 6.

Lemma 9 (Path-factorized structure of the ciphertext). The PFFE scheme defined in Algorithms 3, 4, 5, 6, and 7 has the path-factorized ciphertext structure (Definition 6) with parameters σ_E, $\beta_s := 1$, $\beta_e := \sigma_E\sqrt{\lambda}$.

Proof. Recall that, by the definition of $\mathsf{ct}_{\mathbf{x}}$, we have

$$\begin{aligned}\mathsf{ct}_{\mathbf{x}} :=& \mathbf{c}_{\mathbf{x}} \\ =& \hat{\mathbf{s}}_{\mathbf{z}}\Big(\prod_{i\in[L]} \hat{\mathbf{S}}_{i,x_i}\Big) + \mathbf{e}_{\mathbf{x}},\end{aligned}$$

Algorithm 4 PFFE.KeyGen

Input: msk, f.
Output: sk_f.

1: Parse msk as $\left(\mathbf{B}_L, \mathbf{B}_{L,\gamma_B}^{-1}, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t\right)$.
2: $\mathbf{K}_{\mathsf{att}} \leftarrow\!\$\ \mathbf{B}_{L,\gamma_B}^{-1}(\mathbf{u}_1 \otimes \mathbf{A}_{\mathsf{att}} - \mathbf{T}_{\mathsf{att}} \otimes \mathbf{G})$, where $\mathbf{T}_{\mathsf{att}} := \begin{pmatrix}\mathbf{I}_{L'} \\ \mathbf{0}\end{pmatrix}$.
3: $\mathbf{K}_t \leftarrow\!\$\ \mathbf{B}_{L,\gamma_B}^{-1}(\mathbf{u}_1 \otimes \mathbf{A}_t - \mathbf{T}_t \otimes \mathbf{G})$, where $\mathbf{T}_t := \begin{pmatrix}\mathbf{0} \\ \mathbf{I}_{n+1}\end{pmatrix}$.
4: Let $F[\mathbf{x}] : \{0,1\}^L \to \mathbb{Z}_q^{1\times M}$ be a vector-valued circuit that, for any hard-coded value $\mathbf{x} \in \{0,1\}^L$, computes

$$F[\mathbf{x}](\mathbf{z}, K_B) := \left\lfloor \frac{q}{2} \right\rfloor f(\mathbf{z}, \mathbf{x}) + \mathsf{PRF}_B(K_B, \mathbf{x}). \tag{10}$$

5: $G_F \leftarrow \mathsf{MakeVEvalCkt}(n, m, q, F)$.
6: $\mathbf{A}_G \leftarrow \mathsf{VMEvalC}((\mathbf{A}_{\mathsf{att}}, \mathbf{A}_t), G_F)$.
7: $\mathbf{A}'_G := \mathbf{A}_G\left(\mathbf{I}_M \otimes \mathbf{G}^{-1}(\mathbf{u}_{n+1})\right)$.
8: $\mathbf{K}_G \leftarrow\!\$\ \mathbf{B}_{L,\gamma_B}^{-1}(\mathbf{u}_1 \otimes \mathbf{A}'_G)$.
9: **return** $\mathsf{sk}_f := (\mathbf{K}_{\mathsf{att}}, \mathbf{K}_t, \mathbf{K}_G)$.

where $\mathbf{e}_{\mathbf{x}} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\sigma_E}$. Therefore, the ciphertext $\mathsf{ct}_{\mathbf{x}}$ output by Algorithm 6 is identically distributed as the right-hand side of Eq. (7). This satisfies the first condition.

The norms $\left\|\hat{\mathbf{s}}_{\mathbf{z}}\right\|_\infty$ and $\{\left\|\hat{\mathbf{S}}_{i,b}\right\|_\infty\}_{i\in[L-1],b\in\{0,1\}}$ are bounded by β_s because the maximum absolute value of the entries in $\mathbf{m}_{\mathbf{z}}$, $\mathbf{s}_\epsilon$, $\mathbf{M}_{i,b}$, and $\mathbf{S}_{i,b}$ is 1. Thus the second condition holds.

Moreover, the correctness proof in Lemma 8 still holds when the ciphertext $\mathsf{ct}_{\mathbf{x}}$ is replaced with $\widetilde{\mathsf{ct}}_{\mathbf{x}}$ defined in Eq. (8). This is because Eq. 20 stands for any error $\mathbf{e}'_{\mathbf{x}}$, and the error bound in Eq. (21) still holds when $\left\|\mathbf{e}_{\mathbf{x}}\right\|_\infty$ is replaced with $\left\|\mathbf{e}'_{\mathbf{x}}\right\|_\infty$ as long as $\left\|\mathbf{e}'_{\mathbf{x}}\right\|_\infty \le \sigma_E\sqrt{\lambda}$. Hence, the third condition in Definition 6 is also satisfied. □

Security. Before providing the security proof of the PFFE scheme, we first introduce the formal definition of the all-product LWE assumption. In a nutshell, we consider the following instances:

$$\left\{\mathbf{s}_{\mathbf{x}_L}^T(\mathbf{A} - \mathbf{x}_L^T \otimes \mathbf{G}) + \mathbf{e}_{A,\mathbf{x}_L}^T\right\}_{\mathbf{x}_L\in\{0,1\}^L},$$
$$\left\{\mathbf{s}_{\mathbf{x}_i}^T\mathbf{B}_i + \mathbf{e}_{B,\mathbf{x}_i}^T\right\}_{i\in[0,L],\mathbf{x}_i\in\{0,1\}^i},$$

where $\mathbf{s}_{\mathbf{x}_i}^T := \mathbf{s}^T\prod_{j\in[i]}\mathbf{S}_{j,x_{i,j}}$.

The assumption claims that even if the adversary can simultaneously see the above instances for every pattern indexed by $i \in [0, L]$ and $\mathbf{x}_i \in \{0,1\}^i$, all of them remain pseudorandom, which is formally defined as follows:

Algorithm 5 PFFE.EncPriv

Input: $\mathsf{mpk}, \mathbf{z}$.
Output: $\mathsf{ct}_{\mathbf{z}}$, st.
1: Parse mpk as $(\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t)$.
2: $(\mathbf{t}, \mathsf{pk}) \leftarrow \mathsf{FHE.Setup}(1^\lambda)$, where $\mathbf{t} \in \{0, \pm 1\}^{n+1}$.
3: $K_B \leftarrow\!\!\$\ \{0,1\}^\lambda$.
4: $\mathsf{ct}_{\mathbf{z}} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, (\mathbf{z}, K_B))$.
5: $\mathbf{m}_{\mathbf{z}} := (1, \mathbf{z}', \mathbf{0}_L, \mathbf{t}) \in \{0,1\}^{L'+n+1}$, where $\mathbf{z}' \in \{0,1\}^{\ell'}$ is a bit string of $\mathsf{ct}_{\mathbf{z}}$.
6: $\bar{\mathbf{s}}_\epsilon \leftarrow\!\!\$\ \{0, \pm 1\}^n$ and $\mathbf{s}_\epsilon := (\bar{\mathbf{s}}_\epsilon, -1)$.
7: $\hat{\mathbf{s}}_{\mathbf{z}} := \mathbf{m}_{\mathbf{z}} \otimes \mathbf{s}_\epsilon$.
8: **for** $i \in [L]$ **do**
9: $\quad \bar{\mathbf{S}}_{i,b} \leftarrow\!\!\$\ \{0, \pm 1\}^{n \times n}$ and $\mathbf{S}_{i,b} := \begin{pmatrix} \bar{\mathbf{S}}_{i,b} & 0 \\ 0 & 1 \end{pmatrix}$.
10: $\quad \mathbf{M}_{i,b} := \begin{pmatrix} \mathbf{I}_{\ell'+i} & b & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{I}_{(L-i)+n+1} \end{pmatrix}$.
11: **end for**
12: For every $i \in [L-1]$ and $b \in \{0,1\}$, $\hat{\mathbf{S}}_{i,b} := \mathbf{M}_{i,b} \otimes \mathbf{S}_{i,b}$.
13: For every $b \in \{0,1\}$, $\hat{\mathbf{S}}_{L,b} := (\mathbf{M}_{L,b} \otimes \mathbf{S}_{L,b})\mathbf{B}_L$.
14: $\mathsf{st} := (\hat{\mathbf{s}}_{\mathbf{z}}, \{\hat{\mathbf{S}}_{i,b}\}_{i \in [L],\, b \in \{0,1\}})$.
15: **return** $\mathsf{ct}_{\mathbf{z}}$, st.

Algorithm 6 PFFE.EncPub

Input: $\mathsf{mpk}, \mathbf{x}, \mathsf{st}$.
Output: $\mathsf{ct}_{\mathbf{x}}$.
1: Parse mpk as $(\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t)$.
2: Parse st as $\left(\hat{\mathbf{s}}_{\mathbf{z}}, \{\hat{\mathbf{S}}_{i,b}\}_{i \in [L],\, b \in \{0,1\}}\right)$.
3: $\mathbf{e}_{\mathbf{x}} \leftarrow\!\!\$\ \mathcal{D}^m_{\mathbb{Z}, \sigma_E}$.
4: $\mathbf{c}_{\mathbf{x}} := \hat{\mathbf{s}}_{\mathbf{z}} (\prod_{i \in [L]} \hat{\mathbf{S}}_{i,x_i}) + \mathbf{e}_{\mathbf{x}}$.
5: **return** $\mathsf{ct}_{\mathbf{x}} := \mathbf{c}_{\mathbf{x}}$.

Assumption 4 (All-Product LWE Assumption). Let $n, m_A \geq (n+1)\lceil \log_2 q \rceil$, $\{m_{B,i}\}_{i \in [0,L]}$, q, $L \in \mathbb{N}$ be parameters, and $\lambda \in \mathbb{N}$ be a security parameter. Let σ_A and $\left\{\sigma_{B,i}\right\}_{i \in [0,L]}$ be Gaussian parameters. We define the following two dis-

Algorithm 7 PFFE.Dec

Input: $\mathsf{mpk}, \mathbf{x}, \mathsf{ct} := (\mathsf{ct}_{\mathbf{z}}, \mathsf{ct}_{\mathbf{x}}), f, \mathsf{sk}_f$.
Output: $\mathbf{y} \in \{0,1\}^M$.

Parse mpk as $(\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t)$.
Parse sk_f as $(\mathbf{K}_{\mathsf{att}}, \mathbf{K}_t, \mathbf{K}_G)$.
Parse $\mathsf{ct}_{\mathbf{x}}$ as $\mathbf{c}_{\mathbf{x}}$.
Convert $\mathsf{ct}_{\mathbf{z}}$ into a bit string $\mathbf{z}'$.
$\mathbf{c}_{\mathbf{x},\mathsf{att}} := \mathbf{c}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}$ and $\mathbf{c}_{\mathbf{x},t} := \mathbf{c}_{\mathbf{x}}\mathbf{K}_t$.
Define a circuit $F[\mathbf{x}]$ as in Eq. (10).
$G_F \leftarrow \mathsf{MakeVEvalCkt}(n, m, q, F)$.
$\mathbf{H}_{G,\hat{\mathbf{z}},\mathbf{x}} \leftarrow \mathsf{VMEvalCX}((\mathbf{A}_{\mathsf{att}}, \mathbf{A}_t), G_F, (\hat{\mathbf{z}}, \mathbf{x}))$.
$\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}} := \mathbf{H}_{G,\hat{\mathbf{z}},\mathbf{x}} \left(\mathbf{I}_M \otimes \mathbf{G}^{-1}(\mathbf{u}_{n+1})\right)$.
$\mathbf{v}_{\mathbf{x}} := (\mathbf{c}_{\mathbf{x},\mathsf{att}}, \mathbf{c}_{\mathbf{x},t})\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}}$.
$\mathbf{d}_{\mathbf{x}} := \mathbf{p}_{\mathbf{x}}\mathbf{K}_G$.
$\mu_{\mathbf{x}} := \mathbf{v}_{\mathbf{x}} - \mathbf{d}_{\mathbf{x}}$.
For each $i \in [M]$, set $y_i := 0$ if $\mu_i \in [-\frac{q}{4}, \frac{q}{4})$ and 1 otherwise.
return $\mathbf{y} := (y_1, \ldots, y_M)$.

tributions:

$$\mathcal{D}_{\mathsf{APLWE},0} := \begin{pmatrix} \mathbf{A}, \{\mathbf{B}_i\}_{i\in[0,L]}, \\ \left\{\mathbf{c}_{A,\mathbf{x}_L} := \mathbf{s}_{\mathbf{x}_L}(\mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}) + \mathbf{e}_{A,\mathbf{x}_L}\right\}_{\mathbf{x}_L\in\{0,1\}^L} \\ \left\{\mathbf{c}_{B,\mathbf{x}_i} := \mathbf{s}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{B,\mathbf{x}_i}\right\}_{i\in[0,L],\mathbf{x}_i\in\{0,1\}^i} \end{pmatrix}$$

$$\mathcal{D}_{\mathsf{APLWE},1} := \begin{pmatrix} \mathbf{A}, \{\mathbf{B}_i\}_{i\in[0,L]}, \\ \left\{\mathbf{c}_{A,\mathbf{x}_L} \leftarrow\$ \mathbb{Z}_q^{1\times Lm_A}\right\}_{\mathbf{x}_L\in\{0,1\}^L} \\ \left\{\mathbf{c}_{B,\mathbf{x}_i} \leftarrow\$ \mathbb{Z}_q^{1\times m_{B,i}}\right\}_{i\in[0,L],\mathbf{x}_i\in\{0,1\}^i} \end{pmatrix},$$

where

- $\mathbf{A} \leftarrow\$ \mathbb{Z}_q^{(n+1)\times Lm_A}$,
- $\mathbf{B}_i \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m_{B,i}}$ for every $i \in [0, L]$,
- $\mathbf{e}_{A,\mathbf{x}_L} \leftarrow\$ \mathcal{D}_{\mathbb{Z},\sigma_A}^{Lm_A}$ for every $\mathbf{x}_L \in \{0,1\}^L$,
- $\mathbf{e}_{B,\mathbf{x}_i} \leftarrow\$ \mathcal{D}_{\mathbb{Z},\sigma_{B,i}}^{m_{B,i}}$ for every $i \in [0, L]$ and $\mathbf{x}_i \in \{0,1\}^i$,
- $\bar{\mathbf{s}}_\epsilon \leftarrow\$ \{0, \pm 1\}^n$,
- $\bar{\mathbf{S}}_{i,b} \leftarrow\$ \{0, \pm 1\}^{n\times n}$ for every $i \in [L]$ and $b \in \{0,1\}$,
- $\mathbf{s}_{\mathbf{x}_i} := \left(\bar{\mathbf{s}}_{\mathbf{x}_i} \prod_{j\in[i]} \bar{\mathbf{S}}_{i,x_{i,j}}, -1\right)$ for every $i \in [0, L]$ and $\mathbf{x}_i \in \{0,1\}^i$.

For a function $\kappa = \kappa(\lambda)$, the all-product LWE assumption is said to hold if for every non-uniform adversary $\mathcal{A}$ such that $\mathrm{Size}(\mathcal{A}) \le \mathsf{poly}(\kappa)$, it holds that

$$\left|\Pr\left[\mathcal{A}\left(\mathcal{D}_{\mathsf{APLWE},0}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathcal{D}_{\mathsf{APLWE},1}\right) = 1\right]\right| \le \mathsf{negl}(\kappa).$$

We now state the security of the concrete PFFE scheme.

Theorem 3 (Security of the PFFE scheme). Let $\kappa = \lambda^{L^2 \log \lambda}$. Assume that the following holds:

- the private-coin binding non-uniform κ-evasive LWE assumption (Assumption 2).
- the all-product LWE assumption (Assumption 4).
- the subexponential semantic security of the FHE scheme for the security parameter Λ (Definition 1).
- the subexponential security of the PRF scheme denoted by $\mathsf{PRF}_B : \{0,1\}^\lambda \times \{0,1\}^\lambda \to [-q/4 + B, q/4 - B]^{1\times\ell}$ for the security parameter Λ.

Then, the PFFE scheme defined in Algorithms 3, 4, 5, 6, and 7 is secure in the sense of Definition 7.

Proof. The proof is deferred to Appendix E.2.

4.2 Cryptanalysis of the All-Product LWE Assumption

We conduct a cryptanalytic analysis of the all-product LWE assumption. We mainly analyze the BGG+ encodings $\mathbf{c}_{A,\mathbf{x}_L}$ because the distribution $\mathcal{D}_{\mathsf{APLWE},0}$ in Assumption 4 remains pseudorandom under the subexponential hardness of the LWE assumption once $\mathbf{c}_{A,\mathbf{x}_L}$ is removed. In other words, the main obstacle to reducing all-product LWE to the standard LWE assumption lies in the BGG+ encodings $\mathbf{c}_{A,\mathbf{x}_L}$.

Analysis of exact rank attacks on matrix PRFs. We compare all-product LWE with the exact rank attack on read-once matrix PRFs [33]. A width-w matrix PRF has the form

$$F(\mathbf{x}) := \mathbf{u}_L(\prod_{i\in[L]} \mathbf{M}_{i,x_i})\mathbf{u}_R^T,$$

where $\mathbf{M}_{i,b} \in \mathbb{Z}_q^{w\times w}$. The attack chooses $\rho > w$ prefixes and suffixes and forms the matrix $\mathbf{V}$ whose (i,j)-entry is $F(\mathbf{y}^{(i)}\|\mathbf{z}^{(j)})$. For a matrix PRF, this matrix factors through dimension w, so $\mathrm{rank}(\mathbf{V}) \le w$; for a random function, $\mathbf{V}$ is full rank with high probability.

The encoding $\mathbf{c}_{A,\mathbf{x}_L}$ has a similar path-product structure, except that the encoding contains independent LWE errors for every input $\mathbf{x} \in \{0,1\}^L$. In other words, if $\mathbf{e}_{A,\mathbf{x}_L}$ is zero, each column of $\mathbf{c}_{A,\mathbf{x}_L}$ can be viewed as an output of the matrix-PRF. Thus, for the same prefixes and suffixes as above, the rank attack applied to the fixed coordinate k sees

$$\widetilde{\mathbf{V}} = \mathbf{V} + \mathbf{E},$$

where $\mathbf{V}$ is the noiseless matrix-PRF part and the entries of $\mathbf{E}$ are sampled independently from $\mathcal{D}_{\mathbb{Z},\sigma_A}$. Therefore, whether the exact rank attack succeeds is

governed by the rank of the perturbed matrix $\widetilde{\mathbf{V}}$. What matters is whether the independent errors accidentally preserve the low-rank affine relations satisfied by the noiseless matrix-PRF matrix $\mathbf{V}$.

In what follows, we show that the probability that the independent errors preserve these exact low-rank relations is negligible. Consequently, the independent LWE errors in the all-product LWE samples prevent the direct exact rank attack from distinguishing the encodings $\mathbf{c}_{A,\mathbf{x}_L}$ from random vectors.

When we view each element of $\mathbf{V}$ as the matrix PRF output, the corresponding rank-attack matrix factors through dimension

$$w_{\mathsf{AP}} := (L+1)(n+1).$$

Thus the noiseless matrix $\mathbf{V}$ satisfies

$$\operatorname{rank}(\mathbf{V}) \le w_{\mathsf{AP}}.$$

The smallest nontrivial exact rank test therefore uses[8]

$$\rho = w_{\mathsf{AP}} + 1.$$

For simplicity, suppose first that q is prime and work over $\mathbb{F}_q$.[9] Let

$$\eta := 1 - \max_{a \in \mathbb{F}_q} \Pr_{e \leftarrow \mathcal{D}_{\mathbb{Z},\sigma_A}} [e \bmod q = a].$$

Equivalently, $\mathcal{D}_{\mathbb{Z},\sigma_A} \bmod q$ is η-dense over $\mathbb{F}_q$. Assume $\eta = \Omega(1)$.

For the minimal test size $\rho = w_{\mathsf{AP}} + 1$, the rank condition

$$\operatorname{rank}_{\mathbb{F}_q}(\mathbf{V} + \mathbf{E}) \le w_{\mathsf{AP}}$$

is exactly the singularity event for the $\rho \times \rho$ matrix $\mathbf{V} + \mathbf{E}$, i.e., the event that this matrix is not full-rank over $\mathbb{F}_q$.

We compare this singularity event with the same event for a uniformly random $\rho \times \rho$ matrix over $\mathbb{F}_q$. Conditioning on the fixed noiseless matrix $\mathbf{V}$, the entries of $\mathbf{V} + \mathbf{E}$ are independent entrywise translates of $\mathcal{D}_{\mathbb{Z},\sigma_A} \bmod q$. Translations preserve η-density, so Maples' Theorem [69, Theorem 1.2] applies in the same way to these entrywise translates. Hence

$$\Pr[\mathbf{V} + \mathbf{E} \text{ is non-singular over } \mathbb{F}_q] = \prod_{j=1}^{\infty}(1 - q^{-j}) + O(e^{-c\eta\rho}),$$

where $c > 0$ is an absolute constant.

[8] We focus on the smallest exact rank test because it is the most delicate case. Larger exact rank tests impose additional affine consistency constraints on the extra rows. These can be bounded by the standard Odlyzko-type argument, but we omit this since the minimal test already captures the weakest exact-rank condition.

[9] If q is composite, the same argument applies modulo each prime divisor $p \mid q$.

A uniformly random entry over $\mathbb{F}_q$ is $(1-1/q)$-dense. Applying the same theorem to a uniformly random matrix $\mathbf{U}_q \in \mathbb{F}_q^{\rho\times\rho}$, we also have

$$\Pr[\mathbf{U}_q \text{ is non-singular over } \mathbb{F}_q] = \prod_{j=1}^{\infty}(1-q^{-j}) + O(e^{-c(1-1/q)\rho}).$$

Since $q \geq 2$, the second error term is $O(e^{-c'\rho})$ for an absolute constant $c' > 0$. Taking complements to pass from nonsingularity to singularity gives

$$\begin{aligned}\left|\Pr\left[\mathrm{rank}_{\mathbb{F}_q}(\mathbf{V}+\mathbf{E}) \leq w_{\mathsf{AP}}\right] - \Pr\left[\mathrm{rank}_{\mathbb{F}_q}(\mathbf{U}_q) \leq w_{\mathsf{AP}}\right]\right| \\ \leq O(e^{-c\eta\rho}) + O(e^{-c'\rho}). \end{aligned} \tag{11}$$

Substituting $\rho = w_{\mathsf{AP}} + 1$, and using that $w_{\mathsf{AP}} = (L+1)(n+1) = \mathsf{poly}(\lambda)$, we have $\rho = \omega(\log\lambda)$. Thus both terms in the RHS of Eq. (11) are negligible. Hence this rank test has negligible distinguishing advantage. Consequently, the independent LWE errors in the all-product LWE samples prevent the direct exact rank attack from distinguishing the encodings $\mathbf{c}_{A,\mathbf{x}_L}$ from random vectors.

Comparison of all-product LWE with tensor LWE. To study a relation between all-product LWE and tensor LWE [79], we introduce a variant of tensor LWE as follows:[10]

Definition 8 (A Variant of Tensor LWE). Let λ be a security parameter and n, $m_A \geq (n+1)\lceil\log_2 q\rceil$, q, $L \in \mathbb{N}$ be parameters bounded by $\mathsf{poly}(\lambda)$. Let σ and γ be Gaussian parameters. For all $\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(Q)} \in \{0,1\}^L$, we have

$$\mathbf{A}, \left\{\mathbf{s}\mathbf{R}_i(\mathbf{A}-\mathbf{x}^{(i)}\otimes\mathbf{G}) + \mathbf{e}_i, \mathbf{R}_i\right\}_{i\in[Q]} \overset{\mathrm{c}}{\approx} \mathbf{A}, \{\mathbf{c}_i, \mathbf{R}_i\}_{i\in[Q]},$$

where $\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{(n+1)\times Lm_A}$, $\mathbf{s} \leftarrow\!\$\ \mathbb{Z}_q^{1\times(n+1)m_A}$, $\mathbf{R}_i \leftarrow \mathcal{D}_{\mathbb{Z},\gamma}^{(n+1)m_A\times(n+1)}$, $\mathbf{e}_i \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{Lm_A}$, and $\mathbf{c}_i \leftarrow\!\$\ \mathbb{Z}_q^{1\times Lm_A}$.

This assumption was originally justified because, if $\mathbf{A}$ and $\mathbf{G}$ in Definition 8 are replaced with a low-norm random matrix and the identity matrix $\mathbf{I}$, respectively, then the LWE assumption with a non-uniform public matrix implies the tensor LWE assumption. However, BGG+ encodings defined in this way support key-homomorphic evaluation only for $\mathbf{NC}^1$ circuits.

To bring all-product LWE closer to the form of tensor LWE, we modify the distribution $\mathcal{D}_{\mathsf{APLWE},0}$ in Assumption 4 as follows:

$$\mathcal{D}'_{\mathsf{APLWE},0} := \left(\begin{array}{l}\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{(n+1)\times Lm_A}, \mathbf{D} \leftarrow\!\$\ \mathbb{Z}_q^{(n+1)\times(n+1)}, \{\mathbf{B}_i \leftarrow\!\$\ \mathbb{Z}_q^{(n+1)\times m_{B,i}}\}_{i\in[0,L]}, \\ \left\{\mathbf{c}_{A,\mathbf{x}_L} := \mathbf{s}_{\mathbf{x}_L}\mathbf{D}(\mathbf{A}-\mathbf{x}_L\otimes\mathbf{G}) + \mathbf{e}_{A,\mathbf{x}_L}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{c}_{B,\mathbf{x}_i} := \mathbf{s}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{B,\mathbf{x}_i}\right\}_{i\in[0,L],\mathbf{x}_i\in\{0,1\}^i}, \end{array}\right).$$

[10] Concretely, we insert a public matrix $\mathbf{R} \in \mathbb{Z}_q^{(n+1)\times(n+1)}$ between the row secret $\mathbf{s}$ and the public matrix $(\mathbf{A}-\mathbf{x}_i\otimes\mathbf{G})$, using dimensions consistent with Assumption 4.

At a high level, if we can prove that $\mathbf{s}_{\mathbf{x}_L}\mathbf{D}$ is pseudorandom for each $\mathbf{x}_L \in \{0,1\}^L$, then we can show that each $\mathbf{c}_{A,\mathbf{x}_L}$ is also pseudorandom by applying the LWE assumption 2^L times. It holds that

$$\begin{aligned}\mathbf{c}_{A,\mathbf{x}_L} &:= \mathbf{s}_{\mathbf{x}_L}\mathbf{D}(\mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}) + \mathbf{e}_{A,\mathbf{x}_L} \\ &= (\mathbf{s}_{\mathbf{x}_L}\mathbf{D} + \mathbf{e}_{D,\mathbf{x}_L})(\mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}) - \mathbf{e}_{D,\mathbf{x}_L}(\mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}) + \mathbf{e}_{A,\mathbf{x}_L}, \quad (12)\end{aligned}$$

where $\mathbf{e}_{D,\mathbf{x}_L} \leftarrow \mathcal{D}^{n+1}_{\mathbb{Z},\sigma}$ for each $\mathbf{x}_L \in \{0,1\}^L$. If the leakage from the structured error is statistically hidden by the independent error—that is, if

$$-\mathbf{e}_{D,\mathbf{x}_L}\big(\mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}\big) + \mathbf{e}_{A,\mathbf{x}_L} \approx \mathbf{e}_{A,\mathbf{x}_L} \quad (13)$$

holds, then the LWE assumption implies that $\mathbf{s}_{\mathbf{x}_L}\mathbf{D} + \mathbf{e}_{D,\mathbf{x}_L}$ is pseudorandom for each $\mathbf{x}_L \in \{0,1\}^L$. As in tensor LWE, if $\mathbf{A}$ and $\mathbf{G}$ in $\mathbf{c}_{A,\mathbf{x}_L}$ are replaced with a low-norm random matrix and the identity matrix $\mathbf{I}$, respectively, then the indistinguishability in Eq. (13) holds by noise flooding.

Importantly, if we could regard $\mathbf{e}_{D,\mathbf{x}_L}$ as a secret vector sampled independently for each $\mathbf{x}_L \in \{0,1\}^L$, then the sample

$$-\mathbf{e}_{D,\mathbf{x}_L}\big(\mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}\big) + \mathbf{e}_{A,\mathbf{x}_L}$$

would be pseudorandom under the subexponential hardness of the LWE assumption. However, in Eq. (12), $\mathbf{e}_{D,\mathbf{x}_L}$ is used simultaneously as an LWE error term. Therefore, we still do not know whether assuming that indistinguishability (13) holds as stated is more heuristic than the tensor LWE assumption.

5 Implemented Optimizations and Concrete Efficiency Estimates

5.1 Challenges in Implementing a Log-Depth Key-Homomorphic PRF

The matrix-product FE-to-iO transformation makes input insertion efficient, but it does not by itself make the end-to-end Diamond iO procedure efficient. The remaining bottleneck is the final PFFE decryption, which must homomorphically evaluate a PRF over key-homomorphic encodings.

We first considered implementing a key-homomorphic evaluation of a log-depth PRF. Such a PRF would bound the input-length-dependent noise growth logarithmically, and therefore would be the most direct and efficient way to improve the final PFFE decryption bottleneck. However, we concluded that implementing this approach with noise growth small enough to keep the lattice parameters in a practical range is difficult.

The approach we explored was to evaluate, instead of a standard PRF, a weak PRF, namely a PRF that is secure only against random inputs. This choice is natural because weak PRFs tend to admit shallower circuit implementations

than full PRFs, both in theoretical analyses and in concrete candidate constructions [24]. In the Naor–Reingold design [71], such a weak PRF is bootstrapped into a full PRF with only logarithmic-order noise growth.

However, according to our estimates, even homomorphically evaluating a single weak PRF is difficult within the range of practical noise growth.[11] In the Naor–Reingold design, the weak PRF must take a random input whose length is at least the security parameter. Therefore, for security levels above 100 bits, the input length is also above 100 bits, which implies that any Boolean circuit representing the weak PRF has depth at least 7.

On the other hand, for the largest lattice parameters we used, each FHE multiplication over encodings increases the noise by at least 220 bits from the encoding layer and by at least 20 bits from the FHE ciphertext layer. Thus, the required homomorphic evaluation of the PRF incurs at least 1680 bits of noise growth. This already consumes about 96% and 47% of the modulus size that can achieve 128-bit security for lattice dimensions 2^{16} and 2^{17}, respectively [23].[12]

Moreover, when we consider the depth of concrete weak-PRF candidates, the noise growth is estimated to be even larger. For example, homomorphically evaluating the crypto dark matter weak PRF [17] requires at least 16 levels of homomorphic XOR operations at 128-bit security. Importantly, because we use GSW FHE as the FHE scheme simulated over encodings, even an XOR operation requires one homomorphic multiplication.[13] As a result, this weak PRF requires at least 16 levels of homomorphic multiplication, which we estimate to cause 3840 bits of noise growth. We also considered other weak-PRF candidates such as [10], but did not find one that can be implemented with smaller noise growth.

5.2 Key-Homomorphic PRF via Specialized Noise Refreshing

Our choice of the FHE and PRF schemes. The use of the PRF scheme in our PFFE implementation imposes two requirements on the FHE scheme used in our PFFE implementation. First, the PRF output produced by FHE evaluation must be placed in the low-order part of the $\mathbb{Z}_q$ plaintext, while the FHE noise must remain small enough not to affect the high-order position that encodes the output bit y. Second, the plaintext range available for the PRF output must be close to $q/4$, so that the PRF mask can cover the decryption-noise leakage.[14] We instantiate the FHE scheme in our implementation with the (ring-)GSW FHE

[11] The noise refreshing we discuss below cannot be applied before the PRF output is obtained because the noise refreshing already requires pseudorandom values to prepare fresh errors and masks.

[12] The dimension $n = 2^{16}$ is the largest dimension used in our estimates, while $n = 2^{17}$ is the largest dimension presented as a concrete parameter example in the guideline for FHE implementations [23]. To the best of our knowledge, $n = 2^{17}$ is also the largest dimension typically used in FHE implementations.

[13] Concretely, for input bits x and y, the output is computed as $x + y - 2xy$.

[14] In the security proof, specifically in the precondition of evasive LWE, the mask and error $\mathsf{PRF}_B(K_B, \mathbf{x}) + e$ must not only remain smaller than $q/4$, but also have a distribution that is statistically close to uniform over $[-q/4, q/4]$ [1].

scheme [50], since it satisfies both requirements and can be implemented with small noise growth using lookup tables, as shown in Appendix G.3.[15]

In GSW FHE, each ciphertext encrypts a single bit as its plaintext, and the scheme supports homomorphic evaluation of Boolean circuits. Because of this bit-level nature, homomorphic PRF evaluation quickly becomes too deep for practical noise growth; as discussed in Subsection 5.1, this remains problematic even for several weak-PRF candidates.

We ultimately implement the required homomorphic PRF evaluation by homomorphically evaluating Goldreich's constant-depth PRG [51], which can be implemented with multiplicative depth 3, and then converting this PRG into a PRF using the Goldreich–Goldwasser–Micali (GGM) design [52]. Recall that the GGM construction turns a length-doubling PRG $G : \{0,1\}^\rho \to \{0,1\}^{2\rho}$ into a PRF by interpreting the input $\mathbf{x} = (x_1, \ldots, x_L)$ as a path in a binary tree: starting from the secret seed K_B, it repeatedly applies G and keeps the left or right half according to x_i. The final seed reached at the leaf, or a fixed portion of it, is used as $\mathsf{PRF}_B(K_B, \mathbf{x})$; in our implementation, this computation is carried out homomorphically using Goldreich's constant-depth PRG as G.

We verified that one invocation of this PRG can be simulated over BGG+ encodings with practical noise growth, concretely with less than 1500 bits of noise growth in our setting. However, in the GGM-based PRF, the final noise grows linearly with the PRF input length, which in our setting is the evaluator's input length L. Therefore, if we evaluate this homomorphic PRF directly over BGG+ encodings, for $L \geq 2$, the noise growth already exceeds our admissible range.[16]

Noise-refreshing optimization for GGM-based PRF evaluation over encodings. We address this noise growth by refreshing the encoding-side noise after each PRG evaluation. As a result, the key-homomorphic-encoding layer no longer adds noise that grows with the PRF input length; the remaining input-length-dependent noise is essentially the same as in plain homomorphic PRF evaluation on FHE ciphertexts. For example, when $L = 10$, the final noise growth is below 1300 bits, and the batching technique introduced later extends this to longer evaluator inputs.

The noise-refreshing paradigm for BGG+ encodings, together with its first concrete instantiation, was introduced in [60]. Informally, given a BGG+ encoding of a value $x \in \mathbb{Z}_q$ under secret $\mathbf{s}$, denoted by $\mathbf{c} := \mathbf{s}(\mathbf{A} - x\mathbf{G}) + \mathbf{e}$, noise refreshing transforms it into a fresh BGG+ encoding of the same value x under the same secret $\mathbf{s}$,

$$\mathbf{c}' := \mathbf{s}(\mathbf{A}' - x\mathbf{G}) + \mathbf{e}',$$

where the new noise $\mathbf{e}'$ has norm bounded independently of the old noise $\mathbf{e}$, and the new public matrix $\mathbf{A}'$ is derived from $\mathbf{A}$ without depending on $\mathbf{s}$ or x.

[15] Although each GSW ciphertext encrypts a single bit, a larger integer can be represented by encrypting its bits separately, scaling the ciphertext of each bit by the corresponding power of two, and summing the resulting ciphertexts.

[16] We assume that a modulus bit-length of 2000 bits is the largest size that our implementation can handle without running out of memory.

We do not directly adopt the concrete refreshing procedure of [60]. In our setting, it faces essentially the same obstacle as the lattice-based PRF candidates discussed above: it requires homomorphic multiplication modulo q followed by homomorphic rounding, and simulating such operations over BGG+ encodings leads to prohibitive noise growth. Instead, we instantiate the same noise-refreshing paradigm in a form specialized to our GGM-based PRF evaluation. In the remainder of this subsection, we explain this specialized noise-refreshing procedure.

Noise refreshing when homomorphic PRF is available. Recall that in the GGM-based PRF, at the beginning of round $i \in \{0, \ldots, L-1\}$, the evaluator holds an encoding of a pseudorandom ρ-bit seed $\mathbf{r}_{\mathbf{x}_i} \in \{0,1\}^\rho$, where $\mathbf{x}_i := (x_1, \ldots, x_i)$ is the length-i prefix of the input $\mathbf{x} = \mathbf{x}_L \in \{0,1\}^L$. The evaluator homomorphically evaluates the length-doubling PRG $G : \{0,1\}^\rho \to \{0,1\}^{2\rho}$ on this encoded seed, obtaining encodings of the two candidate next seeds:

$$G(\mathbf{r}_{\mathbf{x}_i}) = \left(\mathbf{r}_{\mathbf{x}_i,0}, \mathbf{r}_{\mathbf{x}_i,1}\right).$$

It then carries to the next round the candidate selected by the next input bit x_{i+1}, namely $\mathbf{r}_{\mathbf{x}_{i+1}} := \mathbf{r}_{\mathbf{x}_i,x_{i+1}}$.

Although this may sound circular, we first describe the refreshing procedure assuming temporary access to homomorphic PRF evaluation over BGG+ encodings: under this assumption, we use the homomorphic PRF to refresh the BGG+ encodings of the current seed $\mathbf{r}_{\mathbf{x}_i}$. Assume that the modulus factors as $q = \prod_{t\in\{1,\ldots,h\}} q_t$, where the moduli $q_1, \ldots, q_h$ are coprime. For each $t \in \{1, \ldots, h\}$, we write $[x]_t := x \bmod q_t$. Let L_{fhe} denote the bit length of an FHE ciphertext encrypting a ρ-bit seed. Let $\hat{\mathbf{r}}_{\mathbf{x}_i} \in \{0,1\}^{L_{\mathsf{fhe}}}$ be the bit string of the FHE encryption of the current seed $\mathbf{r}_{\mathbf{x}_i}$, and let

$$\mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_i}} := \mathbf{s}\left(\mathbf{A}_i - \hat{\mathbf{r}}_{\mathbf{x}_i} \otimes \mathbf{G}\right) + \mathbf{e}_{\hat{\mathbf{r}}_{\mathbf{x}_i}}$$

be its BGG+ encoding under the secret $\mathbf{s}$, where $\mathbf{A}_i$ is the public matrix associated with round i.

If we ignore security, then by giving the evaluator decoders $\mathbf{d}_{i,q_t}$ that enable the following computation, the evaluator can obtain zero-noise encodings.

$$\begin{aligned}\mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_i},q_t} &:= \mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_i}}\left(\mathbf{I} \otimes \mathbf{G}^{-1}(\frac{q}{q_t}\mathbf{G})\right) - \mathbf{d}_{i,q_t} \bmod q \\ &= \frac{q}{q_t}\left(\mathbf{s}\left(\mathbf{A}_{i+1} - \hat{\mathbf{r}}_{\mathbf{x}_i} \otimes \mathbf{G}\right)\right) + \mathbf{e}_{i,q_t} \bmod q.\end{aligned}$$

As long as the noise in $\mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$ is less than $\frac{q}{2q_t}$, the evaluator can recover the corresponding RNS component:

$$\left\lceil \frac{q_t}{q}\mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_i},q_t} \right\rfloor := \left[\mathbf{s}\left(\mathbf{A}_{i+1} - \hat{\mathbf{r}}_{\mathbf{x}_i} \otimes \mathbf{G}\right)\right]_t.$$

That is, the evaluator obtains the q_t-slot of the RNS representation of the noise-free encoding

$$\mathbf{c}'_{\hat{\mathbf{r}}_{\mathbf{x}_i}} := \mathbf{s}\left(\mathbf{A}_{i+1} - \hat{\mathbf{r}}_{\mathbf{x}_i} \otimes \mathbf{G}\right) \bmod q.$$

Therefore, by recombining these RNS components, the evaluator obtains $\mathbf{c}'_{\hat{\mathbf{r}}_{\mathbf{x}_i}} \bmod q$ without any noise. Unfortunately, the noise-refreshing procedure above is insecure for two reasons.

1. The refreshed encoding $\mathbf{c}'_{\hat{\mathbf{r}}_{\mathbf{x}_i}}$ contains no noise, so the adversary can recover the secret $\mathbf{s}$ from it.
2. The scaled encodings $\mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$ reveal the error terms placed in the low-order bits, which again allows the adversary to recover the secret $\mathbf{s}$.

A straightforward solution is to compute the required pseudorandomness by homomorphically evaluating a PRF for each prefix $\mathbf{x}_i \in \{0,1\}^i$, and to make only the encodings with the corresponding pseudorandom terms added to them decodable. More concretely, the evaluator evaluates the PRF over the encodings of the encrypted PRF key K_B and the input prefix $\mathbf{x}_i$, and then decrypts the result using the encoding of the FHE decryption key $\mathbf{t}$. This yields encodings of the scaled fresh error $\frac{q}{q_t}\mathbf{e}'_{\hat{\mathbf{r}}_{\mathbf{x}_i}}$ and of the mask $\mathbf{m}_{\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$ for each $t \in \{1,\ldots,h\}$, which we denote by $\mathbf{c}_{e',\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$ and $\mathbf{c}_{m,\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$, respectively.[17] Then the evaluator adds $\mathbf{c}_{e',\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$ and $\mathbf{c}_{m,\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$ to the scaled encoding $\mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_i},q_t}$ before rounding the RNS component. The decoder $\mathbf{d}_{i,q_t}$ is chosen for this masked target, so the rounded value corresponds to the refreshed encoding with the fresh error added, while the mask hides the low-order leakage that would otherwise expose the old error. Although this idea addresses the security issues mentioned above, this creates a circular dependency: the purpose of introducing noise refreshing was precisely to make such pseudorandom masks computable over BGG+ encodings in the first place!

Noise refreshing without homomorphic PRF evaluation. We break this circular dependency by generating the errors and masks from a PRG rather than from a PRF. The key observation is that, in the $(i-1)$-st round of the GGM-based PRF evaluation, the evaluator already holds an encoding of the FHE-ciphertext bit string $\hat{\mathbf{r}}_{\mathbf{x}_{i-1}} \in \{0,1\}^{L_{\mathsf{fhe}}}$ for the current-prefix seed $\mathbf{r}_{\mathbf{x}_{i-1}}$. Therefore, if the PRG can derive, from the underlying seed, the errors and masks corresponding to just two branches $x_i = 0$ and $x_i = 1$, then these values can be used to refresh the encodings of the next-round seeds $\mathbf{c}_{\hat{\mathbf{r}}_{\mathbf{x}_{i-1}},x_i}[\ell]$. Equivalently, compared with the standard GGM construction, where each PRG invocation expands a ρ-bit seed only into two ρ-bit next seeds, our design expands the PRG

[17] One subtle point is that the mask is sampled independently for each modulus q_t, whereas the fresh error $\mathbf{e}'_{\hat{\mathbf{r}}_{\mathbf{x}_i}}$ is shared across the different q_t's up to the scaling factor. Because we assume $\left|\mathbf{e}'_{\hat{\mathbf{r}}_{\mathbf{x}_i}}\right| < \min_{t\in\{1,\ldots,h\}} q_t$, the RNS recombination of the refreshed encodings still has small noise.

output further so that it also contains the randomness needed for refreshing: the next seed bits, the fresh errors, and the masks.

This remains feasible because Goldreich's PRG can stretch a ρ-bit seed to ρ^e output bits for a constant $e > 1$, whereas each round requires only $\mathcal{O}(\rho)$ bits of randomness. Hence, we can choose a seed length ρ large enough so that the PRG output length ρ^e is at least the total number of bits needed for the next seeds, errors, and masks. We will provide a more formal specification of the key-homomorphic PRF used in our final implementation in the full version.

5.3 Minor Optimizations

In addition to the key-homomorphic PRF, our implementation adopts several minor optimizations that reduce concrete cost. Although the formal construction is written over integer modules for notational simplicity, our implementation uses the corresponding polynomial-ring analogue. For polynomial-ring matrix operations, BGG+ encodings, and trapdoor preimage sampling, we use the implementation techniques and optimizations from [41, 31].

Batching public-input insertion and PRF evaluation. We batch the public-input insertion used in the FE-to-iO transformation. Let τ be the number of public-input bits inserted in one batched round, and let $B_{\mathrm{br}} := 2^\tau$ be the corresponding branch arity. The number of input-insertion rounds is $r := \lceil L/\tau \rceil$. In the unbatched construction in Subsections 3.2 and 4.1, the evaluator performs one sequential preimage multiplication per input bit $x_i \in \{0,1\}$. With batching, each round uses the branch selected by the current input block $(x_i, \ldots, x_{i+\tau-1}) \in \{0,1\}^\tau$, so the number of sequential evaluator-side preimage multiplications decreases from L to r. On the obfuscator side, each batched round requires branch-specific secret factors and preimages for all B_{br} branch values. Thus the obfuscator-side work and the amount of branch-specific material in the obfuscated circuit increase with the branch arity $B_{\mathrm{br}} = 2^\tau$, while the number of sequential evaluator-side multiplication rounds decreases to $r = \lceil L/\tau \rceil$. This is also significant to maintain the modulus q small because the dominant input-insertion noise growth depends on the number of sequential preimage multiplications, namely r, rather than directly on the branch arity B_{br}.

The key-homomorphic PRF computation is batched in the same sense. We implement the GGM-based PRF evaluation using a higher-arity tree. In each batched round, the current input block $(x_i, \ldots, x_{i+\tau-1}) \in \{0,1\}^\tau$ selects one branch among $B_{\mathrm{br}} = 2^\tau$ branches. The PRG interface is expanded so that each branch has its own next seed and refreshing randomness. The evaluator evaluates only the selected branch, while the branch-specific helper data and the required PRG output length increase with B_{br}.

Additional implementation-level optimizations for the PFFE and iPRO schemes are described in Appendix G.

5.4 Implementation and Estimation Methodology

Our implementation consists of three components: a circuit-description layer, a noise-bound simulator, and a concrete efficiency estimator. The implementation is available in the `mxx` library at `https://github.com/MachinaIO/mxx`.

Circuits. The circuit-description layer specifies the computations evaluated over BGG+ encodings. The same circuit description is used for both the obfuscation and evaluation procedures. Specifically, the obfuscation procedure executes the $\mathsf{VMEvalC}$ algorithm on the circuit description to output $\mathbf{A}_G$, while the evaluation procedure executes the $\mathsf{VMEvalCX}$ algorithm on the same circuit description to output $\mathbf{H}_{G,x}$.[18]

The implemented circuit class is a bounded-integer arithmetic circuit with LUT gates; the formal support for LUT evaluation over BGG+ encodings is described in Appendix G.2. Using this class, we implement the circuit that simulates ring-GSW FHE evaluation and decryption, together with additional circuits for homomorphically evaluating Goldreich's PRG and composing the masks and fresh errors used in noise refreshing and PFFE decryption.

Noise-bound simulator. The noise-bound simulator tracks upper bounds on the relevant noise terms through the implemented computation and identifies moduli that satisfy the modeled correctness constraints. It separately tracks the input-insertion noise from products of GGH15 preimage encodings, the encoding-side noise introduced during circuit evaluation over encodings, and the ordinary FHE ciphertext noise that appears when FHE evaluation and decryption are simulated over encodings.

For PFFE security, the ranges of the PRF masks, including the final PRF_B output and the masks used for noise refreshing, must be large enough to make brute-force recovery of the masked values cost more than 2^λ time. Accordingly, for the final PFFE decryption and for each noise-refreshing step, the simulator checks that the relevant accumulated noise bound fits within the decoding margin that remains after reserving range for the corresponding mask. In addition, for each candidate modulus, the simulator invokes the lattice estimator [6] to check the modeled lattice security of the concrete noise distributions used by the construction. The simulator therefore covers the concrete correctness constraints, the PRF-output range constraint, and the modeled lattice-security constraints that determine the parameters. Constraints not captured by this model, such as proof-level noise-flooding requirements, are treated as limitations of the concrete estimates; we will update the simulator to incorporate these constraints as well.

Concrete efficiency estimator. The concrete efficiency estimator decomposes the computation into representative operation classes, combines measured unit costs with operation counts derived from the generated circuit and parameter set, and outputs an aggregate estimate. Because the full PFFE decryption is too large to benchmark as a single monolithic execution, we measure smaller

[18] However, for memory efficiency, our implementation does not materialize the full matrix $\mathbf{H}_{G,x}$; instead, it applies the corresponding local gate update directly to the vector of the BGG+ encodings. This produces the same result as the formal $\mathsf{VMEvalCX}$ computation while reducing peak memory.

representative units and scale their costs by the number of times those units occur. For each representative unit, we run the same operation five consecutive times and use the total running time divided by five as the representative unit time.

We measured each representative unit on a single H200 GPU. For each measured unit, the benchmark fits within the H200 VRAM capacity, and the measured time includes the data movement performed by the benchmarked operation, including CPU–GPU transfers for native GPU units. Because each measured representative unit fits within the H200 VRAM capacity, distributing independent units across multiple H200 GPUs, with each GPU assigned only units of the measured type, is expected to avoid out-of-memory failures at the unit-execution level. The resulting total time should be interpreted as aggregate H200-GPU time, or equivalently as the ideal running time obtained by executing all units sequentially on one H200 GPU.

The following examples illustrate the scaling rules used by the estimator. For a product between a size-m vector and an $m \times t$ matrix, we benchmark only the inner product between the vector and one column of the matrix and multiply the measured time by t. For preimage sampling with a target matrix $\mathbf{P}$ having t columns, we sample the preimage for one column of $\mathbf{P}$ and multiply the measured time by t. For circuit evaluation over encodings, we measure the running time for each gate type and estimate the total circuit-evaluation running time by summing the measured time for each gate type multiplied by the corresponding gate count.

5.5 Comparison Settings

We compare Diamond iO with the prior FE-to-iO transformation for pseudorandom functionalities of [2]. The target functionality is the PRF described in Section 5: equivalently, the obfuscated function outputs one designated bit of the PRG output in the last round of the GGM design. This is the concrete PRF-evaluation setting used in Figure 1. We use the setting where the depth of the PRF evaluation grows linearly with the input length L; logarithmic-depth PRFs are not included in these concrete estimates for the reasons discussed in Subsection 5.1. For each transformation, we compare the estimated running time of the FE-to-iO transformation and its share in the total evaluation running time, as well as the data size of the materials used for the FE-to-iO transformation and its share in the total obfuscated-circuit data size.

For both Diamond iO and the prior transformation, the FE-to-iO transformation cost is the evaluator-side work up to the point where the evaluator obtains an FE ciphertext containing the obfuscator-side secret K and the full public input $\mathbf{x}$. In Diamond iO, this ciphertext is obtained by matrix-product input insertion using the path-factorized PFFE structure. In the prior transformation, it is obtained by recursive input-insertion rounds. In both cases, the subsequent functional decryption is counted separately as the final FE/PFFE decryption

cost. We define the transformation share as

$$\frac{T_{\text{FE-to-iO}}}{T_{\text{FE-to-iO}} + T_{\text{final-dec}}},$$

where $T_{\text{FE-to-iO}}$ is the transformation cost defined above and $T_{\text{final-dec}}$ is the final FE/PFFE decryption cost.

For a fair comparison, both transformations use the same optimized underlying FE/PFFE backend for the final decryption core. This backend includes the noise-refreshed key-homomorphic PRF evaluation of Section 5 and the lookup-table evaluation method of Appendix G.2. The prior transformation uses the corresponding FE backend inside its recursive input-insertion procedure.[19]

For the prior transformation, we use a conservative lower estimate. Each recursive input-insertion round should homomorphically evaluate a full FE encryption circuit whose output is FE ciphertext material for the next round. We do not implement this full encryption circuit. Instead, we count only the homomorphic PRF-evaluation subcircuit needed in the underlying FE/PFFE backend to mask error leakage, and we ignore the remaining arithmetic, randomness generation, and other overheads in the FE encryption circuit. Thus the prior-transformation cost reported below is intentionally underestimated, and the resulting improvement factor for Diamond iO should be read as a lower estimate.

We also give the prior transformation the same batching advantage as Diamond iO. When Diamond iO inserts τ public-input bits per batched round, we allow one recursive input-insertion round of the prior transformation, estimated as above, to insert τ public-input bits as well. The branch arity in each such round is $B_{\text{br}} = 2^{\tau}$, so the obfuscator-side work and obfuscated material increase with B_{br}, while the number of evaluator-side rounds decreases to $\lceil L/\tau \rceil$. This choice is another conservative choice in favor of the prior transformation.

Our concrete estimator directly targets the pseudorandom-functionality setting rather than iO for general circuits. This is still sufficient for estimating the FE-to-iO transformation cost even when the scheme is bootstrapped to iO for general circuits. Additional computation introduced by the bootstrapping method of [29], or by the target functionality itself, is handled by the final FE/PFFE decryption circuit in both approaches and does not change the input-insertion procedure compared here.

Concrete parameters. In the concrete efficiency estimates, we set the batching parameter to $\tau = 10$ bits and consider input sizes L from 70 bits to 110

[19] The prior transformation requires compact FE for pseudorandom functionalities, such as [1]. Our PFFE construction provides the corresponding FE backend when the public-input length is set to zero; in that specialization, the path-factorized public-input component is absent. The remaining differences are that, in PFFE, the input BGG+ encodings are derived by multiplying the LWE sample $\mathbf{c}_{\mathbf{x}}$ by the preimages $\mathbf{K}_{\mathsf{att}}$ and $\mathbf{K}_t$, and that the BGG+ encoding secret and the FHE decryption key are distinct. We regard the cost difference caused by these changes as negligible compared with the dominant bottleneck, namely key-homomorphic PRF evaluation, while the same final decryption core and backend optimizations remain.

Table 2. Concrete lattice parameters used for concrete efficiency estimates. For each FE-to-iO transformation and input bit size, the table reports the bit length of the modulus and the lattice dimension used in the estimates.

FE-to-iO Transformation	Input size	$\log_2 q$	Lattice dimension n
Prior	70	980	2^{16}
Prior	80	1120	2^{16}
Prior	90	1232	2^{16}
Prior	100	1316	2^{16}
Prior	110	1400	2^{16}
Diamond iO	70	1344	2^{16}
Diamond iO	80	1400	2^{16}
Diamond iO	90	1456	2^{16}
Diamond iO	100	1484	2^{16}
Diamond iO	110	1540	2^{16}

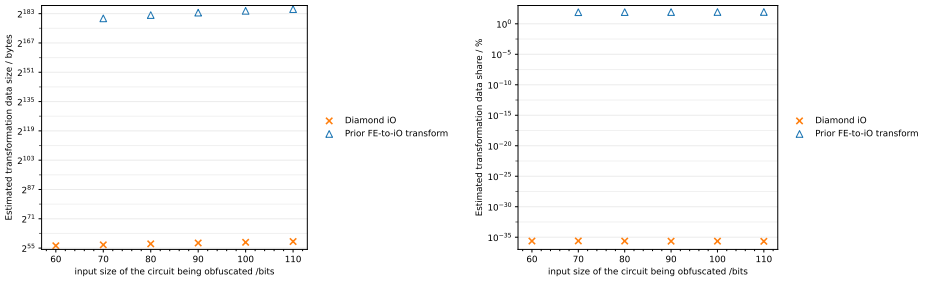
bits. For each input size, the lattice parameters used in the estimate are listed in Table 2. Fresh errors sampled during obfuscation are drawn from a discrete Gaussian distribution with Gaussian parameter $\sigma_E = 4.0$. In the noise-refreshing circuit, each round homomorphically generates the fresh error from a centered binomial distribution with parameter 2. For each selected modulus, we use the lattice dimension for which the lattice estimator [6] reports at least 100 bits of security.[20]

5.6 Estimation Results and Discussion

Figure 1(a) shows that the estimated running time of our FE-to-iO transformation is significantly smaller than that of the prior transformation. Across all tested input sizes, the Diamond iO transformation is at least 2^{118} times faster than the prior transformation. For example, when $L = 110$, the estimated running time of the Diamond iO transformation is 180 hours, while the estimated final PFFE decryption time is 1.03×10^{36} hours ($2^{119.64}$ hours). In contrast, the estimated running time of the prior transformation for the same input size is 7.04×10^{38} hours ($2^{129.05}$ hours).

Figure 1(b) shows the same conclusion from the perspective of the transformation share. For Diamond iO, the transformation share is negligible within the total evaluation running time. By contrast, for the prior transformation, the transformation share is above 99.88% for every input size we tested. **Therefore, Diamond iO removes the concrete FE-to-iO transformation overhead:**

[20] A 128-bit security target would be preferable. In this work, we target 100-bit security to reduce the risk that the resulting parameters become too large and cause out-of-memory failures in the representative-unit benchmarks. We will attempt estimates for 128-bit security in future work. To the best of our knowledge, all prior works presenting implementations of iO evaluate their parameters at security levels of at most 80 bits, so our 100-bit target is already larger than those previous implementation-oriented estimates.



(a) Estimated data size of the FE-to-iO transformation materials.

(b) Estimated FE-to-iO transformation share in total obfuscated-circuit data size.

Fig. 2. Concrete data-size estimates for Diamond iO obfuscations, compared with the prior FE-to-iO transformation for pseudorandom functionalities of [2]. The comparison setting is the one described in Section 5.5, where both transformations are instantiated with the same optimized underlying FE/PFFE backend. In both panels, **smaller vertical-axis values indicate better efficiency**.

after our transformation, the cost of iO evaluation is essentially the cost of the final FE/PFFE decryption.

Figure 2 gives the analogous comparison for obfuscated-circuit data size. Figure 2(a) shows the data size of the materials used for the FE-to-iO transformation. In Diamond iO, these materials are the GGH15 encodings used for matrix-product input insertion. Their estimated data size is at least 2^{123} times smaller than the corresponding materials required by the prior transformation.

Figure 2(b) shows the share of transformation materials in the total obfuscated-circuit data size. For Diamond iO, this share is below $10^{-35}\%$, whereas for the prior transformation it is at least 79.34%. Thus, the FE-to-iO transformation materials are essentially negligible in Diamond iO, while they form a substantial part of the obfuscated circuit in the prior transformation.

The large total data size in both the prior construction and the PFFE backend mainly comes from helper data for key-homomorphic LUT evaluation described in Appendix G.2.[21] The obfuscated circuits contain helper data for each LUT gate so that these gates can be evaluated over BGG+ encodings. This cost belongs to the final FE/PFFE decryption circuit rather than to Diamond iO's matrix-product input-insertion materials, which explains why the transformation-material share of Diamond iO is extremely small even though the total obfuscated-circuit data size remains large.

[21] Standard Boolean-circuit evaluation would avoid this helper data, but its larger depth would make the FHE-simulation noise and modulus size much larger [76]. We therefore view LUT evaluation as a current space-noise tradeoff; moreover, the evaluator only needs the helper data entries selected during evaluation, not all entries at once.

Author Contributions Sora Suegami proposed the Diamond iO construction and led the security proofs and efficiency analysis. The additions introduced in the 2026 version of this paper, namely the PFFE-based modularization, the new cryptanalysis of all-product LWE, the optimizations and end-to-end concrete-efficiency estimates from Section 5 onward, and the updates since November 2025 to the `mxx` library (`https://github.com/MachinaIO/mxx`), were carried out solely by Sora Suegami. Enrico Bottazzi and Gayeong Park contributed to discussions on the idea of the FE-to-iO transformation via matrix products, early investigations of key-homomorphic PRFs, and the writing of the 2025 versions of this paper. Until October 2025, they also contributed to the initial Diamond iO implementation, available at `https://github.com/MachinaIO/diamond-io`, and to the `mxx` library.

References

1. Agrawal, S., Kumari, S., Yamada, S.: Compact pseudorandom functional encryption from evasive LWE. Cryptology ePrint Archive, Report 2024/1719 (2024), `https://eprint.iacr.org/2024/1719`
2. Agrawal, S., Kumari, S., Yamada, S.: Pseudorandom multi-input functional encryption and applications. Cryptology ePrint Archive, Report 2024/1720 (2024), `https://eprint.iacr.org/2024/1720`
3. Agrawal, S., Modi, A., Yadav, A., Yamada, S.: Evasive LWE: Attacks, variants & obfustopia. Cryptology ePrint Archive, Report 2025/375 (2025), `https://eprint.iacr.org/2025/375`
4. Agrawal, S., Rossi, M., Yadav, A., Yamada, S.: Constant input attribute based (and predicate) encryption from evasive and tensor LWE. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, Part IV. Lecture Notes in Computer Science, vol. 14084, pp. 532–564. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38551-3_17
5. Albrecht, M.R., Farshim, P., Han, S., Hofheinz, D., Larraia, E., Paterson, K.G.: Multilinear maps from obfuscation. Journal of Cryptology **33**(3), 1080–1113 (Jul 2020). https://doi.org/10.1007/s00145-019-09340-0
6. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**, 169 – 203 (2015), `https://api.semanticscholar.org/CorpusID:86408`
7. Ananth, P., Jain, A., Lin, H., Matt, C., Sahai, A.: Indistinguishability obfuscation without multilinear maps: New paradigms via low degree weak pseudorandomness and security amplification. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part III. Lecture Notes in Computer Science, vol. 11694, pp. 284–332. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_10
8. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part I. Lecture Notes in Computer Science, vol. 9215, pp. 308–326. Springer, Berlin, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-47989-6_15
9. Apon, D., Huang, Y., Katz, J., Malozemoff, A.J.: Implementing cryptographic program obfuscation. Cryptology ePrint Archive, Report 2014/779 (2014), `https://eprint.iacr.org/2014/779`

10. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012. Lecture Notes in Computer Science, vol. 7237, pp. 719–737. Springer, Berlin, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_42
11. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 1–18. Springer, Berlin, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_1
12. Barrington, D.A.M.: Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. In: 18th Annual ACM Symposium on Theory of Computing. pp. 1–5. ACM Press (May 1986). https://doi.org/10.1145/12130.12131
13. Bitansky, N., Garg, S., Lin, H., Pass, R., Telang, S.: Succinct randomized encodings and their applications. In: Servedio, R.A., Rubinfeld, R. (eds.) 47th Annual ACM Symposium on Theory of Computing. pp. 439–448. ACM Press (Jun 2015). https://doi.org/10.1145/2746539.2746574
14. Bitansky, N., Paneth, O.: ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015: 12th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 9015, pp. 401–427. Springer, Berlin, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-46497-7_16
15. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. In: Guruswami, V. (ed.) 56th Annual Symposium on Foundations of Computer Science. pp. 171–190. IEEE Computer Society Press (Oct 2015). https://doi.org/10.1109/FOCS.2015.20
16. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology – EUROCRYPT 2014. Lecture Notes in Computer Science, vol. 8441, pp. 533–556. Springer, Berlin, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_30
17. Boneh, D., Ishai, Y., Passelègue, A., Sahai, A., Wu, D.J.: Exploring crypto dark matter: New simple PRF candidates and their applications. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018: 16th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 11240, pp. 699–729. Springer, Cham (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_25
18. Boneh, D., Ishai, Y., Sahai, A., Wu, D.J.: Lattice-based SNARGs and their application to more efficient obfuscation. In: Coron, J.S., Nielsen, J.B. (eds.) Advances in Cryptology – EUROCRYPT 2017, Part III. Lecture Notes in Computer Science, vol. 10212, pp. 247–277. Springer, Cham (Apr / May 2017). https://doi.org/10.1007/978-3-319-56617-7_9
19. Boneh, D., Kim, J.: Homomorphic encryption for large integers from nested residue number systems. In: Kalai, Y.T., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025, Part III. Lecture Notes in Computer Science, vol. 16002, pp. 338–370. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01881-6_11
20. Boneh, D., Lewi, K., Montgomery, H.W., Raghunathan, A.: Key homomorphic PRFs and their applications. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013, Part I. Lecture Notes in Computer Science, vol. 8042, pp.

410–428. Springer, Berlin, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_23
21. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011: 8th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 6597, pp. 253–273. Springer, Berlin, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19571-6_16
22. Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) Advances in Cryptology – CRYPTO 2014, Part I. Lecture Notes in Computer Science, vol. 8616, pp. 480–499. Springer, Berlin, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_27
23. Bossuat, J.P., Cammarota, R., Chillotti, I., Curtis, B.R., Dai, W., Gong, H., Hales, E., Kim, D., Kumara, B., Lee, C., Lu, X., Maple, C., Pedrouzo-Ulloa, A., Player, R., Polyakov, Y., Lopez, L.A.R., Song, Y., Yhee, D.: Security guidelines for implementing homomorphic encryption. IACR Communications in Cryptology (CiC) **1**(4), 26 (2024). https://doi.org/10.62056/anxra69p1
24. Boura, C., Couteau, G., Perrin, L., Rotella, Y.: SoK: On shallow weak PRFs A common symmetric building block for MPC protocols. IACR Transactions on Symmetric Cryptology **2025**(3), 289–336 (2025). https://doi.org/10.46586/tosc.v2025.i3.289-336
25. Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Candidate iO from homomorphic encryption schemes. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020, Part I. Lecture Notes in Computer Science, vol. 12105, pp. 79–109. Springer, Cham (May 2020). https://doi.org/10.1007/978-3-030-45721-1_4
26. Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Factoring and pairings are not necessary for IO: Circular-secure LWE suffices. In: Bojanczyk, M., Merelli, E., Woodruff, D.P. (eds.) ICALP 2022: 49th International Colloquium on Automata, Languages and Programming. Leibniz International Proceedings in Informatics (LIPIcs), vol. 229, pp. 28:1–28:20. Schloss Dagstuhl (Jul 2022). https://doi.org/10.4230/LIPIcs.ICALP.2022.28
27. Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Candidate iO from homomorphic encryption schemes. Journal of Cryptology **36**(3), 27 (Jul 2023). https://doi.org/10.1007/s00145-023-09471-5
28. Brakerski, Z., Tsabary, R., Vaikuntanathan, V., Wee, H.: Private constrained PRFs (and more) from LWE. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017: 15th Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 10677, pp. 264–302. Springer, Cham (Nov 2017). https://doi.org/10.1007/978-3-319-70500-2_10
29. Branco, P., Döttling, N., Jain, A., Malavolta, G., Mathialagan, S., Peters, S., Vaikuntanathan, V.: Pseudorandom obfuscation and applications. In: Kalai, Y.T., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025, Part V. Lecture Notes in Computer Science, vol. 16004, pp. 663–698. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01901-1_21
30. Brzuska, C., Ünal, A., Woo, I.K.Y.: Evasive LWE assumptions: Definitions, classes, and counterexamples. In: Chung, K.M., Sasaki, Y. (eds.) Advances in Cryptology – ASIACRYPT 2024, Part IV. Lecture Notes in Computer Science, vol. 15487, pp. 418–449. Springer, Singapore (Dec 2024). https://doi.org/10.1007/978-981-96-0894-2_14

31. Chen, C., Genise, N., Micciancio, D., Polyakov, Y., Rohloff, K.: Implementing token-based obfuscation under (ring) LWE. Cryptology ePrint Archive, Report 2018/1222 (2018), https://eprint.iacr.org/2018/1222
32. Chen, Y., Gentry, C., Halevi, S.: Cryptanalyses of candidate branching program obfuscators. In: Coron, J.S., Nielsen, J.B. (eds.) Advances in Cryptology – EUROCRYPT 2017, Part III. Lecture Notes in Computer Science, vol. 10212, pp. 278–307. Springer, Cham (Apr / May 2017). https://doi.org/10.1007/978-3-319-56617-7_10
33. Chen, Y., Hhan, M., Vaikuntanathan, V., Wee, H.: Matrix PRFs: Constructions, attacks, and applications to obfuscation. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019: 17th Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 11891, pp. 55–80. Springer, Cham (Dec 2019). https://doi.org/10.1007/978-3-030-36030-6_3
34. Chen, Y., Vaikuntanathan, V., Wee, H.: GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, Part II. Lecture Notes in Computer Science, vol. 10992, pp. 577–607. Springer, Cham (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0_20
35. Cheon, J.H., Cho, W., Hhan, M., Kim, J., Lee, C.: Statistical zeroizing attack: Cryptanalysis of candidates of BP obfuscation over GGH15 multilinear map. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part III. Lecture Notes in Computer Science, vol. 11694, pp. 253–283. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_9
36. Cheon, J.H., Fouque, P.A., Lee, C., Minaud, B., Ryu, H.: Cryptanalysis of the new CLT multilinear map over the integers. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part I. Lecture Notes in Computer Science, vol. 9665, pp. 509–536. Springer, Berlin, Heidelberg (May 2016). https://doi.org/10.1007/978-3-662-49890-3_20
37. Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the multilinear map over the integers. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015, Part I. Lecture Notes in Computer Science, vol. 9056, pp. 3–12. Springer, Berlin, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46800-5_1
38. Coron, J.S., Gentry, C., Halevi, S., Lepoint, T., Maji, H.K., Miles, E., Raykova, M., Sahai, A., Tibouchi, M.: Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part I. Lecture Notes in Computer Science, vol. 9215, pp. 247–266. Springer, Berlin, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-47989-6_12
39. Coron, J.S., Lee, M.S., Lepoint, T., Tibouchi, M.: Cryptanalysis of GGH15 multilinear maps. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part II. Lecture Notes in Computer Science, vol. 9815, pp. 607–628. Springer, Berlin, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53008-5_21
40. Coron, J.S., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 476–493. Springer, Berlin, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_26
41. Cousins, D.B., Di Crescenzo, G., Gür, K.D., King, K., Polyakov, Y., Rohloff, K., Ryan, G.W., Savas, E.: Implementing conjunction obfuscation under entropic ring

LWE. In: 2018 IEEE Symposium on Security and Privacy. pp. 354–371. IEEE Computer Society Press (May 2018). https://doi.org/10.1109/SP.2018.00007
42. Curtis, B.R., Player, R.: On the feasibility and impact of standardising sparse-secret lwe parameter sets for homomorphic encryption. In: Proceedings of the 7th ACM Workshop on Encrypted Computing & Applied Homomorphic Cryptography. pp. 1–10. WAHC'19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3338469.3358940, https://doi.org/10.1145/3338469.3358940
43. Devadas, L., Quach, W., Vaikuntanathan, V., Wee, H., Wichs, D.: Succinct LWE sampling, random polynomials, and obfuscation. In: Nissim, K., Waters, B. (eds.) TCC 2021: 19th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 13043, pp. 256–287. Springer, Cham (Nov 2021). https://doi.org/10.1007/978-3-030-90453-1_9
44. Döttling, N., Jain, A., Malavolta, G., Mathialagan, S., Vaikuntanathan, V.: Simple and general counterexamples for private-coin evasive LWE. In: Kalai, Y.T., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025, Part VII. Lecture Notes in Computer Science, vol. 16006, pp. 73–92. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01907-3_3
45. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th Annual Symposium on Foundations of Computer Science. pp. 40–49. IEEE Computer Society Press (Oct 2013). https://doi.org/10.1109/FOCS.2013.13
46. Gay, R., Pass, R.: Indistinguishability obfuscation from circular security. In: Khuller, S., Williams, V.V. (eds.) 53rd Annual ACM Symposium on Theory of Computing. pp. 736–749. ACM Press (Jun 2021). https://doi.org/10.1145/3406325.3451070
47. Genise, N., Micciancio, D.: Faster Gaussian sampling for trapdoor lattices with arbitrary modulus. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part I. Lecture Notes in Computer Science, vol. 10820, pp. 174–203. Springer, Cham (Apr / May 2018). https://doi.org/10.1007/978-3-319-78381-9_7
48. Gentry, C., Gorbunov, S., Halevi, S.: Graph-induced multilinear maps from lattices. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015: 12th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 9015, pp. 498–527. Springer, Berlin, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-46497-7_20
49. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th Annual ACM Symposium on Theory of Computing. pp. 197–206. ACM Press (May 2008). https://doi.org/10.1145/1374376.1374407
50. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 75–92. Springer, Berlin, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_5
51. Goldreich, O.: Candidate one-way functions based on expander graphs. Electronic Colloquium on Computational Complexity (ECCC), Report TR00-090 (2000), https://eccc.weizmann.ac.il/eccc-reports/2000/TR00-090/index.html
52. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: 25th Annual Symposium on Foundations of Computer Science. pp. 464–479. IEEE Computer Society Press (Oct 1984). https://doi.org/10.1109/SFCS.1984.715949

53. Goldwasser, S., Gordon, S.D., Goyal, V., Jain, A., Katz, J., Liu, F.H., Sahai, A., Shi, E., Zhou, H.S.: Multi-input functional encryption. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology – EUROCRYPT 2014. Lecture Notes in Computer Science, vol. 8441, pp. 578–602. Springer, Berlin, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_32
54. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Predicate encryption for circuits from LWE. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part II. Lecture Notes in Computer Science, vol. 9216, pp. 503–523. Springer, Berlin, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_25
55. Halevi, S., Halevi, T., Shoup, V., Stephens-Davidowitz, N.: Implementing BP-obfuscation using graph-induced encoding. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 2017: 24th Conference on Computer and Communications Security. pp. 783–798. ACM Press (Oct / Nov 2017). https://doi.org/10.1145/3133956.3133976
56. Halevi, S., Polyakov, Y., Shoup, V.: An improved RNS variant of the BFV homomorphic encryption scheme. In: Matsui, M. (ed.) Topics in Cryptology – CT-RSA 2019. Lecture Notes in Computer Science, vol. 11405, pp. 83–105. Springer, Cham (Mar 2019). https://doi.org/10.1007/978-3-030-12612-4_5
57. Hopkins, S.B., Jain, A., Lin, H.: Counterexamples to new circular security assumptions underlying iO. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021, Part II. Lecture Notes in Computer Science, vol. 12826, pp. 673–700. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84245-1_23
58. Hsieh, Y.C., Jain, A., Lin, H.: Lattice-based post-quantum iO from circular security with random opening assumption. In: Kalai, Y.T., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025, Part VII. Lecture Notes in Computer Science, vol. 16006, pp. 3–38. Springer, Cham (Aug 2025). https://doi.org/10.1007/978-3-032-01907-3_1
59. Hsieh, Y.C., Jain, A., Lin, H.: Lattice-based post-quantum iO from circular security with random opening assumption (part II: zeroizing attacks against private-coin evasive LWE assumptions). Cryptology ePrint Archive, Report 2025/390 (2025), https://eprint.iacr.org/2025/390
60. Hsieh, Y.C., Lin, H., Luo, J.: Attribute-based encryption for circuits of unbounded depth from lattices. In: 64th Annual Symposium on Foundations of Computer Science. pp. 415–434. IEEE Computer Society Press (Nov 2023). https://doi.org/10.1109/FOCS57990.2023.00031
61. Hsieh, Y.C., Lin, H., Luo, J.: Attribute-based encryption for circuits of unbounded depth from lattices: Garbled circuits of optimal size, laconic functional evaluation, and more. Cryptology ePrint Archive, Report 2023/1716 (2023), https://eprint.iacr.org/2023/1716
62. Huang, T.H., Hung, W.H., Yamada, S.: A note on obfuscation-based attacks on private-coin evasive LWE. IACR Communications in Cryptology (CiC) **2**(4), 20 (2026). https://doi.org/10.62056/a39qudy6b
63. Jain, A., Lin, H., Lou, P., Sahai, A.: Polynomial-time cryptanalysis of the subspace flooding assumption for post-quantum $i\mathcal{O}$. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part I. Lecture Notes in Computer Science, vol. 14004, pp. 205–235. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30545-0_8

64. Jain, A., Lin, H., Luo, J., Wichs, D.: The pseudorandom oracle model and ideal obfuscation. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, Part IV. Lecture Notes in Computer Science, vol. 14084, pp. 233–262. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38551-3_8
65. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: Khuller, S., Williams, V.V. (eds.) 53rd Annual ACM Symposium on Theory of Computing. pp. 60–73. ACM Press (Jun 2021). https://doi.org/10.1145/3406325.3451093
66. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology – EUROCRYPT 2022, Part I. Lecture Notes in Computer Science, vol. 13275, pp. 670–699. Springer, Cham (May / Jun 2022). https://doi.org/10.1007/978-3-031-06944-4_23
67. Lewi, K., Malozemoff, A.J., Apon, D., Carmer, B., Foltzer, A., Wagner, D., Archer, D.W., Boneh, D., Katz, J., Raykova, M.: 5Gen: A framework for prototyping applications using multilinear maps and matrix branching programs. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) ACM CCS 2016: 23rd Conference on Computer and Communications Security. pp. 981–992. ACM Press (Oct 2016). https://doi.org/10.1145/2976749.2978314
68. Lin, H., Matt, C.: Pseudo flawed-smudging generators and their application to indistinguishability obfuscation. Cryptology ePrint Archive, Report 2018/646 (2018), `https://eprint.iacr.org/2018/646`
69. Maples, K.: Singularity of random matrices over finite fields. arXiv preprint arXiv:1012.2372 (2010)
70. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012. Lecture Notes in Computer Science, vol. 7237, pp. 700–718. Springer, Berlin, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_41
71. Naor, M., Reingold, O.: Synthesizers and their application to the parallel construction of pseudo-random functions. In: 36th Annual Symposium on Foundations of Computer Science. pp. 170–181. IEEE Computer Society Press (Oct 1995). https://doi.org/10.1109/SFCS.1995.492474
72. Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation from semantically-secure multilinear encodings. In: Garay, J.A., Gennaro, R. (eds.) Advances in Cryptology – CRYPTO 2014, Part I. Lecture Notes in Computer Science, vol. 8616, pp. 500–517. Springer, Berlin, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_28
73. Ragavan, S., Vafa, N., Vaikuntanathan, V.: Indistinguishability obfuscation from bilinear maps and LPN variants. In: Boyle, E., Mahmoody, M. (eds.) TCC 2024: 22nd Theory of Cryptography Conference, Part IV. Lecture Notes in Computer Science, vol. 15367, pp. 3–36. Springer, Cham (Dec 2024). https://doi.org/10.1007/978-3-031-78023-3_1
74. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th Annual ACM Symposium on Theory of Computing. pp. 84–93. ACM Press (May 2005). https://doi.org/10.1145/1060590.1060603
75. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th Annual ACM Symposium on Theory of Computing. pp. 475–484. ACM Press (May / Jun 2014). https://doi.org/10.1145/2591796.2591825

76. Suegami, S., Bottazzi, E.: Lookup-table evaluation over key-homomorphic encodings and KP-ABE for nonlinear operations. Cryptology ePrint Archive, Report 2025/1870 (2025), https://eprint.iacr.org/2025/1870
77. Vaikuntanathan, V., Wee, H., Wichs, D.: Witness encryption and null-IO from evasive LWE. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022, Part I. Lecture Notes in Computer Science, vol. 13791, pp. 195–221. Springer, Cham (Dec 2022). https://doi.org/10.1007/978-3-031-22963-3_7
78. Waters, B., Wee, H., Wu, D.J.: Multi-authority ABE from lattices without random oracles. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022: 20th Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 13747, pp. 651–679. Springer, Cham (Nov 2022). https://doi.org/10.1007/978-3-031-22318-1_23
79. Wee, H.: Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology – EUROCRYPT 2022, Part II. Lecture Notes in Computer Science, vol. 13276, pp. 217–241. Springer, Cham (May / Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_8
80. Wee, H., Wichs, D.: Candidate obfuscation via oblivious LWE sampling. In: Canteaut, A., Standaert, F.X. (eds.) Advances in Cryptology – EUROCRYPT 2021, Part III. Lecture Notes in Computer Science, vol. 12698, pp. 127–156. Springer, Cham (Oct 2021). https://doi.org/10.1007/978-3-030-77883-5_5

A Asymptotic Analysis

This appendix analyzes the asymptotic cost shared by the final decryption procedures of our PFFE scheme and the FE scheme of Agrawal et al. [1], and then uses this common estimate to compare the prior FE-to-iO transformation for pseudorandom functionalities [2] with the Diamond iO construction in Subsection 3.2. This asymptotic efficiency analysis does not take into account the implementation optimizations described from Section 5 onward; it computes the asymptotic efficiency only from the minimal assumptions needed for the concrete constructions. The PFFE decryption algorithm differs from the FE decryption algorithm of Agrawal et al. only in a preprocessing step: it first multiplies by the preimages $\mathbf{K}_{\mathsf{att}}$ and $\mathbf{K}_t$ to obtain the input BGG+ encodings. We ignore this preprocessing cost because it is lower-order compared with the subsequent evaluation over BGG+ encodings, which is the common bottleneck analyzed below.

Let L denote the circuit input size, and let $k := \lceil \log_2 q \rceil$ be the bit length of the modulus q. The modulus q is represented as a product of pairwise coprime fixed-bit-width integers, so one multiplication over $\mathbb{Z}_q$ costs $\Theta(k)$ modular multiplications over fixed-width CRT-factor moduli. Throughout this analysis we count such fixed-width modular multiplications and ignore lower-order operations such as additions.

We assume that the Gaussian parameter σ used for sampling fresh errors is fixed. For a fixed security level, the lattice dimension n grows proportionally to k [42], and hence

$$n = \Theta(k).$$

All asymptotic expressions below may equivalently substitute $k = \lceil \log_2 q \rceil$.

Cost of the common FE/PFFE decryption core. A single BGG+ encoding is represented as a row vector over $\mathbb{Z}_q$ with $\Theta(nk)$ columns. Multiplying two BGG+ encodings is represented as a vector-matrix multiplication between a row vector of dimension $1 \times \Theta(nk)$ and a matrix of dimension $\Theta(nk) \times \Theta(nk)$. Therefore, the number of scalar multiplications over $\mathbb{Z}_q$ is $\Theta(nk) \cdot \Theta(nk) = \Theta(n^2k^2)$. Since each multiplication over $\mathbb{Z}_q$ costs $\Theta(k)$ fixed-width modular multiplications, the cost of one BGG+ encoding multiplication is

$$C_{\mathsf{enc\text{-}mult}} = \Theta(n^2k^2) \cdot \Theta(k) = \Theta(n^2k^3).$$

Using $n = \Theta(k)$, this becomes

$$C_{\mathsf{enc\text{-}mult}} = \Theta(k^5).$$

A single BGG+ encoding can encode only one small integer, such as one bit, because the noise growth per multiplication depends on the size of the encoded value [16]. Instead of simulating one multiplication modulo q directly, the circuit can represent q by $h = \Theta(k)$ fixed-bit-width pairwise coprime RNS moduli and simulate the corresponding modular multiplication for each RNS modulus in a separate parallel subcircuit. Then, computing the encoded values of all RNS slots of the product modulo q requires $\Theta(k)$ multiplications between BGG+ encodings. The cost of simulating one multiplication over $\mathbb{Z}_q$ on encodings is therefore

$$C_{\mathbb{Z}_q\text{-}\mathsf{mult\text{-}sim}} = \Theta(k) \cdot \Theta(n^2k^3) = \Theta(n^2k^4).$$

Using $n = \Theta(k)$, this becomes

$$C_{\mathbb{Z}_q\text{-}\mathsf{mult\text{-}sim}} = \Theta(k^6).$$

We assume that the FHE scheme used in this common decryption core is instantiated by a ring variant of the GSW FHE scheme [50]. Consider polynomial multiplication in a polynomial ring of dimension n and modulus q. Since one encoding can hold only one coefficient, simulating one such polynomial multiplication over BGG+ encodings requires $\Theta(n^2)$ simulated $\mathbb{Z}_q$ multiplications over encodings.[22] Therefore,

$$C_{\mathsf{poly\text{-}mult\text{-}sim}} = \Theta(n^2) \cdot \Theta(n^2k^4) = \Theta(n^4k^4).$$

Using $n = \Theta(k)$, this becomes

$$C_{\mathsf{poly\text{-}mult\text{-}sim}} = \Theta(k^8).$$

[22] Although polynomial multiplication can be implemented using the number-theoretic transform (NTT) with only $\Theta(n \log_2 n)$ simulated $\mathbb{Z}_q$ multiplications, it requires $\Theta(\log_2 n)$ sequential butterfly stages. Since our goal is to minimize the evaluation depth—and hence the noise growth and the size of k—we instantiate polynomial multiplication as vector-matrix multiplication in coefficient representation.

Assume that the 1-bit PRF is instantiated by a circuit whose number of multiplication gates is linear in the input size. For input size i, homomorphically evaluating a PRF with one output bit over encodings requires $\Theta(i)$ simulated polynomial multiplications over encodings. Therefore,

$$C_{\mathsf{1bit\text{-}PRF}}(i) = \Theta(i) \cdot \Theta(n^4 k^4) = \Theta(i n^4 k^4).$$

Using $n = \Theta(k)$, this becomes

$$C_{\mathsf{1bit\text{-}PRF}}(i) = \Theta(i k^4 k^4) = \Theta(i k^8).$$

We fix the circuit evaluated during decryption to a simple pseudorandom function, for example, a circuit that computes the PRF on the given inputs. For input size i and one output bit, the common decryption core requires $\Theta(k)$ evaluations of the 1-bit PRF over encodings. Thus,

$$C^{\mathsf{1bit}}_{\mathsf{FE.Dec}}(i) = \Theta(k) \cdot \Theta(i n^4 k^4) = \Theta(i n^4 k^5).$$

Using $n = \Theta(k)$, we get

$$C^{\mathsf{1bit}}_{\mathsf{FE.Dec}}(i) = \Theta(i k^4 k^5) = \Theta(i k^9).$$

For circuit input size $i = L$, the cost of the common decryption core is

$$C^{\mathsf{1bit}}_{\mathsf{FE.Dec}}(L) = \Theta(L n^4 k^5),$$

and, using $n = \Theta(k)$,

$$C^{\mathsf{1bit}}_{\mathsf{FE.Dec}}(L) = \Theta(L k^9).$$

Cost of the prior FE-to-iO transformation. Under the BGG+ encoding-based representation, an FE ciphertext consists of $\Theta(nk^2)$ bits per plaintext bit. Thus, if the plaintext has t bits, the corresponding FE ciphertext size is $\Theta(tnk^2)$ bits. For the lower-bound analysis below, it is enough to use $\Omega(tnk^2)$.

In the i-th round of the prior FE-to-iO transformation, the required FE decryption has input size $\Theta(L - i)$ and output size at least $\Omega((L - i)nk^2)$. The 1-bit FE decryption cost for input size $L - i$ is $\Theta((L - i)n^4 k^5)$. Therefore, the total FE decryption cost in the i-th round is at least

$$C_i = \Omega((L - i)nk^2) \cdot \Theta((L - i)n^4 k^5) = \Omega((L - i)^2 n^5 k^7).$$

The transformation repeats the above computation from the $(L-1)$-th round down to the first round. Hence, setting $j = L - i$, the total cost is at least

$$\begin{aligned} C_{\mathsf{prior\text{-}FE\text{-}to\text{-}iO}} &= \sum_{i=1}^{L-1} \Omega((L - i)^2 n^5 k^7) \\ &= \Omega(n^5 k^7) \sum_{j=1}^{L-1} j^2 \\ &= \Omega(n^5 k^7 L^3). \end{aligned}$$

Under the assumption $n = \Theta(k)$, we have

$$C_{\mathsf{prior\text{-}FE\text{-}to\text{-}iO}} = \Omega(k^{12}L^3).$$

If $k = \Theta(L)$, then

$$C_{\mathsf{prior\text{-}FE\text{-}to\text{-}iO}} = \Omega(k^{15}) = \Omega(L^{15}).$$

For an obfuscated circuit with L input bits and one output bit, define the overhead ratio of the prior FE-to-iO transformation as

$$r_{\mathsf{prior}} := \frac{C_{\mathsf{prior\text{-}FE\text{-}to\text{-}iO}}}{C^{\mathsf{1bit}}_{\mathsf{FE.Dec}}(L)}.$$

From the lower-bound analysis above, $C_{\mathsf{prior\text{-}FE\text{-}to\text{-}iO}} = \Omega(n^5k^7L^3)$. From the common decryption-core analysis, the final 1-bit decryption cost is $C^{\mathsf{1bit}}_{\mathsf{FE.Dec}}(L) = \Theta(Ln^4k^5)$ and is therefore upper bounded by $O(Ln^4k^5)$. Therefore,

$$r_{\mathsf{prior}} = \Omega\left(\frac{n^5k^7L^3}{Ln^4k^5}\right) = \Omega(nk^2L^2).$$

Using $n = \Theta(k)$, this becomes

$$r_{\mathsf{prior}} = \Omega(k^3L^2).$$

If $k = \Theta(L)$, then

$$r_{\mathsf{prior}} = \Omega(L^5).$$

Cost of the Diamond FE-to-iO transformation. In the Diamond FE-to-iO transformation, each round multiplies a row vector over $\mathbb{Z}_q$ of size $\Theta(Lnk)$ by a matrix over $\mathbb{Z}_q$ whose number of rows and columns are both $\Theta(Lnk)$. In the i-th round, the required vector-matrix multiplication has dimensions $1 \times \Theta(Lnk)$ and $\Theta(Lnk) \times \Theta(Lnk)$. Therefore, the number of scalar multiplications over $\mathbb{Z}_q$ is

$$\Theta(Lnk) \cdot \Theta(Lnk) = \Theta(L^2n^2k^2).$$

Since one multiplication over $\mathbb{Z}_q$ costs $\Theta(k)$ modular multiplications over fixed-width factor moduli, the cost of the i-th round is

$$C_i^{\mathsf{Diamond}} = \Theta(L^2n^2k^2) \cdot \Theta(k) = \Theta(L^2n^2k^3).$$

The Diamond FE-to-iO transformation repeats the above computation for L rounds. Hence, the total cost is

$$C_{\mathsf{Diamond\text{-}FE\text{-}to\text{-}iO}} = \sum_{i=1}^{L} \Theta(L^2n^2k^3) = \Theta(L^3n^2k^3).$$

Under the assumption $n = \Theta(k)$, we obtain

$$C_{\mathsf{Diamond\text{-}FE\text{-}to\text{-}iO}} = \Theta(L^3 k^5).$$

Because $k = \Theta(L)$ in Diamond iO, this becomes

$$C_{\mathsf{Diamond\text{-}FE\text{-}to\text{-}iO}} = \Theta(L^8).$$

The ratio of the Diamond FE-to-iO transformation cost to the final PFFE decryption cost, represented by the same common-core quantity, is

$$\begin{aligned} r_{\mathsf{Diamond}} &:= \frac{C_{\mathsf{Diamond\text{-}FE\text{-}to\text{-}iO}}}{C^{\mathsf{1bit}}_{\mathsf{FE.Dec}}(L)} \\ &= \Theta\left(\frac{L^3 k^5}{L k^9}\right) = \Theta\left(\frac{L^2}{k^4}\right). \end{aligned}$$

Because $k = \Theta(L)$ in Diamond iO, it follows that

$$r_{\mathsf{Diamond}} = \Theta\left(\frac{1}{L^2}\right).$$

These derivations give the asymptotic costs and overhead ratios reported in Table 1.

B Missing Proofs in Section 2

B.1 Missing Properties for Fully Homomorphic Encryption

We show that a variant of the Gentry, Sahai, and Waters (GSW) leveled FHE scheme [50] satisfies the claimed properties. Let σ_E be a Gaussian parameter such that the LWE assumption with the parameters (n, m, q, σ_E) is subexponentially hard. We adopt the syntax and properties from [61] with necessary modifications.

- $\mathsf{KeyGen}(1^\lambda) \to (\mathbf{t}, \mathbf{A}_{\mathsf{fhe}})$:

$$\begin{aligned} &\text{(secret)} \quad \mathbf{t} := \left(\bar{\mathbf{t}}, -1\right) \in \mathbb{Z}_q^{n+1}, \\ &\text{(public)} \quad \mathbf{A}_{\mathsf{fhe}} := \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \bar{\mathbf{t}}\bar{\mathbf{A}}_{\mathsf{fhe}} + \mathbf{e}_{\mathsf{fhe}} \end{pmatrix} \in \mathbb{Z}_q^{(n+1)\times m}, \end{aligned}$$

  where $\bar{\mathbf{t}} \leftarrow\!\$\, \{0,1\}^n$, $\bar{\mathbf{A}}_{\mathsf{fhe}} \leftarrow\!\$\, \mathbb{Z}_q^{n\times m}$, $\bar{\mathbf{e}}_{\mathsf{fhe}} \leftarrow\!\$\, \mathcal{D}^m_{\mathbb{Z},\sigma_E}$.
- $\mathsf{Enc}(\mathbf{A}_{\mathsf{fhe}}, \mathbf{z}) \to \mathbf{Z}$:

$$\mathbf{Z} := \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathbf{z} \otimes \mathbf{G}_{n+1} \in \mathbb{Z}_q^{(n+1)\times \ell m},$$

  where $\mathbf{R} \leftarrow\!\$\, \{0,1\}^{m\times \ell m}$ is the encryption randomness.
- $\mathsf{Eval}(\mathbf{A}_{\mathsf{fhe}}, C, \mathbf{Z}) \to \mathbf{Y}$ where C is a vector-valued circuit $C : \{0,1\}^L \to \mathbb{Z}_q^{1\times M}$.

We now prove the claimed properties.

Subexponential semantic security. We show that the public key $\mathbf{A}_{\mathsf{fhe}}$ and the ciphertext $\mathbf{X}$ are indistinguishable from random matrices, respectively. Specifically, it holds that

$$\begin{aligned}
&\left(\mathbf{A} := \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \bar{\mathbf{b}} := \bar{\mathbf{t}}\bar{\mathbf{A}}_{\mathsf{fhe}} + \mathbf{e}_{\mathsf{fhe}} \end{pmatrix}, \mathbf{Z} := \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathbf{z} \otimes \mathbf{G}_{n+1}\right) \\
\overset{\mathrm{c}}{\approx} &\left(\mathbf{A} := \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \bar{\mathbf{b}} \leftarrow\!\$\, \mathbb{Z}_q^m \end{pmatrix}, \mathbf{Z} := \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathbf{z} \otimes \mathbf{G}_{n+1}\right) \\
\overset{\mathrm{s}}{\approx} &\left(\mathbf{A}_{\mathsf{fhe}} := \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \bar{\mathbf{b}} \leftarrow\!\$\, \mathbb{Z}_q^m \end{pmatrix}, \mathbf{Z}, \leftarrow\!\$\, \mathbb{Z}_q^{(n+1)\times \ell m}\right).
\end{aligned}$$

Here, the computational indistinguishability follows from the subexponential hardness of the LWE assumption (Assumption 1). The statistical indistinguishability is implied by the leftover hash lemma (Lemma 1) since $\ell m \geq 2n\lceil \log_2 q \rceil$. Hence, this completes the proof of the claimed security.

Correctness. For an input ciphertext $\mathbf{X} := \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathbf{x} \otimes \mathbf{G}$ and a vector-valued circuit $C : \{0,1\}^\ell \to \mathbb{Z}_q^{1\times M}$, Lemma 5 in [61] shows that there exists a matrix $\mathbf{R}_C$ such that the following holds:

$$\mathbf{Y} \;:=\; \mathbf{A}_{\mathsf{fhe}}\mathbf{R}_C \;-\; \begin{pmatrix} \mathbf{0}_{n\times M} \\ C(\mathbf{x}) \end{pmatrix} \;\in\; \mathbb{Z}_q^{(n+1)\times M},$$

where $\|\mathbf{R}_C\|_\infty \leq (m+2)^{\mathsf{dep}}\lceil \log_2 q \rceil \max_{\ell\in[L]} \|\mathbf{R}_\ell\|_\infty$. Therefore, the decryption equation is

$$\mathbf{t}\mathbf{Y} \;=\; -\mathbf{e}_{\mathsf{fhe}}\mathbf{R}_C + C(\mathbf{x}).$$

Recall that, by Lemma 2, $\|\mathbf{e}_{\mathsf{fhe}}\|_\infty \leq \sqrt{\lambda}\sigma_E$ with overwhelming probability. Additionally, we have $\|\mathbf{R}_i\|_\infty \leq 1$ for every $i \in [L]$ since $\mathbf{R}_\ell$ is sampled from $\{0,1\}^{m\times Lm}$. Substituting these bounds we obtain:

$$\begin{aligned}
\|\mathbf{e}\|_\infty &\leq \sqrt{\lambda}\sigma_E m(m+2)^{\mathsf{dep}}\lceil \log_2 q \rceil \\
&\leq \mathsf{poly}(\lambda)\cdot m^{\mathcal{O}(\mathsf{dep})}
\end{aligned}$$

which satisfies the claimed bound.

Shallowness. Recall that $C[\mathbf{x}] : \{0,1\}^\ell \to \mathbb{Z}_q^{1\times M}$ is a circuit of depth at most dep in which $\mathbf{x} \in \{0,1\}^L$ is hardwired. For the proof, we consider the associated circuit $C' : \{0,1\}^{\ell+L} \to \mathbb{Z}_q^{1\times M}$ obtained from $C[\mathbf{x}]$ by replacing the hardwired bits $x_1, \ldots, x_L$ with fresh input wires $u_1, \ldots, u_L$. Thus, for every $\mathbf{z} \in \{0,1\}^\ell$,

$$C'(\mathbf{z}, \mathbf{x}) = C[\mathbf{x}](\mathbf{z}).$$

By Lemma 5 in [61], there exists an efficient algorithm

$$\mathsf{MakeVEvalCkt}(n, m, q, C') \to G'_C$$

that outputs a circuit G'_C for homomorphically evaluating C'. Namely, G'_C takes as input $\ell + L$ ciphertexts and outputs a ciphertext of

$$C'(\mathbf{z}, \mathbf{u}).$$

Moreover, the depth of G'_C is bounded by

$$\mathsf{dep}(G'_C) \leq \mathsf{dep} \cdot \mathcal{O}(\log m \log\log q) + \mathcal{O}(\log^2 \log q).$$

We construct $\mathsf{MakeVEvalCkt}$ for the hardwired circuit $C[\mathbf{x}]$ by outputting a circuit G_C that first converts each hardwired bit x_i into the corresponding dummy ciphertext

$$\mathbf{D}_{i,x_i} := -x_i \cdot \mathbf{G}_{n+1} \in \mathbb{Z}_q^{(n+1)\times m},$$

and then invokes G'_C on the ℓ input ciphertexts together with these L dummy ciphertexts. Consequently, the output of G_C is a ciphertext of

$$C'(\mathbf{z}, \mathbf{x}) = C[\mathbf{x}](\mathbf{z}).$$

The conversion of the hardwired bits into dummy ciphertexts consists only of multiplications by hardcoded constants and adds one layer. Hence,

$$\mathsf{dep}(G_C) \leq \mathsf{dep}(G'_C) + 1 \leq \mathsf{dep} \cdot \mathsf{poly}(\log m, \log\log q).$$

□

B.2 Missing Proof of Lemma 7

The j-th output entry of $\mathbf{tY}$ depends only on the j-th column of $\mathbf{Y}$. Thus the M output positions can be handled independently, and the final matrices $\mathbf{A}_G$ and $\mathbf{H}_{G,x}$ are obtained by concatenating the matrices constructed for each output position. It is therefore enough to describe the construction for one output position, i.e., for the case $M = 1$.

Let $G : \{0,1\}^L \to \mathbb{Z}_q^{t\times 1}$ and write

$$\mathbf{Y} = G(\mathbf{x}) = \begin{pmatrix} Y_1 \\ \vdots \\ Y_t \end{pmatrix}.$$

Let G' be the Boolean circuit whose output is the bit representation of $\mathbf{Y}$. For each $i \in [t]$ and $k \in [\lceil \log_2 q \rceil]$, let $y_{i,k} \in \{0,1\}$ denote the k-th bit of Y_i, so that

$$Y_i = \sum_{k\in[\lceil \log_2 q\rceil]} 2^{k-1} y_{i,k}.$$

The output bits of G' are ordered as

$$G'(\mathbf{x})_{(k-1)t+i} = y_{i,k}.$$

Let

$$\mathbf{A}_{G'} \leftarrow \mathsf{EvalC}(\mathbf{A}, G'), \qquad \mathbf{H}_{G',x} \leftarrow \mathsf{EvalCX}(\mathbf{A}, G', \mathbf{x}).$$

Then

$$(\mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G})\, \mathbf{H}_{G',x} = \mathbf{A}_{G'} - \mathsf{bits}(\mathbf{Y}) \otimes \mathbf{G}.$$

Let $\mathbf{A}_{G',y_{i,k}} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{H}_{G',x,y_{i,k}} \in \mathbb{Z}_q^{(L+1)m \times m}$ denote the blocks corresponding to the bit $y_{i,k}$. Also, let $\mathbf{H}_{t,i} \in \mathbb{Z}_q^{tm \times m}$ be the block-selection matrix whose i-th $m \times m$ block is $\mathbf{I}_m$ and whose other blocks are zero.

For fixed $i \in [t]$ and $k \in [\lceil \log_2 q \rceil]$, define

$$\mathbf{K}_{i,k} := \begin{pmatrix} \mathbf{H}_{G',x,y_{i,k}} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_{t,i} \end{pmatrix} \begin{pmatrix} \mathbf{G}_n^{-1}(\mathbf{A}_t \mathbf{H}_{t,i}) \\ y_{i,k} \mathbf{I}_m \end{pmatrix} \mathbf{G}_n^{-1}(2^{k-1}\mathbf{G}).$$

Then

$$\begin{aligned}
&(\mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G},\ \mathbf{A}_t - \mathbf{t} \otimes \mathbf{G})\, \mathbf{K}_{i,k} \\
&= \left(\mathbf{A}_{G',y_{i,k}} - y_{i,k}\mathbf{G},\ \mathbf{A}_t \mathbf{H}_{t,i} - t_i \mathbf{G}\right) \begin{pmatrix} \mathbf{G}_n^{-1}(\mathbf{A}_t \mathbf{H}_{t,i}) \\ y_{i,k} \mathbf{I}_m \end{pmatrix} \mathbf{G}_n^{-1}(2^{k-1}\mathbf{G}) \\
&= \mathbf{A}_{G',y_{i,k}} \mathbf{G}_n^{-1}(\mathbf{A}_t \mathbf{H}_{t,i}) \mathbf{G}_n^{-1}(2^{k-1}\mathbf{G}) - 2^{k-1} t_i y_{i,k} \mathbf{G}.
\end{aligned}$$

Therefore, define

$$\mathbf{H}_{G,x} := \sum_{k \in [\lceil \log_2 q \rceil]} \sum_{i \in [t]} \mathbf{K}_{i,k},$$

and

$$\mathbf{A}_G := \sum_{k \in [\lceil \log_2 q \rceil]} \sum_{i \in [t]} \mathbf{A}_{G',y_{i,k}} \mathbf{G}_n^{-1}(\mathbf{A}_t \mathbf{H}_{t,i}) \mathbf{G}_n^{-1}(2^{k-1}\mathbf{G}).$$

Since

$$\mathbf{t}\mathbf{Y} = \sum_{k \in [\lceil \log_2 q \rceil]} \sum_{i \in [t]} 2^{k-1} t_i y_{i,k},$$

we obtain

$$(\mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G},\ \mathbf{A}_t - \mathbf{t} \otimes \mathbf{G})\, \mathbf{H}_{G,x} = \mathbf{A}_G - (\mathbf{t}\mathbf{Y}) \otimes \mathbf{G}.$$

It remains to bound the norm. Recall that

$$\left\|\mathbf{H}_{G',x}\right\|_\infty \le m^{\mathcal{O}(\mathsf{dep})}.$$

We have

$$\begin{aligned}
\left\|\mathbf{K}_{i,k}\right\|_\infty &\le m \left\| \begin{pmatrix} \mathbf{H}_{G',x,y_{i,k}} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_{t,i} \end{pmatrix} \begin{pmatrix} \mathbf{G}^{-1}(\mathbf{A}_t \mathbf{H}_{t,i}) \\ y_{i,k} \mathbf{I}_m \end{pmatrix} \right\|_\infty \\
&\le m^2 \cdot \max \left\|\mathbf{H}_{G',x,y_{i,k}}\right\|_\infty \mathbf{G}^{-1}(\mathbf{A}_t \mathbf{H}_{t,i}), y_{i,k} \left\|\mathbf{H}_{t,i}\right\|_\infty \\
&\le m^{\mathcal{O}(\mathsf{dep})}.
\end{aligned}$$

Therefore, it holds that

$$\begin{aligned}\left\|\mathbf{H}_{G,x}\right\|_\infty &\le \sum_{k\in[\lceil \log_2 q\rceil]}\sum_{i\in[t]}\left\|\mathbf{K}_{i,k}\right\|_\infty \\ &\le t\lceil \log_2 q\rceil m^{\mathcal{O}(\mathsf{dep})}.\end{aligned}$$

C Missing Proofs in Section 3

C.1 Missing Proof of Lemma 8

The $\mathsf{iPRO.Eval}$ algorithm simply forwards the output of the $\mathsf{PFFE.Dec}$ algorithm. Therefore, it suffices to show that the ciphertext $(\mathsf{ct}_{\mathbf{z}}, \widetilde{\mathsf{ct}}_{\mathbf{x}})$ is a legitimate encryption of $(\mathbf{z}, \mathbf{x})$ under the PFFE scheme.

The only difference between the ciphertexts provided in the $\mathsf{iPRO.Eval}$ algorithm and the legitimate output of the $\mathsf{PFFE.Enc}$ algorithm is the replacement of $\mathsf{ct}_{\mathbf{x}} \leftarrow \mathsf{PFFE.EncPub}(\mathsf{mpk}, \mathbf{x}, \mathsf{st})$ by $\widetilde{\mathsf{ct}}_{\mathbf{x}}$. Recall that by Definition 6, the correctness guarantee of the PFFE scheme is preserved even when $\widetilde{\mathsf{ct}}_{\mathbf{x}}$ is used, if it holds that

$$\begin{aligned}\left\|\mathbf{e}'_{\mathbf{x}}\right\|_\infty &:= \left\|\widetilde{\mathsf{ct}}_{\mathbf{x}} - \hat{\mathbf{s}}_{\mathbf{z}}\left(\prod_{i=1}^{L}\hat{\mathbf{S}}_{i,x_i}\right)\right\|_\infty \\ &\le \beta_e, \end{aligned} \tag{14}$$

where $\beta_e := \sigma'_E \cdot \Theta(\sqrt{\lambda})$ in our parameter setting.

We now show that the error term $\mathbf{e}'_{\mathbf{x}}$ satisfies the bound in Eq. (14). For every $i \in [0, L]$, let $\mathbf{x}_i \in \{0,1\}^i$ be the prefix of $\mathbf{x}$ of length i, where $\mathbf{x}_0 := \epsilon$ and $\mathbf{x}_L := \mathbf{x}$. For every $i \in [0, L-1]$, define

$$\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i},$$

where $\hat{\mathbf{s}}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{z}}\prod_{j=1}^{i}\hat{\mathbf{S}}_{j,x_j}$. The error is defined by $\mathbf{e}_{\mathbf{x}_0} := \mathbf{e}_\epsilon$ and, for every $i \in [L-1]$,

$$\mathbf{e}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_{i-1}}\mathbf{E}_{K,i,x_i} + \mathbf{e}_{\mathbf{x}_{i-1}}\mathbf{K}_{i,x_i}.$$

Since

$$\mathbf{B}_{i-1}\mathbf{K}_{i,x_i} = \hat{\mathbf{S}}_{i,x_i}\mathbf{B}_i + \mathbf{E}_{K,i,x_i},$$

a direct induction gives, for every $i \in [0, L-1]$,

$$\mathbf{p}_{\mathbf{x}_i}\mathbf{K}_{i+1,x_{i+1}} = \mathbf{p}_{\mathbf{x}_{i+1}}.$$

For $i = L$, we have

$$\begin{aligned}\mathbf{p}_{\mathbf{x}_{L-1}}\mathbf{K}_{L,x_L} &= \hat{\mathbf{s}}_{\mathbf{x}_{L-1}}\mathbf{B}_{L-1}\mathbf{K}_{L,x_L} + \mathbf{e}_{\mathbf{x}_{L-1}}\mathbf{K}_{L,x_L} \\ &= \hat{\mathbf{s}}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}},\end{aligned}$$

where $\mathbf{e}_{\mathbf{x}} := \hat{\mathbf{s}}_{\mathbf{x}_{L-1}} \mathbf{E}_{K,L,x_L} + \mathbf{e}_{\mathbf{x}_{L-1}} \mathbf{K}_{L,x_L}$.

For every $i \in [0, L]$, define a constant $B_i \in \mathbb{N}$ such that $\left\|\mathbf{e}_{\mathbf{x}_i}\right\|_\infty \leq B_i$ for any $\mathbf{x}_i \in \{0,1\}^i$. From Lemma 2, it follows that

$$\begin{aligned} B_0 &\leq \sigma_E \sqrt{\lambda}, \\ B_i &\leq \left\|\hat{\mathbf{s}}_{\mathbf{x}_{i-1}} \mathbf{E}_{K,i,x_i}\right\|_\infty + \left\|\mathbf{e}_{\mathbf{x}_{i-1}} \mathbf{K}_{i,x_i}\right\|_\infty \\ &\leq \hat{n}^{i-1} \beta_s^i \cdot \sigma S \sqrt{\lambda} + B_{i-1} \cdot \gamma B \sqrt{\lambda} \end{aligned}$$

Solving this recurrence gives the following simple upper bound:

$$B_L \leq \sigma_E \sqrt{\lambda} \left(\gamma B \sqrt{\lambda}\right)^L + L \beta_s \sigma S \sqrt{\lambda} \cdot (\hat{n} \beta_s)^{L-1},$$

where we assume that $\hat{n}\beta_s > \gamma B \sqrt{\lambda}$ for a sufficiently large $\hat{n}$. Therefore, by the choice of our parameters in Eq. (9), the inequality in Eq. (14) holds, and the iPRO scheme is correct.

C.2 Missing Proof of Theorem 1

Fix any sampler $\mathsf{PROSamp}$ for Definition 4, any keyed circuit $C : \{0,1\}^\ell \times \{0,1\}^L \to \{0,1\}^M$, and let $(K, \mathsf{aux}_K) \leftarrow \mathsf{PROSamp}(1^\lambda)$. Let $K^{(0)} := K$, and let $K^{(1)} := K'$ be a key such that $C(K, \cdot)$ and $C(K', \cdot)$ are functionally equivalent. The iPRO precondition says that, for each $\eta \in \{0,1\}$,

$$\left\{1^\kappa, C, \{C(K^{(\eta)}, \mathbf{x})\}_{\mathbf{x} \in \{0,1\}^L}, \mathsf{aux}_K\right\} \overset{\mathrm{c}}{\approx} \left\{1^\kappa, C, \{\Delta_{\mathbf{x}} \leftarrow\!\$\, \{0,1\}^M\}_{\mathbf{x} \in \{0,1\}^L}, \mathsf{aux}_K\right\}.$$

We prove that the obfuscation with each key $K^{(\eta)}$ is indistinguishable from a common random distribution.

Our PFFE private-input sampler. For each $\eta \in \{0,1\}$, define the PFFE private-input sampler $\mathsf{PFFESamp}_{\mathbf{z},\eta}(1^\lambda)$ as follows. It samples $(K, \mathsf{aux}_K) \leftarrow \mathsf{PROSamp}(1^\lambda)$, lets $\mathbf{z}_\eta := K^{(\eta)}$, and outputs

$$(f, \mathbf{z}, \mathsf{aux}_{\mathbf{z}}) := (C, \mathbf{z}_\eta, \mathsf{aux}_K).$$

Thus, for every $\mathbf{x} \in \{0,1\}^L$,

$$C(\mathbf{z}_\eta, \mathbf{x}) = C(K^{(\eta)}, \mathbf{x}). \tag{15}$$

Hybrid distributions. For $\eta \in \{0,1\}$, the real obfuscation view for $K^{(\eta)}$ is

$$\mathsf{Real}^\eta = \left(C, \mathsf{mpk}, \mathsf{ct}_{\mathbf{z}}, \mathbf{p}_\epsilon, \{\mathbf{K}_{i,b}\}_{i \in [L],\, b \in \{0,1\}}, \mathsf{sk}_C, \mathsf{aux}_K\right).$$

For $h \in [L]$, define the hybrid distributions $\mathsf{Hyb}_0^{h,\eta}$ and $\mathsf{Hyb}_1^{h,\eta}$ as follows.

$$\mathsf{Hyb}_0^{h,\eta} = \begin{pmatrix} C, \mathsf{mpk}, \mathsf{ct}_{\mathbf{z}}, \{\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i}\}_{i=[0,h-1],\, \mathbf{x}_i \in \{0,1\}^i}, \\ \{\mathbf{K}_{i,b}\}_{i=[h,L],\, b\in\{0,1\}}, \mathsf{sk}_C, \mathsf{aux}_K \end{pmatrix},$$

$$\mathsf{Hyb}_1^{h,\eta} = \begin{pmatrix} C, \mathsf{mpk}, \mathsf{ct}^\star_{\mathbf{z}} \leftarrow\!\$\, \mathcal{CT}^z_{\lambda,\ell,L,M}, \{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\!\$\, \mathbb{Z}_q^{\hat{m}}\}_{i=[0,h-1],\, \mathbf{x}_i\in\{0,1\}^i}, \\ \{\mathbf{K}^\star_{i,b} \leftarrow\!\$\, \mathcal{D}^{\hat{m}\times\hat{m}}_{\mathbb{Z},\gamma_B}\}_{i=[h,L],\, b\in\{0,1\}}, \mathsf{sk}_C, \mathsf{aux}_K \end{pmatrix},$$

where $\hat{\mathbf{s}}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{z}} \prod_{j\in[i]} \hat{\mathbf{S}}_{j,x_j}$. For $h = L+1$, $\mathsf{Hyb}^{L+1,\eta}$ is the top distribution where the final samples $\{\mathbf{c}_{\mathbf{x}_L}\}_{\mathbf{x}_L\in\{0,1\}^L}$ are identified with the PFFE public ciphertexts $\{\mathsf{ct}_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}$, namely,

$$\mathsf{Hyb}_0^{L+1,\eta} = \begin{pmatrix} C, \mathsf{mpk}, \mathsf{ct}_{\mathbf{z}}, \{\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i}\}_{i=[0,L-1],\, \mathbf{x}_i\in\{0,1\}^i}, \\ \{\mathbf{c}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_i}\}_{\mathbf{x}_L\in\{0,1\}^L}, \mathsf{sk}_C, \mathsf{aux}_K \end{pmatrix},$$

and

$$\mathsf{Hyb}_1^{L+1,\eta} = \begin{pmatrix} C, \mathsf{mpk}, \mathsf{ct}^\star_{\mathbf{z}} \leftarrow\!\$\, \mathcal{CT}^z_{\lambda,\ell,L,M}, \{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\!\$\, \mathbb{Z}_q^{\hat{m}}\}_{i=[0,L-1],\, \mathbf{x}_i\in\{0,1\}^i}, \\ \{\mathbf{c}^\star_{\mathbf{x}_i} \leftarrow\!\$\, \mathbb{Z}_q^{\hat{m}}\}_{\mathbf{x}_L\in\{0,1\}^L}, \mathsf{sk}_C, \mathsf{aux}_K \end{pmatrix}.$$

The distribution $\mathsf{Hyb}_0^{1,\eta}$ is the same as the real obfuscation view Real^η, and the distribution $\mathsf{Hyb}_1^{1,\eta}$ no longer contains any information about $K^{(\eta)}$. Therefore, it suffices to show that these hybrid distributions are indistinguishable from each other. In what follows, we fix $\eta \in \{0,1\}$ because the same argument applies to both $\eta = 0$ and $\eta = 1$.

For every $h \in [L+1]$ and for any adversary $\mathcal{A}$, define an advantage function for the h-th hybrids by

$$\mathrm{Adv}^h_{\mathcal{A},\eta}(\lambda) := \left|\Pr\left[\mathcal{A}(\mathsf{Hyb}_0^{h,\eta}) = 1\right] - \Pr\left[\mathcal{A}(\mathsf{Hyb}_1^{h,\eta}) = 1\right]\right|.$$

When η is fixed, we write $\mathrm{Adv}^h_{\mathcal{A}}$ for $\mathrm{Adv}^h_{\mathcal{A},\eta}$.

Base case: $h = L+1$. We first show that, assuming the security of the PFFE scheme (Definition 7), the hybrids $\mathsf{Hyb}_0^{L+1,\eta}$ and $\mathsf{Hyb}_1^{L+1,\eta}$ are indistinguishable for every $\eta \in \{0,1\}$.

Lemma 10. Assume that the PFFE scheme is secure for the private-input sampler class $\{\mathsf{PFFESamp}_{\mathbf{z},\eta}\}_{\eta\in\{0,1\}}$, and the precondition of the iPRO scheme holds for $\mathsf{PROSamp}$, namely

$$\left\{1^\kappa, C, \left\{\mathbf{x}, C(K^{(\eta)}, \mathbf{x})\right\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}_K\right\} \stackrel{\mathrm{c}}{\approx} \left\{1^\kappa, C, \left\{\mathbf{x}, \Delta_{\mathbf{x}} \leftarrow\!\$\, \{0,1\}^M\right\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}_K\right\} \tag{16}$$

holds for every non-uniform adversary $\mathcal{B}$ of size $\mathsf{poly}(\kappa)$. Then, for every non-uniform adversary $\mathcal{A}$ of size $\mathsf{poly}(\kappa)$, $\mathrm{Adv}^{L+1}_{\mathcal{A},\eta}(\lambda)$ is negligible in κ.

Proof. We invoke the PFFE security for the private-input sampler $\mathsf{PFFESamp}_{\mathbf{z},\eta}$. For every non-uniform adversary $\mathcal{B}$, the advantage of $\mathcal{B}$ in the precondition and that of $\mathcal{A}$ in the postcondition are defined as follows:

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{PFFE.PRE}}_{\mathcal{B}}(\lambda) &:= \Pr\Big[\mathcal{B}\big(C, \mathsf{aux}_K, \{\mathbf{x}, C(K^{(\eta)}, \mathbf{x})\}_{\mathbf{x}\in\{0,1\}^L}\big) = 1\Big] \\
&\quad - \Pr\Big[\mathcal{B}\big(C, \mathsf{aux}_K, \{\mathbf{x}, \mathbf{y}^\star_{\mathbf{x}} \leftarrow\!\!\$\ \{0,1\}^M\}_{\mathbf{x}\in\{0,1\}^L}\big) = 1\Big], \\
\mathrm{Adv}^{\mathrm{PFFE.POST}}_{\mathcal{A}}(\lambda) &:= \Pr\big[\mathcal{A}\big(C, \mathsf{aux}_K, \mathsf{mpk}, \mathsf{aux}_{\mathsf{pffe}}, \mathsf{sk}_C, \mathsf{ct}_{\mathbf{z}}, \{\mathbf{x}, \mathsf{ct}_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}\big) = 1\big] \\
&\quad - \Pr\Big[\mathcal{A}\big(C, \mathsf{aux}_K, \mathsf{mpk}, \mathsf{aux}^\star_{\mathsf{pffe}}, \mathsf{sk}_C, \mathsf{ct}^\star_{\mathbf{z}}, \{\mathbf{x}, \mathsf{ct}^\star_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}\big) = 1\Big],
\end{aligned}$$

where $f(\mathbf{z}, \mathbf{x}) = C(K^{(\eta)}, \mathbf{x})$ holds by Eq. (15). By the indistinguishability in Eq. (16), the above advantage is negligible in κ. Thus, by the PFFE security, it holds that

$$\begin{aligned}
&\mathrm{Adv}^{\mathrm{PFFE.PRE}}_{\mathcal{B}}(\lambda) \ \geq\ \mathrm{Adv}^{\mathrm{PFFE.POST}}_{\mathcal{A}}(\lambda)/Q(\lambda') \ -\ \mathrm{negl}(\kappa), \\
\Rightarrow\quad &\mathrm{Adv}^{\mathrm{PFFE.POST}}_{\mathcal{A}}(\lambda) \leq Q(\lambda') \cdot \mathrm{Adv}^{\mathrm{PFFE.PRE}}_{\mathcal{B}}(\lambda) + \mathrm{negl}(\kappa) \ \leq\ \mathrm{negl}(\kappa).
\end{aligned}$$

Similarly,

$$\begin{aligned}
&\mathrm{Size}(\mathcal{B}) \ \leq\ Q(\lambda') \ \cdot\ \mathrm{Size}(\mathcal{A}) \ \leq\ \mathsf{poly}(\kappa) \\
\Rightarrow\quad &\mathrm{Size}(\mathcal{A}) \leq \mathsf{poly}(\kappa).
\end{aligned}$$

We finally confirm that the postcondition view of the PFFE security in the real and random cases are identical to the distributions $\mathsf{Hyb}^{L+1,\eta}_0$ and $\mathsf{Hyb}^{L+1,\eta}_1$, respectively. In the real case, by the path-factorized ciphertext structure (Definition 6), it follows that

$$\begin{aligned}
\mathsf{aux}_{\mathsf{pffe}} &= \Big(\{\mathbf{p}_{\mathbf{x}_i}\}_{i\in[0,L-1],\,\mathbf{x}_i\in\{0,1\}^i}\Big), \\
\{\mathsf{ct}_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L} &= \Big\{\hat{\mathbf{s}}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_L}\Big\}_{\mathbf{x}\in\{0,1\}^L}.
\end{aligned}$$

In the random case, $\mathsf{aux}^\star_{\mathsf{pffe}}$ is sampled uniformly in the same ambient space as $\mathsf{aux}_{\mathsf{pffe}}$, and $\mathsf{ct}^\star_{\mathbf{x}}$ is sampled uniformly in the same ambient space as $\mathsf{ct}_{\mathbf{x}}$ for every $\mathbf{x} \in \{0,1\}^L$. Therefore, the postcondition views in the real and random cases are identical to $\mathsf{Hyb}^{L+1,\eta}_0$ and $\mathsf{Hyb}^{L+1,\eta}_1$, respectively. This implies that $\mathrm{Adv}^{\mathrm{PFFE.POST}}_{\mathcal{A}}(\lambda) = \mathrm{Adv}^{L+1}_{\mathcal{A},\eta}(\lambda) \leq \mathsf{negl}(\lambda)(\kappa)$. This concludes the base step of the hybrid argument. □

Induction step: $h + 1 \to h$**.** We next prove that, for each $h \in [L]$, if the hybrids $\mathsf{Hyb}^{h+1,\eta}_0$ and $\mathsf{Hyb}^{h+1,\eta}_1$ are indistinguishable for every $\eta \in \{0,1\}$, then so are $\mathsf{Hyb}^{h,\eta}_0$ and $\mathsf{Hyb}^{h,\eta}_1$ for every $\eta \in \{0,1\}$.

Lemma 11. Assume that the following holds:

Algorithm 8 Samp_η^L

Input: 1^λ.
Output: $\widetilde{\mathbf{S}}_L \in \mathbb{Z}_q^{2^{L-1}\times \hat{n}}$, $\widetilde{\mathbf{P}}'_L \in \mathbb{Z}_q^{\hat{n}\times 2\hat{m}}$, $\mathsf{aux}_{L,1}$, $\mathsf{aux}_{L,2}$.

1: $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{PFFE.Setup}(1^\lambda, 1^\ell, 1^L, 1^M)$.
2: $(K, \mathsf{aux}_K) \leftarrow \mathsf{PROSamp}(1^\lambda)$.
3: $\mathbf{z}_\eta := K^{(\eta)}$.
4: $(\mathsf{ct}_{\mathbf{z}}, \mathsf{st}) \leftarrow \mathsf{PFFE.EncPriv}(\mathsf{mpk}, \mathbf{z}_\eta)$.
5: $(\hat{\mathbf{s}}_{\mathbf{z}}, \{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],b\in\{0,1\}}) \leftarrow \mathsf{PFFE.Extract}(\mathsf{st})$.
6: For every $i \in [0, L-2]$, $\mathbf{B}_i \leftarrow\!\!\$\ \mathbb{Z}_q^{\hat{n}\times\hat{m}}$.
7: $\mathcal{P}_{<L-1} := \left\{\mathbf{p}_{\mathbf{x}_i}\right\}_{i\in[0,L-2],\mathbf{x}_i\in\{0,1\}^i}$, where

$$\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i},$$
$$\hat{\mathbf{s}}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{z}} \prod_{j=1}^{i} \hat{\mathbf{S}}_{j,x_j},$$
$$\mathbf{e}_{\mathbf{x}_i} \leftarrow\!\!\$\ \mathcal{D}^{\hat{m}}_{\mathbb{Z},\sigma_E}.$$

8: $\mathcal{K}_{>L} := \emptyset$.
9: $\widetilde{\mathbf{S}}_L := \begin{pmatrix} \hat{\mathbf{s}}_{\mathbf{0}_{L-1}} \\ \vdots \\ \hat{\mathbf{s}}_{\mathbf{1}_{L-1}} \end{pmatrix}$.
10: $\widetilde{\mathbf{P}}'_L := (\hat{\mathbf{S}}_{L,0}, \hat{\mathbf{S}}_{L,1})$.
11: $\mathsf{sk}_C \leftarrow \mathsf{PFFE.KeyGen}(\mathsf{msk}, C)$.
12: $\mathsf{aux}_{L,1} := (\mathsf{ct}_{\mathbf{z}}, \mathcal{P}_{<L-1}, \mathcal{K}_{>L})$.
13: $\mathsf{aux}_{L,2} := (C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)$.
14: **return** $\widetilde{\mathbf{S}}_L, \widetilde{\mathbf{P}}'_L, \mathsf{aux}_{L,1}, \mathsf{aux}_{L,2}$.

– The LWE assumption (Assumption 1) for the parameters used in the iPRO construction.
– The private-coin hiding evasive LWE assumption (Assumption 3) for the parameters and samplers defined below.

Then, for every $h \in [L]$ and for every non-uniform adversary $\mathcal{A}$ of size $\mathsf{poly}(\kappa)$, there exists another non-uniform adversary $\mathcal{B}$ and a polynomial Q such that the following holds:

$$\mathrm{Adv}^{h+1}_{\mathcal{B}}(\lambda) \geq \mathrm{Adv}^{h}_{\mathcal{A}}(\lambda)/Q(\lambda') - \mathsf{negl}(\kappa),$$
$$\mathrm{Size}(\mathcal{B}) \leq Q(\lambda') \cdot \mathrm{Size}(\mathcal{A}),$$

where $\lambda' := \lambda^L$.

Proof. We apply Lemma 5, proven secure under the given assumption, to the matrix $\mathbf{B}_{h-1}$. Let Samp^h_η take as input 1^λ and output a tuple $(\widetilde{\mathbf{S}}_h, \widetilde{\mathbf{P}}'_h, \mathsf{aux}_{h,1}, \mathsf{aux}_{h,2})$ as defined in Algorithms 8, 9, using Algorithm 8 when $h = L$ and Algorithm 9 when $h \in [L-1]$. Let $\widetilde{\mathbf{E}}_h \leftarrow\!\!\$\ \mathcal{D}^{\hat{m}\times\hat{m}}_{\mathbb{Z},\sigma_S}$ and $\widetilde{\mathbf{P}}_h := \widetilde{\mathbf{P}}'_h + \widetilde{\mathbf{E}}_h$.

If $h = L$, the precondition requires that the following indistinguishability holds:

$$
\begin{aligned}
&\left(\widetilde{\mathbf{S}}_L\mathbf{B}_{L-1} + \mathbf{E}_L,\ \widetilde{\mathbf{S}}_L\widetilde{\mathbf{P}}_L + \mathbf{E}'_L,\ \mathsf{aux}_{L,1},\ \mathsf{aux}_{L,2}\right)\\
&\overset{\mathsf{dist}}{\equiv}\begin{pmatrix}
\{\mathbf{p}_{\mathbf{x}_{L-1}} := \hat{\mathbf{s}}_{\mathbf{x}_{L-1}}\mathbf{B}_{L-1} + \mathbf{e}_{\mathbf{x}_{L-1}}\}_{\mathbf{x}_{L-1}\in\{0,1\}^{L-1}},\\
\{\mathbf{c}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_i}\}_{\mathbf{x}_L\in\{0,1\}^L},\\
\left(\mathsf{ct}_{\mathbf{z}}, \{\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i}\}_{i=[0,L-2],\,\mathbf{x}_i\in\{0,1\}^i}\right)\\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix}\\
&\overset{\mathrm{c}}{\approx}\begin{pmatrix}
\{\mathbf{p}^\star_{\mathbf{x}_{L-1}} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{\mathbf{x}_{L-1}\in\{0,1\}^{L-1}},\\
\{\mathbf{c}^\star_{\mathbf{x}_i} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{\mathbf{x}_L\in\{0,1\}^L},\\
\left(\mathsf{ct}^\star_{\mathbf{z}} \leftarrow\!\$\ \mathcal{CT}^z_{\lambda,\ell,L,M}, \{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{i=[0,L-2],\,\mathbf{x}_i\in\{0,1\}^i}\right)\\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix}\\
&\overset{\mathsf{dist}}{\equiv}\left(\widetilde{\mathbf{C}}^\star_{L,\mathsf{PRE}} \leftarrow\!\$\ \mathbb{Z}_q^{2^{L-1}\times\hat{n}},\ \widetilde{\mathbf{C}}^\star_{L,\mathsf{POST}} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{n}\times 2\hat{m}},\ \mathsf{aux}^\star_{L,1} \leftarrow\!\$\ \mathcal{S},\ \mathsf{aux}_{L,2}\right).
\end{aligned}
$$

The corresponding postcondition guarantees that the following indistinguishability holds:

$$
\begin{aligned}
&\left(\widetilde{\mathbf{S}}_L\mathbf{B}_{L-1} + \mathbf{E}_L,\ \widetilde{\mathbf{K}}_L \leftarrow\!\$\ \mathbf{B}^{-1}_{L-1,\gamma_B}(\widetilde{\mathbf{P}}_L),\ \mathsf{aux}_{L,1},\ \mathsf{aux}_{L,2}\right)\\
&\overset{\mathsf{dist}}{\equiv}\begin{pmatrix}
\{\mathbf{p}_{\mathbf{x}_{L-1}} := \hat{\mathbf{s}}_{\mathbf{x}_{L-1}}\mathbf{B}_{L-1} + \mathbf{e}_{\mathbf{x}_{L-1}}\}_{\mathbf{x}_{L-1}\in\{0,1\}^{L-1}},\\
\{\mathbf{K}_{L,b} \leftarrow\!\$\ \mathbf{B}^{-1}_{L-1,\gamma_B}\left(\hat{\mathbf{S}}_{L,b} + \mathbf{E}_{K,L,b}\right)\}_{b\in\{0,1\}},\\
\left(\mathsf{ct}_{\mathbf{z}}, \{\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i}\}_{i=[0,L-2],\,\mathbf{x}_i\in\{0,1\}^i}\right)\\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix}\\
&\overset{\mathrm{c}}{\approx}\begin{pmatrix}
\{\mathbf{p}^\star_{\mathbf{x}_{L-1}} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{\mathbf{x}_{L-1}\in\{0,1\}^{L-1}},\\
\{\mathbf{K}^\star_{L,b} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{b\in\{0,1\}},\\
\left(\mathsf{ct}^\star_{\mathbf{z}} \leftarrow\!\$\ \mathcal{CT}^z_{\lambda,\ell,L,M}, \{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{i=[0,L-2],\,\mathbf{x}_i\in\{0,1\}^i}\right)\\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix}\\
&\overset{\mathsf{dist}}{\equiv}\left(\widetilde{\mathbf{C}}^\star_{L,\mathsf{PRE}} \leftarrow\!\$\ \mathbb{Z}_q^{2^{L-1}\times\hat{n}},\ \widetilde{\mathbf{K}}^\star_L \leftarrow\!\$\ \mathcal{D}^{\hat{m}\times\hat{m}}_{\mathbb{Z},\gamma_B},\ \mathsf{aux}^\star_{L,1} \leftarrow\!\$\ \mathcal{S},\ \mathsf{aux}_{L,2}\right).
\end{aligned}
$$

We can observe that the precondition and postcondition correspond to the indistinguishabilities between hybrids $\mathsf{Hyb}_0^{L+1,\eta}$ and $\mathsf{Hyb}_1^{L+1,\eta}$, and between $\mathsf{Hyb}_0^{L,\eta}$ and $\mathsf{Hyb}_1^{L,\eta}$, respectively.

For $h \in [L-1]$, the precondition requires that the following indistinguishability holds:

$$
\begin{aligned}
&\big(\widetilde{\mathbf{S}}_h \mathbf{B}_{h-1} + \mathbf{E}_h,\ \widetilde{\mathbf{S}}_h \widetilde{\mathbf{P}}_h + \mathbf{E}'_h,\ \mathsf{aux}_{h,1},\ \mathsf{aux}_{h,2}\big) \\
&\overset{\mathsf{dist}}{=} \begin{pmatrix}
\{\mathbf{p}_{\mathbf{x}_{h-1}} := \hat{\mathbf{s}}_{\mathbf{x}_{h-1}} \mathbf{B}_{h-1} + \mathbf{e}_{\mathbf{x}_{h-1}}\}_{\mathbf{x}_{h-1} \in \{0,1\}^{h-1}}, \\
\{\mathbf{p}_{\mathbf{x}_h} := \hat{\mathbf{s}}_{\mathbf{x}_h} \mathbf{B}_h + \mathbf{e}_{\mathbf{x}_h}\}_{\mathbf{x}_h \in \{0,1\}^h}, \\
\Big(\mathsf{ct}_{\mathbf{z}}, \{\mathbf{p}_{\mathbf{x}_i}\}_{i=[0,h-2],\, \mathbf{x}_i \in \{0,1\}^i}, \{\mathbf{K}_{i,b}\}_{i \in [h+1,L],\, b \in \{0,1\}}\Big) \\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix} \\
&\overset{c}{\approx} \begin{pmatrix}
\{\mathbf{p}^\star_{\mathbf{x}_{h-1}} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{\mathbf{x}_{h-1} \in \{0,1\}^{h-1}}, \\
\{\mathbf{p}^\star_{\mathbf{x}_h} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{\mathbf{x}_h \in \{0,1\}^h}, \\
\Big(\mathsf{ct}^\star_{\mathbf{z}} \leftarrow\!\$\ \mathcal{CT}^z_{\lambda,\ell,L,M}, \{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m} \times \hat{m}}\}_{i \in [0,h-2],\, \mathbf{x}_i \in \{0,1\}^i}, \{\mathbf{K}^\star_{i,b} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{i \in [h+1,L],\, b \in \{0,1\}}\Big) \\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix} \\
&\overset{\mathsf{dist}}{=} \big(\widetilde{\mathbf{C}}^\star_{h,\mathsf{PRE}} \leftarrow\!\$\ \mathbb{Z}_q^{2^{h-1} \times \hat{n}},\ \widetilde{\mathbf{C}}^\star_{h,\mathsf{POST}} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{n} \times 2\hat{m}},\ \mathsf{aux}^\star_{h,1} \leftarrow\!\$\ \mathcal{S},\ \mathsf{aux}_{h,2}\big).
\end{aligned}
$$

The corresponding postcondition guarantees that the following indistinguishability holds:

$$
\begin{aligned}
&\big(\widetilde{\mathbf{S}}_h \mathbf{B}_{h-1} + \mathbf{E}_h,\ \widetilde{\mathbf{K}}_h \leftarrow\!\$\ \mathbf{B}^{-1}_{h-1,\gamma_B}(\widetilde{\mathbf{P}}_h),\ \mathsf{aux}_{h,1},\ \mathsf{aux}_{h,2}\big) \\
&\overset{\mathsf{dist}}{=} \begin{pmatrix}
\{\mathbf{p}_{\mathbf{x}_{h-1}} := \hat{\mathbf{s}}_{\mathbf{x}_{h-1}} \mathbf{B}_{h-1} + \mathbf{e}_{\mathbf{x}_{h-1}}\}_{\mathbf{x}_{h-1} \in \{0,1\}^{h-1}}, \\
\{\mathbf{K}_{h,b} \leftarrow\!\$\ \mathbf{B}^{-1}_{h-1,\gamma_B}\left(\hat{\mathbf{S}}_{h,b} \mathbf{B}_h + \mathbf{E}_{K,h,b}\right)\}_{b \in \{0,1\}}, \\
\Big(\mathsf{ct}_{\mathbf{z}}, \{\mathbf{p}_{\mathbf{x}_i}\}_{i \in [0,h-2],\, \mathbf{x}_i \in \{0,1\}^i}, \{\mathbf{K}_{i,b}\}_{i \in [h+1,L],\, b \in \{0,1\}}\Big) \\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix} \\
&\overset{c}{\approx} \begin{pmatrix}
\{\mathbf{p}^\star_{\mathbf{x}_{h-1}} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{\mathbf{x}_{h-1} \in \{0,1\}^{h-1}}, \\
\{\mathbf{K}^\star_{h,b} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{b \in \{0,1\}}, \\
\Big(\mathsf{ct}^\star_{\mathbf{z}} \leftarrow\!\$\ \mathcal{CT}^z_{\lambda,\ell,L,M}, \{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m}}\}_{i \in [0,h-2],\, \mathbf{x}_i \in \{0,1\}^i}, \{\mathbf{K}^\star_{i,b} \leftarrow\!\$\ \mathbb{Z}_q^{\hat{m} \times \hat{m}}\}_{i \in [h+1,L],\, b \in \{0,1\}}\Big) \\
(C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)
\end{pmatrix} \\
&\overset{\mathsf{dist}}{=} \big(\widetilde{\mathbf{C}}^\star_{h,\mathsf{PRE}} \leftarrow\!\$\ \mathbb{Z}_q^{2^{h-1} \times \hat{n}},\ \widetilde{\mathbf{K}}^\star_h \leftarrow\!\$\ \mathcal{D}^{\hat{m} \times \hat{m}}_{\mathbb{Z},\gamma_B},\ \mathsf{aux}^\star_{h,1} \leftarrow\!\$\ \mathcal{S},\ \mathsf{aux}_{h,2}\big).
\end{aligned}
$$

We can observe that the precondition and postcondition correspond to the indistinguishabilities between hybrids $\mathsf{Hyb}_0^{h+1,\eta}$ and $\mathsf{Hyb}_1^{h+1,\eta}$, and between $\mathsf{Hyb}_0^{h,\eta}$ and $\mathsf{Hyb}_1^{h,\eta}$, respectively.

Therefore, for every $h \in [L]$, Lemma 5 implies that there exists a non-uniform adversary $\mathcal{B}$ and a polynomial Q such that the following holds:

$$
\begin{aligned}
&\mathrm{Adv}^{h+1}_{\mathcal{B}}(\lambda) \geq \mathrm{Adv}^{h}_{\mathcal{A}}(\lambda)/Q(\lambda') - \mathsf{negl}(\kappa), \\
&\mathrm{Size}(\mathcal{B}) \leq Q(\lambda') \cdot \mathrm{Size}(\mathcal{A}).
\end{aligned}
$$

□

To show that $\mathrm{Adv}^1_{\mathcal{B}}$ is negligible in λ, we adopt Lemma 3.12 of [2] with necessary modifications as follows.

Lemma 12. Suppose that $\mathrm{Adv}^{L+1}_{\mathcal{A}}(\lambda)$ is negligible in κ for every non-uniform adversary $\mathcal{A}$ such that $\mathrm{Size}(\mathcal{A}) \leq \mathsf{poly}(\kappa)$. Then $\mathrm{Adv}^1_{\mathcal{B}}(\lambda)$ is negligible in λ for every non-uniform adversary $\mathcal{B}$ such that $\mathrm{Size}(\mathcal{B}) \leq \mathsf{poly}(\lambda)$.

Proof. We prove the claim by contradiction. Suppose that there exists a non-uniform adversary $\mathcal{B}$ such that $\mathrm{Adv}^1_{\mathcal{B}}(\lambda)$ is non-negligible in λ and $\mathrm{Size}(\mathcal{B})$ is bounded by a polynomial in λ. Then there exist an infinite set $\mathcal{L} \subseteq \mathbb{N}$ and a polynomial p such that, for every $\lambda \in \mathcal{L}$,

$$\mathrm{Adv}^1_{\mathcal{B}}(\lambda) \geq \frac{1}{p(\lambda)} \qquad \text{and} \qquad \mathrm{Size}(\mathcal{B}) \leq p(\lambda).$$

Let

$$F(\lambda) := \lambda^{L \log \lambda}.$$

For every $h \in [L+1]$, define

$$t_h(\lambda) := p(\lambda)F(\lambda)^{h-1}, \qquad \epsilon_h(\lambda) := \frac{1}{p(\lambda)F(\lambda)^{h-1}}.$$

For each λ and $h \in [L+1]$, let $\mathsf{Statement}_{\lambda,h}$ be the assertion that there exists an adversary $\mathcal{A}_\lambda$ such that

$$\mathrm{Size}(\mathcal{A}_\lambda) \leq t_h(\lambda) \qquad \text{and} \qquad \mathrm{Adv}^h_{\mathcal{A}_\lambda}(\lambda) \geq \epsilon_h(\lambda).$$

By the choice of $\mathcal{B}$, $\mathsf{Statement}_{\lambda,1}$ is true for every $\lambda \in \mathcal{L}$.

Since $\kappa = \lambda^{L^2 \log \lambda} = F(\lambda)^L$, we have

$$t_{L+1}(\lambda) = p(\lambda)\kappa, \qquad \epsilon_{L+1}(\lambda) = \frac{1}{p(\lambda)\kappa}.$$

Since p is a polynomial in λ and κ eventually dominates every polynomial in λ, for all sufficiently large λ we have $p(\lambda) \leq \kappa$. Hence

$$t_{L+1}(\lambda) \leq \kappa^2, \qquad \epsilon_{L+1}(\lambda) \geq \kappa^{-2}.$$

By the assumption on Adv^{L+1}, $\mathsf{Statement}_{\lambda,L+1}$ is false for all sufficiently large λ. Indeed, otherwise, by non-uniformity, witnesses to $\mathsf{Statement}_{\lambda,L+1}$ for infinitely many λ would form a non-uniform adversary family of size at most κ^2 and advantage at least κ^{-2} infinitely often, contradicting negligibility in κ. Thus, there exists λ_0 such that for every $\lambda > \lambda_0$, $\mathsf{Statement}_{\lambda,L+1}$ is false.

For every $\lambda \in \mathcal{L}$ with $\lambda > \lambda_0$, $\mathsf{Statement}_{\lambda,1}$ is true whereas $\mathsf{Statement}_{\lambda,L+1}$ is false. Therefore, there exists $h^*_\lambda \in [L]$ such that $\mathsf{Statement}_{\lambda,h^*_\lambda}$ is true and $\mathsf{Statement}_{\lambda,h^*_\lambda+1}$ is false. Since $[L]$ is finite, by passing to an infinite subset $\mathcal{L}' \subseteq \mathcal{L}$, we may fix $h^* \in [L]$ such that $\mathsf{Statement}_{\lambda,h^*}$ is true and $\mathsf{Statement}_{\lambda,h^*+1}$ is false for all $\lambda \in \mathcal{L}'$.

By non-uniformity, choose witnesses to $\mathsf{Statement}_{\lambda,h^*}$ for $\lambda \in \mathcal{L}'$ and combine them into a non-uniform adversary family $\mathcal{A}$ of size at most $t_{h^*}(\lambda)$ on those security parameters. For all sufficiently large $\lambda \in \mathcal{L}'$, this size is at most $\mathsf{poly}(\kappa)$. Lemma 11 yields a non-uniform adversary $\mathcal{A}'$ and a polynomial Q such that

$$\begin{aligned}\mathrm{Adv}^{h^*+1}_{\mathcal{A}'}(\lambda) &\geq \frac{\mathrm{Adv}^{h^*}_{\mathcal{A}}(\lambda)}{Q(\lambda')} - \mathsf{negl}(\kappa) \\ &\geq \frac{\epsilon_{h^*}(\lambda)}{Q(\lambda')} - \mathsf{negl}(\kappa),\end{aligned}$$

and

$$\mathrm{Size}(\mathcal{A}') \leq Q(\lambda') \cdot \mathrm{Size}(\mathcal{A}).$$

Since $\epsilon_{h^*}(\lambda)/Q(\lambda')$ is inverse-polynomial in κ, the negligible term is at most $\epsilon_{h^*}(\lambda)/(2Q(\lambda'))$ for all sufficiently large $\lambda \in \mathcal{L}'$. Therefore,

$$\mathrm{Adv}^{h^*+1}_{\mathcal{A}'}(\lambda) \geq \frac{\epsilon_{h^*}(\lambda)}{2Q(\lambda')}.$$

Since Q is a fixed polynomial and $\lambda' = \lambda^L$, for all sufficiently large λ we have

$$F(\lambda) = \lambda^{L \log \lambda} \geq 2Q(\lambda').$$

Hence

$$\mathrm{Adv}^{h^*+1}_{\mathcal{A}'}(\lambda) \geq \frac{\epsilon_{h^*}(\lambda)}{F(\lambda)} = \epsilon_{h^*+1}(\lambda).$$

Moreover,

$$\begin{aligned}\mathrm{Size}(\mathcal{A}') &\leq Q(\lambda')\mathrm{Size}(\mathcal{A}) \\ &\leq Q(\lambda')t_{h^*}(\lambda) \\ &\leq F(\lambda)t_{h^*}(\lambda) \\ &= t_{h^*+1}(\lambda).\end{aligned}$$

Thus $\mathsf{Statement}_{\lambda,h^*+1}$ is true for sufficiently large $\lambda \in \mathcal{L}'$, contradicting the choice of h^* and $\mathcal{L}'$.

Therefore, $\mathrm{Adv}^1_{\mathcal{B}}(\lambda)$ is negligible in λ for every non-uniform adversary $\mathcal{B}$ such that $\mathrm{Size}(\mathcal{B}) \leq \mathsf{poly}(\lambda)$. □

By Lemmas 10 and 12, we conclude that $\mathrm{Adv}^1_{\mathcal{B}}(\lambda)$ is negligible in λ for all non-uniform adversaries $\mathcal{B}$ such that $\mathrm{Size}(\mathcal{B}) \leq \mathsf{poly}(\lambda)$ and for each $\eta \in \{0,1\}$. Thus, the real obfuscation views for $K^{(0)} = K$ and $K^{(1)} = K'$ are both indistinguishable from the same random hybrid. Hence, we obtain

$$\left\{C, \mathsf{iPRO.Obf}(1^\lambda, C, K), \mathsf{aux}_K\right\} \stackrel{c}{\approx} \left\{C, \mathsf{iPRO.Obf}(1^\lambda, C, K'), \mathsf{aux}_K\right\},$$

which implies the security of the iPRO construction. □

D Cryptanalysis of Evasive LWE

In this section, we review previous works on attacks against the evasive LWE assumption. For each category of attacks, we show that our concrete choice of the parameters and samplers in Section 3 is secure against all known attacks.

According to [59] and [3], the evasive LWE assumption relies on two heuristics:

1. **Heuristic 1: an adversary can use a preimage only in a semi-honest manner.** A preimage $\mathbf{K} \leftarrow\!\!\$\ \mathbf{B}^{-1}(\mathbf{P})$ cannot be used for anything other than deriving $\mathbf{SP}+\mathbf{EK}$ from $\mathbf{SB}+\mathbf{E}$; specifically, it does not provide an adversary with a non-trivial advantage in distinguishing the postcondition.
2. **Heuristic 2: an adversary cannot exploit structured error.** Although the error $\mathbf{EK}$ in the postcondition is structured, $\mathbf{SP}+\mathbf{EK}$ does not provide an adversary with non-trivial information about secrets beyond an LWE instance with unstructured error in the precondition—namely, $\mathbf{SP}+\mathbf{E}'$.

These heuristics have been used to build counterexamples against the evasive-LWE assumption, which satisfy the precondition but fail to satisfy the postcondition.

Early attacks (and one recent attack introduced in [62]) exploit the first heuristic [77, 30]. However, each relies on contrived data that fall outside the "natural" distribution of the matrix $\mathbf{P}$ and auxiliary information aux—for example, an obfuscation that hardcodes a trapdoor for $\mathbf{P}$ [77], instance-hiding witness encryption [62], and a trapdoor for a submatrix of $\mathbf{P}$ [30] [23]. Since the evasive LWE samplers employed in the security proof of Theorem 1 do not contain such contrived data, none of these attacks applies to our PRO and iO constructions.

Recent attacks exploit the second heuristic [44, 3, 59]. They are comparatively simpler and require only minor modifications to the "natural" distributions of P and aux used by existing schemes based on the evasive LWE assumption. Nonetheless, our construction can avoid these attacks by carefully choosing the parameters and samplers. We explain this by grouping the attacks into three categories and showing how attacks in each category are thwarted.

Attacks exploiting error correlation in GSW-FHE. The first category comprises the attacks described in Subsections 4.1, 5.1, and 5.2 of [3] and Subsection 7.3 of [59]. In short, these attacks are implemented by maliciously performing the homomorphic evaluation of a PRF within the GSW FHE scheme. Concretely, they employ a contrived circuit for that homomorphic evaluation—originally proposed by Hopkins et al. [57]—so that the PRF output and the FHE decryption error cancel each other out modulo 2. This is problematic because an adversary can then distinguish the postcondition distributions by checking whether certain linear equations over integers are solvable.

[23] One might argue that this last counterexample still looks "natural." To rule out such a counterexample, Brzuska et al. [30] formalized the private-coin binding and hiding evasive-LWE assumptions. Since our construction is proved under these assumptions, the trapdoor-based attack no longer applies to our construction.

For example, with a variant of the sampler adopted in the original construction of the AKY24 FE scheme [1], an adversary can obtain $\mathbf{v} := f(\mathbf{x}) + \mathbf{e}_B\mathbf{K} + \mathbf{e}_{\mathsf{fhe}} - \mathbf{e}_A\mathbf{H}_{f,x}$ over $\mathbb{Z}_q$, where $\mathbf{e}_A$ and $\mathbf{e}_B$ are LWE errors provided by an honest encryptor, $\mathbf{H}_{f,x}$ is the output of the EvalCX algorithm, $\mathbf{K}$ is a preimage, and $f(\mathbf{x})$ is a function that returns pseudorandom outputs by computing a PRF [3]. Here, $\left\|\mathbf{e}_B\mathbf{K} + \mathbf{e}_{\mathsf{fhe}} - \mathbf{e}_A\mathbf{H}_{f,x}\right\|_\infty$ is upper bounded by some small B. As shown in [3], this allows the adversary to obtain $\mathbf{v}$ over integers. Consequently, the contrived circuit enables the adversary to obtain $\mathbf{v} \mod 2 = \mathbf{e}_B\mathbf{K} - \mathbf{e}_A\mathbf{H}_{f,x}$, where $f(\mathbf{x}) + \mathbf{e}_{\mathsf{fhe}} = 0 \mod 2$ is used to obtain the RHS of the equation. This can be easily solved for $\mathbf{e}_B$ and $\mathbf{e}_A$ only in the real case where $\mathbf{sB} + \mathbf{e}_B$ is provided, since $\mathbf{K}$ and $\mathbf{H}_{f,x}$ are known to the adversary in the postcondition. Therefore, the existence of the solutions tells the adversary whether it is in the real or random case in the postcondition.

We stress, however, that these attacks crucially rely on a maliciously contrived circuit for the homomorphic evaluation of the PRF [3, 59]. Therefore, they do not pose a problem when the honest obfuscator avoids such a circuit. This is an acceptable assumption in our setting, since the obfuscator is assumed to be a trusted party.

If one wishes to make our construction secure against this class of attacks independently of how the homomorphic PRF-evaluation circuit is designed, it suffices to incorporate the same modification introduced in the updated version of the FE scheme of [1]. Specifically, that modification applies rounding operations to the BGG+ encodings, thereby replacing the correlated FHE decryption error with uncorrelated rounding errors. However, since our primary goal is to obtain an implementable iO construction, we prioritize simplicity and concrete efficiency, and deliberately refrain from introducing this additional modification.

Attacks when the error in the precondition is larger. The second category of attacks is described in Subsection 4.2 of [3]. The attack assumes that the error added to $\mathbf{SP}$ in the precondition is larger than one in the postcondition, i.e., $\|\mathbf{E}'\|_\infty > \|\mathbf{EK}\|_\infty$. Our construction can invalidate this attack simply by avoiding such parameter choices.

Attacks with an overlapping auxiliary instance. The last category is for an attack introduced in [44]. This assumes that aux contains an instance that overlaps with $\mathbf{SP} + \mathbf{E}'$ up to a small noise, and exploits the correlation between the two instances to distinguish the postcondition distributions.

We believe that both our PRO scheme and our PFFE scheme are resistant to this attack. This is because the samplers defined when private-coin evasive LWE is invoked in their security proofs, namely Samp_b^L and Samp_b^h in Algorithms 8, 9 for the PRO scheme and $\mathsf{Samp}_{\mathsf{PFFESamp}}$ in Algorithm 10 for the PFFE scheme, do not include in aux any instance that coincides with $\mathbf{SP}$ up to small noise.

E Missing Proofs in Section 4

E.1 Missing Proof of Lemma 8

We show that

$$\mathsf{Dec}(\mathsf{mpk}, \mathbf{x}, \mathsf{ct}, f, \mathsf{sk}_f) = f(\mathbf{z}, \mathbf{x})$$

holds, where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^L, 1^M)$, $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$, and $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathbf{z}, \mathbf{x})$. Recall that $\mathsf{ct} := (\mathsf{ct}_{\mathbf{z}}, \mathsf{ct}_{\mathbf{x}})$, where $\mathsf{ct}_{\mathbf{z}}$ and $\mathsf{ct}_{\mathbf{x}} := \mathbf{c}_{\mathbf{x}}$ are the outputs of $\mathsf{PFFE.EncPriv}$ and $\mathsf{PFFE.EncPub}$, respectively.

For every $i \in [0, L]$ and $\mathbf{x}_i \in \{0,1\}^i$, define

$$\mathbf{m}_{\mathbf{z},\mathbf{x}_i} := (1, \mathbf{z}', \mathbf{x}_i, \mathbf{0}_{L-i}, \mathbf{t}) \in \{0,1\}^{L'+n+1}, \tag{17}$$

where $i = 0$ corresponds to the empty string $\mathbf{x}_0 := \epsilon$. Then it holds that

$$\mathbf{m}_{\mathbf{z},\mathbf{x}_i} \mathbf{M}_{i+1,x_{i+1}} = \mathbf{m}_{\mathbf{z},\mathbf{x}_{i+1}}. \tag{18}$$

Let

$$\mathbf{s}_{\mathbf{x}} := \mathbf{s}_\epsilon \prod_{i\in[L]} \mathbf{S}_{i,x_i}.$$

Since the last extracted path factor is defined as $\hat{\mathbf{S}}_{L,b} := (\mathbf{M}_{L,b} \otimes \mathbf{S}_{L,b})\mathbf{B}_L$, we have

$$\begin{aligned}
\mathbf{c}_{\mathbf{x}} &:= \hat{\mathbf{s}}_{\mathbf{z}} \left(\prod_{i\in[L]} \hat{\mathbf{S}}_{i,x_i} \right) + \mathbf{e}_{\mathbf{x}} \\
&= (\mathbf{m}_{\mathbf{z}} \otimes \mathbf{s}_\epsilon) \left(\prod_{i\in[L-1]} (\mathbf{M}_{i,x_i} \otimes \mathbf{S}_{i,x_i}) \right) (\mathbf{M}_{L,x_L} \otimes \mathbf{S}_{L,x_L})\mathbf{B}_L + \mathbf{e}_{\mathbf{x}} \\
&= \left(\left(\mathbf{m}_{\mathbf{z}} \prod_{i\in[L]} \mathbf{M}_{i,x_i} \right) \otimes \left(\mathbf{s}_\epsilon \prod_{i\in[L]} \mathbf{S}_{i,x_i} \right) \right) \mathbf{B}_L + \mathbf{e}_{\mathbf{x}} \\
&= \left(\mathbf{m}_{\mathbf{z},\mathbf{x}_L} \otimes \mathbf{s}_{\mathbf{x}} \right) \mathbf{B}_L + \mathbf{e}_{\mathbf{x}},
\end{aligned}$$

where the last equality follows from Eq. (18).

By the definition of $\mathbf{K}_{\mathsf{att}}$, we have

$$\begin{aligned}
\mathbf{c}_{\mathbf{x},\mathsf{att}} &:= \mathbf{c}_{\mathbf{x}} \mathbf{K}_{\mathsf{att}} \\
&= \left(\mathbf{m}_{\mathbf{z},\mathbf{x}_L} \otimes \mathbf{s}_{\mathbf{x}} \right) \mathbf{B}_L \mathbf{K}_{\mathsf{att}} + \mathbf{e}_{\mathbf{x}} \mathbf{K}_{\mathsf{att}} \\
&= \left(\mathbf{m}_{\mathbf{z},\mathbf{x}_L} \otimes \mathbf{s}_{\mathbf{x}} \right) (\mathbf{u}_1 \otimes \mathbf{A}_{\mathsf{att}} - \mathbf{T}_{\mathsf{att}} \otimes \mathbf{G}) + \mathbf{e}_{\mathbf{x}} \mathbf{K}_{\mathsf{att}} \\
&= \mathbf{s}_{\mathbf{x}} \left(\mathbf{A}_{\mathsf{att}} - (1, \mathbf{z}', \mathbf{x}_L) \otimes \mathbf{G} \right) + \mathbf{e}_{\mathbf{x}} \mathbf{K}_{\mathsf{att}}.
\end{aligned}$$

Similarly, we have

$$\begin{aligned}\mathbf{c}_{\mathbf{x},t} &:= \mathbf{c}_{\mathbf{x}}\mathbf{K}_t \\ &= \mathbf{s}_{\mathbf{x}}\left(\mathbf{A}_t - \mathbf{t}\otimes\mathbf{G}\right) + \mathbf{e}_{\mathbf{x}}\mathbf{K}_t.\end{aligned}$$

By the key-homomorphic property in Lemma 7, it holds that

$$\begin{aligned}\mathbf{v}_{\mathbf{x}} &:= (\mathbf{c}_{\mathbf{x},\mathsf{att}}, \mathbf{c}_{\mathbf{x},t})\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}} \\ &= \mathbf{s}_{\mathbf{x}}\left(\left(\mathbf{A}_{\mathsf{att}} - (1,\mathbf{z}',\mathbf{x}_L)\otimes\mathbf{G}\right), \left(\mathbf{A}_t - \mathbf{t}\otimes\mathbf{G}\right)\right) \\ &\qquad + \left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}} \\ &= \mathbf{s}_{\mathbf{x}}\left(\mathbf{A}_G - (\mathbf{t}\mathbf{Y})\otimes\mathbf{G}\right)\left(\mathbf{I}_M \otimes \mathbf{G}^{-1}(\mathbf{u}_{n+1})\right) \\ &\qquad + \left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}} \\ &= \mathbf{s}_{\mathbf{x}}\mathbf{A}'_G + \mathbf{t}\mathbf{Y} + \left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}},\end{aligned}$$

where $\mathbf{Y} := G_F(\hat{\mathbf{z}}, \mathbf{x})$ is the FHE encryption of $F[\mathbf{x}](\mathbf{z}, K_B)$. From the linear decryption property in Lemma 6, it follows that

$$\begin{aligned}\mathbf{v}_{\mathbf{x}} &= \mathbf{s}_{\mathbf{x}}\mathbf{A}'_G + (F[\mathbf{x}](\mathbf{z}, K_B) + \mathbf{e}_{\mathsf{fhe}}) + \left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}} \\ &= \mathbf{s}_{\mathbf{x}}\mathbf{A}'_G + \left\lfloor\frac{q}{2}\right\rfloor f(\mathbf{z},\mathbf{x}) + \mathsf{PRF}_B(K_B,\mathbf{x}) + \mathbf{e}_{\mathsf{fhe}} + \left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}}.\end{aligned} \tag{19}$$

The decryption algorithm subtracts

$$\begin{aligned}\mathbf{d}_{\mathbf{x}} &:= \mathbf{c}_{\mathbf{x}}\mathbf{K}_G \\ &= \left(\mathbf{m}_{\mathbf{z},\mathbf{x}_L}\otimes\mathbf{s}_{\mathbf{x}}\right)\mathbf{B}_L\mathbf{K}_G + \mathbf{e}_{\mathbf{x}}\mathbf{K}_G \\ &= \left(\mathbf{m}_{\mathbf{z},\mathbf{x}_L}\otimes\mathbf{s}_{\mathbf{x}}\right)(\mathbf{u}_1\otimes\mathbf{A}'_G) + \mathbf{e}_{\mathbf{x}}\mathbf{K}_G \\ &= \mathbf{s}_{\mathbf{x}}\mathbf{A}'_G + \mathbf{e}_{\mathbf{x}}\mathbf{K}_G.\end{aligned}$$

Therefore, it holds that

$$\begin{aligned}\mu_{\mathbf{x}} &:= \mathbf{v}_{\mathbf{x}} - \mathbf{d}_{\mathbf{x}} \\ &= \left\lfloor\frac{q}{2}\right\rfloor f(\mathbf{z},\mathbf{x}) + \mathsf{PRF}_B(K_B,\mathbf{x}) + \mathbf{e}_{\mathsf{fhe}} + \left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}} - \mathbf{e}_{\mathbf{x}}\mathbf{K}_G.\end{aligned} \tag{20}$$

To ensure that the decryption output $\mathbf{y} \in \{0,1\}^M$ is recovered from $\mu_{\mathbf{x}}$ correctly, it suffices to show that the absolute value of the noise term in $\mu_{\mathbf{x}}$ is less than B because the output of $\mathsf{PRF}_B(K_B,\mathbf{x})$ is bounded by $\frac{q}{4} - B$. By Lemma 2,

$$\begin{aligned}&\left\|\mathbf{e}_{\mathbf{x}}\right\|_\infty \le \sigma_E\sqrt{\lambda}, \\ &\left\|\mathbf{K}_{\mathsf{att}}\right\|_\infty, \left\|\mathbf{K}_t\right\|_\infty, \left\|\mathbf{K}_G\right\|_\infty \le \gamma_B\sqrt{\lambda}.\end{aligned}$$

Hence, it follows that

$$\left\|\left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_{t}\right)\right\|_{\infty} \leq \sigma_E \gamma_B \lambda \cdot \max(L'm,\ (n+1)m),$$

and similarly

$$\left\|\mathbf{e}_{\mathbf{x}}\mathbf{K}_{G}\right\|_{\infty} \leq \sigma_E \gamma_B \lambda m.$$

Let $\mathsf{dep}(G)$ be the depth of the circuit G_F. By the shallowness property in Lemma 6, we have

$$\begin{aligned}\mathsf{dep}(G) &\leq \max(\mathsf{dep}_{\mathsf{PRF}_B}, \mathsf{dep}) \cdot \mathsf{poly}(\log m, \log\log q) \\ &\leq \mathsf{dep} \cdot \mathsf{poly}(\lambda, \log m, \log\log q).\end{aligned}$$

Therefore, by Lemma 7, we obtain

$$\begin{aligned}&\left\|\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}}\right\|_{\infty} \\ &\leq m \cdot \left\|\mathbf{H}_{G,\hat{\mathbf{z}},\mathbf{x}}\right\|_{\infty} \\ &\leq (n+1)\lceil \log_2 q \rceil \cdot m^{\mathcal{O}(\mathsf{dep}\cdot\mathsf{poly}(\lambda, \log m, \log\log q))} \\ &\leq m^{\mathcal{O}(\mathsf{dep}\cdot\mathsf{poly}(\lambda, \log m, \log\log q))},\end{aligned}$$

where the factor $(n+1)\lceil \log_2 q \rceil$ is absorbed into the $m^{\mathcal{O}(\mathsf{dep}\cdot\mathsf{poly}(\lambda, \log m, \log\log q))}$ term because $m \geq (n+1)\lceil \log_2 q \rceil$.

By Lemma 6,

$$\|\mathbf{e}_{\mathsf{fhe}}\|_{\infty} \leq \mathsf{poly}(\lambda) \cdot m^{\mathcal{O}(\mathsf{dep})}.$$

Consequently, we have

$$\begin{aligned}&\left\|\left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_{t}\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}}\right\|_{\infty} \\ &\leq (L'+n+1)m \cdot \left\|\left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_{t}\right)\right\|_{\infty} \cdot \left\|\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}}\right\|_{\infty} \\ &\leq (L'+n+1)m \cdot \sigma_E \gamma_B \lambda \cdot (L'+n+1)m \cdot m^{\mathcal{O}(\mathsf{dep}\cdot\mathsf{poly}(\lambda, \log m, \log\log q))} \\ &\leq (L')^2 \sigma_E \gamma_B \lambda \cdot m^{\mathcal{O}(\mathsf{dep}\cdot\mathsf{poly}(\lambda, \log m, \log\log q))}.\end{aligned}$$

Hence,

$$\begin{aligned}&\|\mathbf{e}_{\mathsf{fhe}}\|_{\infty} + \left\|\left(\mathbf{e}_{\mathbf{x}}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}}\mathbf{K}_{t}\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}}\right\|_{\infty} + \left\|\mathbf{e}_{\mathbf{x}}\mathbf{K}_{G}\right\|_{\infty} \\ &\leq \mathsf{poly}(\lambda) \cdot m^{\mathcal{O}(\mathsf{dep})} + (L')^2 \sigma_E \gamma_B \lambda \cdot m^{\mathcal{O}(\mathsf{dep}\cdot\mathsf{poly}(\lambda, \log m, \log\log q))} + \sigma_E \gamma_B \lambda m \\ &\leq (L')^2 \sigma_E \gamma_B \cdot \mathsf{poly}(\lambda) \cdot m^{\mathcal{O}(\mathsf{dep}\cdot\mathsf{poly}(\lambda, \log m, \log\log q))}. \qquad (21)\end{aligned}$$

Therefore, if we set B to the RHS of the inequality in Eq. (21), then the decryption output $\mathbf{y}$ is correctly recovered from $\mu_{\mathbf{x}}$. This completes the proof of correctness.

E.2 Missing Proofs of Theorem 3

Recall that the precondition and postcondition of the PFFE security (Definition 7) claim the following indistinguishability:

$$\begin{aligned}
&\big(f, \mathsf{aux}_{\mathbf{z}}, \{\mathbf{x}, f(\mathbf{z},\mathbf{x})\}_{\mathbf{x}\in\{0,1\}^L}\big)\\
\overset{\text{c}}{\approx}&\big(f, \mathsf{aux}_{\mathbf{z}}, \{\mathbf{x}, \mathbf{y}^\star_{\mathbf{x}} \leftarrow\!\!\$\ \{0,1\}^M\}_{\mathbf{x}\in\{0,1\}^L}\big), \qquad (22)\\
&\big(f, \mathsf{aux}_{\mathbf{z}}, \mathsf{mpk}, \mathsf{sk}_f, \mathsf{ct}_{\mathbf{z}}, \{\mathbf{x}, \mathsf{ct}_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}_{\mathsf{pffe}}\big)\\
\overset{\text{c}}{\approx}&\big(f, \mathsf{aux}_{\mathbf{z}}, \mathsf{mpk}, \mathsf{sk}_f, \mathsf{ct}^\star_{\mathbf{z}}, \{\mathbf{x}, \mathsf{ct}^\star_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L}, \mathsf{aux}^\star_{\mathsf{pffe}}\big). \qquad (23)
\end{aligned}$$

We prove that if the above indistinguishability in the precondition holds, then the above indistinguishability in the postcondition also holds. To prove it, we apply Lemma 4 to the matrix $\mathbf{B}_L$, used to sample preimages in the functional secret key sk_f. This allows us to reduce the task of proving the pseudorandomness of the ciphertexts ct in the setting where the adversary is given the preimages to proving it in the setting where the adversary is instead given the samples obtained by multiplying these preimages by $\mathsf{ct}_{\mathbf{x}} := \mathbf{c}_{\mathbf{x}}$.

Specifically, for a private-input sampler $\mathsf{PFFESamp}_{\mathbf{z}}$, let the sampler $\mathsf{Samp}_{\mathsf{PFFESamp}}$ for Lemma 4 take as input 1^λ and output a tuple $(\widetilde{\mathbf{S}}, \widetilde{\mathbf{P}}, \mathsf{aux}_{h,1}, \mathsf{aux}_{h,2})$ as defined in Algorithm 10. There exists a deterministic polynomial-time algorithm $\mathsf{Reconstruct}$ that on input aux_2 outputs $\widetilde{\mathbf{P}}$; in particular, it extracts $\mathbf{A}_{\mathsf{att}}, \mathbf{A}_t$ from aux_2 and derives $\mathbf{A}'_G$ from $\mathbf{A}_{\mathsf{att}}, \mathbf{A}_t$ and f.

The precondition requires the following indistinguishability:

$$\begin{aligned}
&\big(\mathbf{B}_L, \widetilde{\mathbf{P}}, \widetilde{\mathbf{S}}\mathbf{B}_L + \widetilde{\mathbf{E}}, \widetilde{\mathbf{S}}\widetilde{\mathbf{P}} + \widetilde{\mathbf{E}}', \mathsf{aux}_1, \mathsf{aux}_2\big)\\
\overset{\text{c}}{\approx}&\big(\mathbf{B}_L, \widetilde{\mathbf{P}}, \widetilde{\mathbf{C}}^\star_{\mathsf{PRE}} \leftarrow\!\!\$\ \mathbb{Z}_q^{2^L\times m_B}, \widetilde{\mathbf{C}}^\star_{\mathsf{POST}} \leftarrow\!\!\$\ \mathbb{Z}_q^{2^L\times((L'+n+1)m+M)}, \mathsf{aux}^\star_1 \leftarrow\!\!\$\ \mathcal{S}, \mathsf{aux}_2\big). \qquad (24)
\end{aligned}$$

The postcondition guarantees the following indistinguishability:

$$\begin{aligned}
&\big(\mathbf{B}_L, \widetilde{\mathbf{P}}, \widetilde{\mathbf{S}}\mathbf{B}_L + \widetilde{\mathbf{E}}, \widetilde{\mathbf{K}}, \mathsf{aux}_1, \mathsf{aux}_2\big)\\
\overset{\text{c}}{\approx}&\big(\mathbf{B}_L, \widetilde{\mathbf{P}}, \widetilde{\mathbf{C}}^\star_{\mathsf{PRE}} \leftarrow\!\!\$\ \mathbb{Z}_q^{2^L\times m_B}, \widetilde{\mathbf{K}}, \mathsf{aux}^\star_1 \leftarrow\!\!\$\ \mathcal{S}, \mathsf{aux}_2\big), \qquad (25)
\end{aligned}$$

where $\widetilde{\mathbf{K}} \leftarrow\!\!\$\ \mathbf{B}^{-1}_{L,\gamma_B}(\widetilde{\mathbf{P}})$.

Notably, it holds that

$$\{\mathsf{ct}_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^L} = \widetilde{\mathbf{S}}\mathbf{B}_L + \widetilde{\mathbf{E}}, \quad \mathsf{sk}_f = \widetilde{\mathbf{K}}.$$

The other information provided to the adversary in the postconditions of Equations (25) and (23) can be simulated from each other. Therefore, by Lemma 4, it suffices to show that if the indistinguishability in Eq. (22) holds, the indistinguishability in Eq. (24) also holds.

The indistinguishability in Eq. (24) is identical to the following indistinguishability:

$$\left(\begin{array}{l}\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t, \mathbf{A}'_G, \\ \left\{\mathbf{c}_{\mathbf{x}_L,\mathsf{att}} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_{\mathsf{att}} - \mathbf{m}_{\mathbf{x}_L,\mathsf{att}} \otimes \mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att}}\right\}, \\ \left\{\mathbf{c}_{\mathbf{x}_L,t} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_t - \mathbf{t} \otimes \mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,t}\right\}, \\ \left\{\mathbf{d}_{\mathbf{x}_L} := \mathbf{s}_{\mathbf{x}_L}\mathbf{A}'_G + \mathbf{e}_{\mathbf{x}_L,d}\right\}_{\mathbf{x}_L \in \{0,1\}^L}, \\ \left\{\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i}\right\}_{i \in [0,L], \mathbf{x}_i \in \{0,1\}^i}, \\ \mathsf{ct}_{\mathbf{z}} := \mathsf{FHE.Enc}(\mathsf{pk}, \mathbf{z}), f, \mathsf{aux}_{\mathbf{z}}\end{array}\right)$$

$$\stackrel{\mathrm{c}}{\approx} \left(\begin{array}{l}\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t, \mathbf{A}'_G, \\ \left\{\mathbf{c}^\star_{\mathbf{x}_L,\mathsf{att}} \leftarrow\$ \mathbb{Z}_q^{L'm}\right\}_{\mathbf{x}_L \in \{0,1\}^L}, \\ \left\{\mathbf{c}^\star_{\mathbf{x}_L,t} \leftarrow\$ \mathbb{Z}_q^{(n+1)m}\right\}_{\mathbf{x}_L \in \{0,1\}^L}, \\ \left\{\mathbf{d}^\star_{\mathbf{x}_L} \leftarrow\$ \mathbb{Z}_q^{M}\right\}_{\mathbf{x}_L \in \{0,1\}^L}, \\ \left\{\mathbf{p}^\star_{\mathbf{x}_i} \leftarrow\$ \mathbb{Z}_q^{m_B}\right\}_{i \in [0,L], \mathbf{x}_i \in \{0,1\}^i}, \\ \mathsf{ct}^\star_{\mathbf{z}} := \mathsf{FHE.Enc}(\mathsf{pk}, \mathbf{0}), f, \mathsf{aux}_{\mathbf{z}}\end{array}\right), \tag{26}$$

where, for every $\mathbf{x}_L \in \{0,1\}^L$ and some Gaussian parameters χ_c, χ_d, we define

$$\begin{aligned}&\mathbf{s}_{\mathbf{x}_L} := \mathbf{s}_\epsilon \prod_{i \in [L]} \mathbf{S}_{i,x_i} \in \mathbb{Z}_q^n, \\ &\mathbf{e}_{\mathbf{x}_L,\mathsf{att}} \leftarrow\$ \mathcal{D}^{L'm}_{\mathbb{Z},\chi_c}, \mathbf{e}_{\mathbf{x}_L,t} \leftarrow\$ \mathcal{D}^{(n+1)m}_{\mathbb{Z},\chi_c}, \mathbf{e}_{\mathbf{x}_L,d} \leftarrow\$ \mathcal{D}^{M}_{\mathbb{Z},\chi_d}, \\ &\mathbf{m}_{\mathbf{x}_L,\mathsf{att}} := (1, \mathbf{z}', \mathbf{x}_L) \in \{0,1\}^{L'},\end{aligned}$$

and relabel $\mathbf{c}_{\mathbf{x}_L}$ and $\mathbf{c}^\star_{\mathbf{x}_L}$ with $\mathbf{p}_{\mathbf{x}_L}$ and $\mathbf{p}^\star_{\mathbf{x}_L}$, respectively, for every $\mathbf{x}_L \in \{0,1\}^L$. To prove the above indistinguishability, we define a sequence of hybrids as follows.

Hyb_0. This is the same as the LHS of Eq. (26).

Hyb_1. This is the same as Hyb_0 except that, for every $\mathbf{x}_L \in \{0,1\}^L$, $\mathbf{d}_{\mathbf{x}_L}$ is replaced with

$$\mathbf{d}_{\mathbf{x}_L} := \mathbf{v}^T_{\mathbf{x}_L} - \left\lfloor \frac{q}{2} \right\rfloor f(\mathbf{z}, \mathbf{x}) - \mathbf{r}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_L,d}, \tag{27}$$

where

$$\mathbf{v}^T_{\mathbf{x}_L} := (\mathbf{c}_{\mathbf{x}_L,\mathsf{att}}, \mathbf{c}_{\mathbf{x}_L,t})\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}_L}$$

as defined in Algorithm 7, and $\mathbf{r}_{\mathbf{x}_L} := \mathsf{PRF}_B(K_B, \mathbf{x}_L)$.

Hyb_2. This is the same as Hyb_1 except that, for every $i \in [0, L]$ and $\mathbf{x}_i \in \{0,1\}^i$, $\mathbf{c}_{\mathbf{x}_L,\mathsf{att}}$, $\mathbf{c}_{\mathbf{x}_L,t}$, and $\mathbf{p}_{\mathbf{x}_i}$ are replaced with their randomized vectors $\mathbf{c}^\star_{\mathbf{x}_L,\mathsf{att}}$, $\mathbf{c}^\star_{\mathbf{x}_L,t}$, and $\mathbf{p}^\star_{\mathbf{x}_i}$.

Hyb_3. This is the same as Hyb_2 except that $\mathsf{ct}_{\mathbf{z}}$ is replaced with $\mathsf{ct}^\star_{\mathbf{z}}$, and $\mathbf{z}'$ used in $\mathbf{m}_{\mathbf{x}_L,\mathsf{att}}$ is replaced with a bit string of $\mathsf{ct}^\star_{\mathbf{z}}$.

Hyb_4. This is the same as Hyb_3 except that, for every $\mathbf{x}_L \in \{0,1\}^L$, $\mathbf{r}_{\mathbf{x}_L}$ is sampled from a uniform distribution over $[-\frac{q}{4}+B, \frac{q}{4}-B]^{1\times M}$.

Hyb_5. This is the same as Hyb_4 except that, for every $\mathbf{x}_L \in \{0,1\}^L$, $\mathbf{r}_{\mathbf{x}_L}$ is sampled from a uniform distribution over $[-\frac{q}{4}, \frac{q}{4}]^{1\times M}$.

Hyb_6. This is the same as Hyb_5 except that, for every $\mathbf{x}_L \in \{0,1\}^L$, $\mathbf{d}_{\mathbf{x}_L}$ is replaced with $\mathbf{d}^\star_{\mathbf{x}_L} \leftarrow\!\!\$\ \mathbb{Z}_q^M$.

We note that the last hybrid is identical to the RHS of Eq. (26). Therefore, it suffices to show that each pair of consecutive hybrids is indistinguishable.

Indistinguishability between Hyb_0 and Hyb_1. From Eq. (19), it holds that, for every $\mathbf{x}_L \in \{0,1\}^L$,

$$\begin{aligned}\mathbf{v}^T_{\mathbf{x}_L} &:= (\mathbf{c}_{\mathbf{x}_L,\mathsf{att}}, \mathbf{c}_{\mathbf{x}_L,t})\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}_L} \\ &= \mathbf{s}_{\mathbf{x}_L}\mathbf{A}'_G + \left\lfloor \frac{q}{2} \right\rfloor f(\mathbf{z},\mathbf{x}_L) + \mathbf{r}_{\mathbf{x}_L} + \mathbf{e}_{\mathsf{fhe}} + \left(\mathbf{e}_{\mathbf{x}_L}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}_L}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}_L}.\end{aligned}$$

Assuming

$$\chi_d \geq \lambda^{\omega(1)} \left\| \mathbf{e}_{\mathsf{fhe}} + \left(\mathbf{e}_{\mathbf{x}_L}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}_L}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}_L} \right\|_\infty,$$

the noise flooding (Lemma 3) ensures that

$$\begin{aligned}\mathbf{d}_{\mathbf{x}_L} &:= \mathbf{v}^T_{\mathbf{x}_L} - \left\lfloor \frac{q}{2} \right\rfloor f(\mathbf{z},\mathbf{x}) - \mathbf{r}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_L,d} \\ &= \mathbf{s}_{\mathbf{x}_L}\mathbf{A}'_G + \mathbf{e}_{\mathsf{fhe}} + \left(\mathbf{e}_{\mathbf{x}_L}\mathbf{K}_{\mathsf{att}}, \mathbf{e}_{\mathbf{x}_L}\mathbf{K}_t\right)\mathbf{H}'_{G,\hat{\mathbf{z}},\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_L,d} \\ &\stackrel{\mathrm{s}}{\approx} \mathbf{s}_{\mathbf{x}_L}\mathbf{A}'_G + \mathbf{e}_{\mathbf{x}_L,d}.\end{aligned}$$

Therefore, Hyb_0 and Hyb_1 are statistically indistinguishable.

Indistinguishability between Hyb_1 and Hyb_2. We prove that assuming the all-product LWE assumption (Assumption 4), Hyb_1 and Hyb_2 are computationally indistinguishable. Specifically, suppose there exists an adversary $\mathcal{A}$ that can distinguish between Hyb_1 and Hyb_2 with non-negligible advantage. Then we can construct an adversary $\mathcal{B}$ that can break the security of the all-product LWE assumption by internally invoking $\mathcal{A}$ as a subroutine. The assumption is instantiated with the parameters n, $m_A := m$, $\{m_{B,i} := m_B\}_{i\in[0,L-1]}$, $m_{B,L} = m_B + (L' - L + n + 1)m$, q, L, λ, and Gaussian parameters $\sigma_A := \chi_c$ and $\sigma_B := \sigma_{B,1} = \cdots = \sigma_{B,L}$ that satisfy

$$\sigma_E \geq (L' + n + 1)\sigma_B, \tag{28}$$

$$\sigma_A \geq \sigma_B. \tag{29}$$

The adversary $\mathcal{B}$ is constructed as follows.

1. $\mathcal{B}$ parses the instances provided by the all-product LWE challenger as follows:

$$\begin{pmatrix} \mathbf{A}, \left\{\mathbf{B}_i := \left(\mathbf{B}_{i,1}, \ldots, \mathbf{B}_{i,L'+n+1}\right)\right\}_{i\in[0,L-1]}, \\ \mathbf{B}_L := \left(\mathbf{B}_{L,1}, \ldots, \mathbf{B}_{L,L'+n+1}, \bar{\mathbf{A}}_{\mathsf{att},z}, \bar{\mathbf{A}}_t\right) \\ \left\{\mathbf{c}_{A,\mathbf{x}_L} := \mathbf{c}_{\mathbf{x}_L,\mathsf{att},x} \in \mathbb{Z}_q^{Lm}\right\}_{\mathbf{x}_L\in\{0,1\}^L} \\ \left\{\mathbf{c}_{B,\mathbf{x}_i} := \left(\mathbf{c}_{B,\mathbf{x}_i,1}, \ldots, \mathbf{c}_{B,\mathbf{x}_i,L'+n+1}\right)\right\}_{i\in[0,L-1],\mathbf{x}_i\in\{0,1\}^i} \\ \left\{\mathbf{c}_{B,\mathbf{x}_L} := \left(\mathbf{c}_{B,\mathbf{x}_L,1}, \ldots, \mathbf{c}_{B,\mathbf{x}_L,L'+n+1}, \mathbf{c}_{\mathbf{x}_L,\mathsf{att},z}, \mathbf{c}_{\mathbf{x}_L,\mathsf{att},t}\right)\right\}_{\mathbf{x}_L\in\{0,1\}^L} \end{pmatrix},$$

   where $\mathbf{B}_{i,j} \in \mathbb{Z}_q^{(n+1)\times m_B}$ for every $i \in [0, L]$ and $j \in [L' + n + 1]$, $\bar{\mathbf{A}}_{\mathsf{att},z} \in \mathbb{Z}_q^{(n+1)\times(L'-L)m}$, $\bar{\mathbf{A}}_t \in \mathbb{Z}_q^{(n+1)\times(n+1)m}$, $\mathbf{c}_{B,\mathbf{x}_i,j} \in \mathbb{Z}_q^{m_B}$ for every $i \in [0, L], \mathbf{x}_i \in \{0,1\}^i$ and $j \in [L' + n + 1]$, $\mathbf{c}_{\mathbf{x}_L,\mathsf{att},z} \in \mathbb{Z}_q^{(L'-L)m}$, and $\mathbf{c}_{\mathbf{x}_L,\mathsf{att},t} \in \mathbb{Z}_q^{(n+1)m}$ for every $\mathbf{x}_L \in \{0,1\}^L$.
2. Recall that Eq. (17) defines a vector $\mathbf{m}_{\mathbf{z},\mathbf{x}_i} \in \{0,1\}^{L'+n+1}$. For every $i \in [0, L]$ and $\mathbf{x}_i \in \{0,1\}^i$, compute

$$\begin{aligned} &\mathbf{A}_{\mathsf{att}} := \left(\bar{\mathbf{A}}_{\mathsf{att},z} + (1, \mathbf{z}') \otimes \mathbf{G},\ \mathbf{A}\right), \mathbf{A}_t := \bar{\mathbf{A}}_t + \mathbf{t} \otimes \mathbf{G}, \\ &\mathbf{c}_{\mathbf{x}_L,\mathsf{att}} := \left(\mathbf{c}_{\mathbf{x}_L,\mathsf{att},z}, \mathbf{c}_{A,\mathbf{x}_L}\right), \quad \mathbf{c}_{\mathbf{x}_L,\mathsf{att},t} := \mathbf{c}_{\mathbf{x}_L,\mathsf{att},t}, \\ &\mathbf{p}_{\mathbf{x}_i} := \mathbf{m}_{\mathbf{z},\mathbf{x}_i} \cdot \mathsf{diag}(\mathbf{c}_{B,\mathbf{x}_i,1}, \ldots, \mathbf{c}_{B,\mathbf{x}_i,L'+n+1}). \end{aligned} \tag{30}$$

   The other instances in Hyb_1 and Hyb_2 can be generated by $\mathcal{B}$ in the same manner, without any knowledge of the secret information in the all-product LWE instances. We later verify that if the instances provided by the all-product LWE challenger are real and random ones, then $\mathcal{B}$ simulates the view of $\mathcal{A}$ in Hyb_1 and Hyb_2, respectively.
3. $\mathcal{B}$ invokes $\mathcal{A}$ with the simulated instances, and outputs whatever $\mathcal{A}$ outputs.

The real case corresponds to Hyb_1. First, for every $i \in [0, L]$ and $\mathbf{x}_i \in \{0,1\}^i$, it follows that

$$\begin{aligned} \mathbf{p}_{\mathbf{x}_i} &= \hat{\mathbf{s}}_{\mathbf{z}} \left(\prod_{j\in[i]} \hat{\mathbf{S}}_{j,x_j}\right) \mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i} \\ &= (\mathbf{m}_{\mathbf{z}} \otimes \mathbf{s}_\epsilon) \left(\prod_{j\in[i]} (\mathbf{M}_{j,x_j} \otimes \mathbf{S}_{j,x_j})\right) \mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i} \\ &= \left(\left(\mathbf{m}_{\mathbf{z}} \prod_{j\in[i]} \mathbf{M}_{j,x_j}\right) \otimes \left(\mathbf{s}_\epsilon \prod_{j\in[i]} \mathbf{S}_{j,x_j}\right)\right) \mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i} \\ &= \left(\mathbf{m}_{\mathbf{z},\mathbf{x}_i} \otimes \mathbf{s}_{\mathbf{x}_i}\right) \mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i} \\ &= \mathbf{m}_{\mathbf{z},\mathbf{x}_i} \cdot (\mathbf{I}_{L'+n+1} \otimes \mathbf{s}_{\mathbf{x}_i}) \mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i}, \end{aligned}$$

where the third equality follows from Eq. (18). Assume that $\mathcal{B}$ adds additional Gaussian noises to $\mathbf{c}_{B,\mathbf{x}_i,j}$ so that the following equality holds; this is possible since the inequality in Eq. (28) holds.

$$\begin{aligned}
&\mathbf{m}_{\mathbf{z},\mathbf{x}_i} \cdot (\mathbf{I}_{L'+n+1} \otimes \mathbf{s}_{\mathbf{x}_i})\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i} \\
\overset{\mathsf{dist}}{=} &\mathbf{m}_{\mathbf{z},\mathbf{x}_i} \cdot \mathsf{diag}\left(\underbrace{\mathbf{s}_{\mathbf{x}_i}\mathbf{B}_{i,1} + \mathbf{e}_{\mathbf{x}_i,1}}_{=\mathbf{c}_{B,\mathbf{x}_i,1}}, \ldots, \underbrace{\mathbf{s}_{\mathbf{x}_i}\mathbf{B}_{i,L'+n+1} + \mathbf{e}_{\mathbf{x}_i,L'+n+1}}_{=\mathbf{c}_{B,\mathbf{x}_i,L'+n+1}} \right).
\end{aligned}$$

We also have

$$\begin{aligned}
\mathbf{c}_{\mathbf{x}_L,\mathsf{att}} =&\mathbf{s}_{\mathbf{x}_L}(\mathbf{A}_{\mathsf{att}} - (1, \mathbf{z}', \mathbf{x}_L) \otimes \mathbf{G}) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att}} \\
=&\mathbf{s}_{\mathbf{x}_L}\left(\bar{\mathbf{A}}_{\mathsf{att},z},\ \mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att}}, \\
\overset{\mathsf{dist}}{=} &\left(\underbrace{\mathbf{s}_{\mathbf{x}_L}\mathbf{A}_{\mathsf{att}} + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},z}}_{=\mathbf{c}_{A,\mathbf{x}_L,\mathsf{att},z}},\ \underbrace{\mathbf{s}_{\mathbf{x}_L}(\mathbf{A} - \mathbf{x}_L \otimes \mathbf{G}) + \mathbf{e}_{A,\mathbf{x}_L}}_{=\mathbf{c}_{A,\mathbf{x}_L}} \right),
\end{aligned}$$

where the last equality also assumes that $\mathcal{B}$ adds additional Gaussian noises to $\mathbf{c}_{A,\mathbf{x}_L}$ so that the following equality holds, which is possible since the inequality in Eq. (29) holds. Similarly,

$$\begin{aligned}
\mathbf{c}_{\mathbf{x}_L,\mathsf{att},t} =&\mathbf{s}_{\mathbf{x}_L}(\mathbf{A}_t - \mathbf{t} \otimes \mathbf{G}) + \mathbf{e}_{\mathbf{x}_L,t} \\
=&\mathbf{s}_{\mathbf{x}_L}\bar{\mathbf{A}}_t + \mathbf{e}_{\mathbf{x}_L,t},
\end{aligned}$$

where the RHS is identical to the corresponding part in $\mathbf{c}_{B,\mathbf{x}_i}$ provided by the all-product LWE challenger in the real case. Therefore, $\mathcal{B}$ simulates the view of $\mathcal{A}$ in Hyb_1 if the provided instances are real ones.

We can easily verify that $\mathcal{B}$ simulates the view of $\mathcal{A}$ in Hyb_2, especially $\mathbf{c}^\star_{\mathbf{x}_L,\mathsf{att}}$, $\mathbf{c}^\star_{\mathbf{x}_L,t}$, and $\mathbf{p}^\star_{\mathbf{x}_i}$, if the provided instances are random ones. Hence, $\mathcal{B}$ can break the security of the all-product LWE assumption with non-negligible advantage, which is a contradiction. This implies that Hyb_1 and Hyb_2 are computationally indistinguishable under the all-product LWE assumption.

Indistinguishability between Hyb_2 and Hyb_3. The only difference between Hyb_2 and Hyb_3 is the ciphertext $\mathsf{ct}_{\mathbf{z}}$, where $\mathsf{ct}_{\mathbf{z}}$ is an encryption of $\mathbf{z}$ in Hyb_2 and an encryption of $\mathbf{0}$ in Hyb_3. Recall that there is no information about the FHE decryption key $\mathbf{t}$ in both hybrids since $\mathbf{c}_{\mathbf{x}_L,\mathsf{att}}$, $\mathbf{c}_{\mathbf{x}_L,t}$, and $\mathbf{p}_{\mathbf{x}_i}$ are replaced with random vectors. Therefore, by the subexponential security of the FHE scheme (Definition 1), these two hybrids are computationally indistinguishable from each other.

Indistinguishability between Hyb_3 and Hyb_4. The only difference between Hyb_3 and Hyb_4 is whether $\mathbf{r}_{\mathbf{x}_L}$ is computed as the output of the PRF or sampled from the uniform distribution. Because there is no information about

the PRF key K_B in the adversary's view, as it is removed from $\mathsf{ct}_{\mathbf{z}}$ in Hyb_3, the subexponential security of the PRF scheme implies their indistinguishability.

Indistinguishability between Hyb_4 and Hyb_5. The only difference between Hyb_4 and Hyb_5 is the range of $\mathbf{r}_{\mathbf{x}_L}$ for every $\mathbf{x}_L \in \{0,1\}^L$, which is

$$\begin{aligned}\mathcal{U}_1 =&[-\frac{q}{4}+B, \frac{q}{4}-B] \text{ in } \mathsf{Hyb}_4\\ \mathcal{U}_2 =&[-\frac{q}{4}, \frac{q}{4}] \text{ in } \mathsf{Hyb}_5.\end{aligned}$$

Since B is set to be exponentially smaller than $\frac{q}{4}$, the statistical distance between $\mathcal{U}_1$ and $\mathcal{U}_2$ is negligible. Therefore, these hybrids are statistically indistinguishable.

Indistinguishability between Hyb_5 and Hyb_6. In Hyb_5, the highest-order bits of $\mathbf{v}_{\mathbf{x}_L}$ are the function output $f(\mathbf{z},\mathbf{x})$, whereas those in Hyb_6 are a uniformly random bit string. Because there is no information about $\mathbf{z}$ in the adversary's view, the indistinguishability in Eq. (22) implies that this difference is computationally indistinguishable. Additionally, in both Hyb_5 and Hyb_6, the other bits of $\mathbf{v}_{\mathbf{x}_L}$ are uniformly random over $[-\frac{q}{4}, \frac{q}{4}]^{1\times\ell}$. Hence, $\mathbf{v}_{\mathbf{x}_L}$ in Hyb_5 is indistinguishable from $\mathbf{v}_{\mathbf{x}_L} \leftarrow\!\!\$\ \mathbb{Z}_q^\ell$ in Hyb_6. By the relation between $\mathbf{v}_{\mathbf{x}_L}$ and $\mathbf{d}_{\mathbf{x}_L}$ shown in Eq. (27), $\mathbf{d}_{\mathbf{x}_L}$ in Hyb_5 is indistinguishable from $\mathbf{d}_{\mathbf{x}_L} \leftarrow\!\!\$\ \mathbb{Z}_q^M$ in Hyb_6. Therefore, Hyb_5 and Hyb_6 are computationally indistinguishable. □

F Alternative Construction of PFFE without All-Product LWE

In this appendix, we provide an alternative construction of the PFFE scheme that does not rely on the all-product LWE assumption.

Alternative key-homomorphic encoding. Recall that the construction in Section 4 requires all-product LWE to prove the pseudorandomness of the BGG+ encodings. Specifically, an encoding corresponding to each input bit $x_i \in \{0,1\}$ is defined as follows:

$$\mathbf{c}_{\mathbf{x}_L,\mathsf{att},i} := \mathbf{s}_{\mathbf{x}_L}(\mathbf{A}_i - x_i\mathbf{G}) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i},$$

where $\mathbf{s}_{\mathbf{x}_L} := \mathbf{s}_\epsilon \prod_{j\in[L]} \mathbf{S}_{j,x_j} \in \mathbb{Z}_q^{(n+1)}$ is a secret vector that depends on the entire input $\mathbf{x}_L$ and $\mathbf{A}_i \in \mathbb{Z}_q^{(n+1)\times m}$ is a public matrix that depends on the input index i.

The alternative construction instead defines the encoding as follows:

$$\mathbf{c}_{\mathbf{x}_L,\mathsf{att},i} \approx \mathbf{s}_{\mathbf{x}_L}(\mathbf{A}_i - x_i\mathbf{D}) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i},$$

where $\mathbf{D} \in \mathbb{Z}_q^{(n+1)\times m}$ is a fixed public matrix that is uniformly sampled over $\mathbb{Z}_q^{(n+1)\times m}$. To prove the pseudorandomness of this encoding, we can simply in-

voke the subexponentially hardness of LWE for each term as follows:

$$
\begin{aligned}
&\mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_i - x_i\mathbf{D}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i} \\
&= \mathbf{s}_{\mathbf{x}_{L-1}}\mathbf{S}_{L,x_L}\left(\mathbf{A}_i - x_i\mathbf{D}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i} \\
&\overset{\mathrm{s}}{\approx} \mathbf{s}_{\mathbf{x}_{L-1}}\left(\left(\mathbf{S}_{L,x_L}\mathbf{A}_i + \mathbf{e}'_{x_L,i}\right) - x_i\left(\mathbf{S}_{L,x_L}\mathbf{D} + \mathbf{e}'_{x_L,D}\right)\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i} \\
&\overset{\mathrm{c}}{\approx} \mathbf{s}_{\mathbf{x}_{L-1}}\left(\mathbf{A}'_{x_L,i} - x_i\mathbf{D}'_{x_L}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i} \\
&\cdots \\
&\approx \mathbf{s}_{\epsilon,x_1,\ldots,x_{x_j}}\left(\mathbf{A}'_{x_{j+1},\ldots,x_L,i} - x_i\mathbf{D}'_{x_{j+1},\ldots,x_L}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i} \\
&\overset{\mathrm{s}}{\approx} \mathbf{s}_{\epsilon,x_1,\ldots,x_{x_{j-1}}}\left(\left(\mathbf{S}_{j,x_j}\mathbf{A}'_{x_{j+1},\ldots,x_L,i} + \mathbf{e}'_{x_j,\ldots,x_L,i}\right) - x_i\left(\mathbf{S}_{j,x_j}\mathbf{D}'_{x_{j+1},\ldots,x_L} + \mathbf{e}'_{x_j,\ldots,x_L,D}\right)\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i} \\
&\overset{\mathrm{c}}{\approx} \mathbf{s}_{\epsilon,x_1,\ldots,x_{j-1}}\left(\mathbf{A}'_{x_j,\ldots,x_L,i} - x_i\mathbf{D}'_{x_j,\ldots,x_L}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},i} \\
&\cdots \\
&\approx \mathbf{r}_{x_1,\ldots,x_L,i} - x_i\mathbf{v}_{x_1,\ldots,x_L}. \qquad (31)
\end{aligned}
$$

In the hybrids above, for each statistical indistinguishability step, fresh error terms $\mathbf{e}_{x_j,\ldots,x_L,A_i}$ and $\mathbf{e}_{x_j,\ldots,x_L,D}$ are inserted for every $j \in \{L, L-1, \ldots, 0\}$ via noise flooding, which is allowed because the secret $\mathbf{s}_{\mathbf{x}_j}$ is a low-norm vector. For computational indistinguishability, we invoke LWE with respect to $\mathbf{S}_{j,x_j}$ as the LWE secret and

$$\left\{\mathbf{A}'_{x_{j+1},\ldots,x_L,i}, \mathbf{D}'_{x_{j+1},\ldots,x_L,i}\right\}_{(x_{j+1},\ldots,x_L)\in\{0,1\}^{L-j}}$$

as public matrices, thereby replacing them with random matrices $\mathbf{A}'_{x_j,\ldots,x_L,i}$ and $\mathbf{D}'_{x_j,\ldots,x_L,i}$.

Modification for homomorphic multiplication. Unfortunately, this encoding does not seem to support homomorphic multiplication, because an evaluator cannot sample a preimage with respect to $\mathbf{D}$ without its trapdoor $\mathbf{D}^{-1}_{\gamma_D}$. Therefore, the obfuscator provides additional preimages for each multiplication gate in the circuit G_F being evaluated over encodings in Algorithm 7. Specifically, for a multiplication gate g with input wires u and v and output wire w, the evaluator holds two input encodings:

$$
\begin{aligned}
\mathbf{c}_{\mathbf{x}_L,u} &:= \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_u - x_u\mathbf{D}\right) + \mathbf{e}_{\mathbf{x}_L,u}, \\
\mathbf{c}_{\mathbf{x}_L,v} &:= \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_v - x_v\mathbf{D}\right) + \mathbf{e}_{\mathbf{x}_L,v}.
\end{aligned}
$$

The obfuscator who holds the trapdoor $\mathbf{D}^{-1}_{\gamma_D}$ provides a preimage $\mathbf{L}_v \leftarrow\!\!\$\ \mathbf{D}^{-1}_{\gamma_D}(\mathbf{A}_v)$.

Then the evaluator can compute the output encoding as follows:

$$
\begin{aligned}
\mathbf{c}_{\mathbf{x}_L,w} &:= \mathbf{c}_{\mathbf{x}_L,u}\,\mathbf{L}_v + \mathbf{c}_{\mathbf{x}_L,v}\left(x_u\mathbf{I}\right) \\
&\approx \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_u\mathbf{L}_v - x_u\mathbf{A}_v\right) + \mathbf{s}_{\mathbf{x}_L}\left(x_v\mathbf{A}_v - (x_u\cdot x_v)\mathbf{D}\right) \\
&= \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_u\mathbf{L}_v - (x_u\cdot x_v)\mathbf{D}\right).
\end{aligned}
$$

The matrix $\mathbf{A}_u\mathbf{L}_v$ depends only on $\mathbf{A}_u$ and $\mathbf{A}_v$ so that the obfuscator can derive it without knowing the inputs x_u, x_v.

We next discuss how to prove the security of the PFFE construction with the above modifications to encodings under the LWE and evasive LWE assumptions. Assuming that the matrix $\mathbf{A}_v$ is uniformly random, the preimage $\mathbf{L}_v$ and the target matrix $\mathbf{A}_v$ are statistically indistinguishable from random matrices [49]:

$$\left\{\mathbf{D}, \mathbf{L}_v \leftarrow\$ \mathbf{D}_{\gamma_D}^{-1}(\mathbf{A}_v), \mathbf{A}_v \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m}\right\} \overset{\text{s}}{\approx} \left\{\mathbf{D}, \mathbf{L}_v \leftarrow\$ \mathcal{D}_{\mathbb{Z},\gamma D}^{m_D \times m}, \mathbf{A}_v := \mathbf{D}\mathbf{L}_v\right\}. \tag{32}$$

Therefore, in the hybrid where $\mathbf{L}_v$ no longer depends on the trapdoor $\mathbf{D}_{\gamma_D}^{-1}$, we can invoke the LWE assumption with respect to $\mathbf{D}$ to show the pseudorandomness of $\mathbf{c}_{\mathbf{x}_L,w}$ as in Eq. (31).

However, unless v is an input wire of the circuit, $\mathbf{A}_v$ is generally not a uniformly random matrix, and therefore the indistinguishability in Eq. (32) does not hold. To address this issue, the obfuscator also provides a preimage for every gate that replaces the non-uniform matrix after gate evaluation, namely $\mathbf{A}_u + \mathbf{A}_v$ for addition gate and $\mathbf{A}_u\mathbf{L}_v$ for multiplication gate, with a uniformly random matrix sampled for each gate-output wire w.

Recall that the ciphertext $\mathsf{ct}_{\mathbf{x}_L}$ is defined by $\mathbf{c}_{\mathbf{x}_L} := \left(\mathbf{m}_{\mathbf{z},\mathbf{x}} \otimes \mathbf{s}_{\mathbf{x}_L}\right)\mathbf{B}_L + \mathbf{e}_{\mathbf{x}_L}$, where the first element of $\mathbf{m}_{\mathbf{z},\mathbf{x}}$ is 1. For each multiplication gate g, the obfuscator samples $\mathbf{A}_w \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m}$ and provides a preimage $\mathbf{M}_g \leftarrow\$ \mathbf{B}_{L,\gamma_B}^{-1}(\mathbf{u}_1 \otimes (\mathbf{A}_w - \mathbf{A}_u\mathbf{L}_v))$. Then the evaluator computes

$$\begin{aligned}
\mathbf{c}_{\mathbf{x}_L,w} &:= \mathbf{c}_{\mathbf{x}_L}\mathbf{M}_g + \mathbf{c}_{\mathbf{x}_L,u}\,\mathbf{L}_v + \mathbf{c}_{\mathbf{x}_L,v}\,(x_u\mathbf{I}) \\
&\approx \mathbf{s}_{\mathbf{x}_L}\,(\mathbf{A}_w - \mathbf{A}_u\mathbf{L}_v) + \mathbf{s}_{\mathbf{x}_L}\,(\mathbf{A}_u\mathbf{L}_v - (x_u \cdot x_v)\mathbf{D}) \\
&= \mathbf{s}_{\mathbf{x}_L}\,(\mathbf{A}_w - (x_u \cdot x_v)\mathbf{D})\,.
\end{aligned}$$

A similar procedure applies to addition gates. Hence, the matrix $\mathbf{A}_w$ corresponding to any wire in the circuit is uniformly random, which ensures that the indistinguishability in Eq. (32) always holds and, consequently, the preimage $\mathbf{L}_v$ is indistinguishable from a random Gaussian matrix.

The remaining task for proving the security of the modified PFFE construction is to show that the preimage $\mathbf{M}_g$ preserves the pseudorandomness of the LWE samples and encodings. To this end, we include each additional preimage $\mathbf{M}_g$ as additional columns of the matrix $\widetilde{\mathbf{P}}$ output by the evasive LWE sampler $\mathsf{Samp}_{\mathsf{PFFESamp}}$.

For simplicity, we first consider the case where the circuit contains only one multiplication gate g with input wires u and v and output wire w. When we invoke Lemma 4 for this modified sampler, its precondition adds the following

samples to the LHS and RHS distributions in Eq. (26):

$$\begin{pmatrix} \mathbf{A}_u, \mathbf{A}_v, \mathbf{A}_w \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m}, \\ \mathbf{D} \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m_D}, \mathbf{L}_v \leftarrow\$ \mathbf{D}_{\gamma_D}^{-1}(\mathbf{A}_v), \\ \left\{\mathbf{b}_{\mathbf{x}_L,g} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - \mathbf{A}_u\mathbf{L}_v\right) + \mathbf{e}_{\mathbf{x}_L,g}\right\}_{\mathbf{x}_L\in\{0,1\}^L} \end{pmatrix}$$
$$\approx^{\mathrm{c}} \begin{pmatrix} \mathbf{A}_u, \mathbf{A}_v, \mathbf{A}_w \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m}, \\ \mathbf{D} \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m_D}, \mathbf{L}_v \leftarrow\$ \mathbf{D}_{\gamma_D}^{-1}(\mathbf{A}_v), \\ \left\{\mathbf{b}_{\mathbf{x}_L,g} \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m}\right\}_{\mathbf{x}_L\in\{0,1\}^L} \end{pmatrix},$$

By a noise flooding argument, it holds that

$$\begin{aligned} \mathbf{b}_{\mathbf{x}_L,g} &\approx^{\mathrm{s}} \mathbf{s}_{\mathbf{x}_L}\mathbf{A}_w - \left(\mathbf{c}_{\mathbf{x}_L,u}\,\mathbf{L}_v + \mathbf{c}_{\mathbf{x}_L,v}\,(x_u\mathbf{I})\right) - (x_u\cdot x_v)\mathbf{s}_{\mathbf{x}_L}\mathbf{D} + \mathbf{e}_{\mathbf{x}_L,g} \\ &= \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - (x_u\cdot x_v)\mathbf{D}\right) + \mathbf{e}_{\mathbf{x}_L,g} - \left(\mathbf{c}_{\mathbf{x}_L,u}\,\mathbf{L}_v + \mathbf{c}_{\mathbf{x}_L,v}\,(x_u\mathbf{I})\right) \end{aligned}$$

Therefore, to prove the pseudorandomness of $\mathbf{b}_{\mathbf{x}_L,g}$, it suffices to show that the encoding $\mathbf{c}_{\mathbf{x}_L,w} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - (x_u\cdot x_v)\mathbf{D}\right) + \mathbf{e}_{\mathbf{x}_L,g}$ is pseudorandom. We can show this by first applying the statistical indistinguishability in Eq. (32) to remove the dependence on the trapdoor $\mathbf{D}_{\gamma_D}^{-1}$ from the adversary's view, and then applying the LWE assumption as in Eq. (31).

In the general case where the circuit contains multiple gates, the wire w may be used as an input wire v' of a subsequent gate g'. In this case, after applying the indistinguishability in Eq. (32), the matrix $\mathbf{A}_w$ is replaced with $\mathbf{D}\mathbf{L}_w$, where $\mathbf{L}_w \leftarrow\$ \mathcal{D}_{\mathbb{Z},\gamma_D}^{m_D\times m}$. Applying the LWE assumption with respect to $\mathbf{D}$, we have

$$\begin{aligned} \mathbf{c}_{\mathbf{x}_L,w} &:= \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{D}\mathbf{L}_w - (x_u\cdot x_v)\mathbf{D}\right) + \mathbf{e}_{\mathbf{x}_L,g} \\ &\approx^{\mathrm{s}} \left((\mathbf{s}_{\mathbf{x}_L}\mathbf{D} + \mathbf{e}'_{\mathbf{x}_L,g,D})\mathbf{L}_w\right) - (x_u\cdot x_v)\left(\mathbf{s}_{\mathbf{x}_L}\mathbf{D} + \mathbf{e}'_{\mathbf{x}_L,g,D}\right) + \mathbf{e}_{\mathbf{x}_L,g} \\ &\approx^{\mathrm{c}} \left(\mathbf{v}_{\mathbf{x}_L}\mathbf{L}_w\right) - (x_u\cdot x_v)\mathbf{v}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_L,g} \\ &\approx^{\mathrm{s}} \left(\mathbf{v}_{\mathbf{x}_L}\mathbf{L}_w + \mathbf{e}''_{\mathbf{x}_L,g,D}\right) - (x_u\cdot x_v)\mathbf{v}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_L,g} \\ &\approx^{\mathrm{c}} \mathbf{r}_{\mathbf{x}_L,w} - (x_u\cdot x_v)\mathbf{v}_{\mathbf{x}_L} + \mathbf{e}_{\mathbf{x}_L,g}, \end{aligned}$$

where the statistical indistinguishabilities hold by noise flooding, the first computational indistinguishability holds by invoking the LWE assumption with respect to $\mathbf{D}$ multiple times as in Eq. (31), and the second computational indistinguishability holds by the LWE assumption with respect to the low-norm matrix $\mathbf{L}_w$ [20]. Since $\mathbf{r}_{\mathbf{x}_L,w}$ is uniformly random for every input $\mathbf{x}_L$ and wire w, this shows the pseudorandomness of $\mathbf{c}_{\mathbf{x}_L,w}$ and $\mathbf{b}_{\mathbf{x}_L,g}$.

G Details of Minor Optimizations

G.1 Reducing Peak Memory Usage in Preimage Sampling

We describe an optimization of our PFFE and iPRO schemes in a non-black-box manner. The goal of this optimization is to reduce the number of rows $\hat{n} = n_B$ of the matrix $\mathbf{B}_i \in \mathbb{Z}_q^{\hat{n} \times \hat{m}}$ that is sampled together with its trapdoor $\mathbf{B}_{i,\gamma_B}^{-1}$, and more specifically to make $\hat{n}$ constant, independently of the input size L. This is significant for making the peak memory usage of preimage sampling constant, independently of L. In our implementation, preimage sampling is the most memory-consuming operation; indeed, the data size of a trapdoor grows on the order of $\hat{n}^2 \cdot \lceil \log_2 q \rceil$ [31]. Therefore, if $\hat{n}$ grows proportionally to L, then preimage sampling becomes infeasible on real machines because of its memory consumption.

Recall that in our PFFE scheme, the $\mathbf{x}$-dependent ciphertext $\mathsf{ct}_{\mathbf{x}}$ is defined by

$$\mathsf{ct}_{\mathbf{x}} = \left(\mathbf{m}_{\mathbf{z},\mathbf{x}_L} \otimes \mathbf{s}_{\mathbf{x}_L}\right) \mathbf{B}_L + \mathbf{e}_{\mathbf{x}_L},$$

where

$$\mathbf{m}_{\mathbf{z},\mathbf{x}_L} := \mathbf{m}_{\mathbf{z}} \prod_{i \in [L]} \mathbf{M}_{i,x_i} = (1, \mathbf{z}', \mathbf{x}_L, \mathbf{t}),$$
$$\mathbf{s}_{\mathbf{x}_L} := \mathbf{s}_{\epsilon} \prod_{i \in [L]} \mathbf{S}_{i,x_i},$$

We introduce two optimizations to this ciphertext structure.

First, we simply remove the FHE encryption of the private input $\mathbf{z}'$ from $\mathbf{m}_{\mathbf{z},\mathbf{x}_L}$, and instead hardcode it into the circuit f being evaluated over BGG+ encodings $\mathbf{c}_{\mathbf{x}_L,\mathsf{att}}$. This is possible because in the use of PFFE for the iPRO construction, the obfuscator provides both the private input $\mathbf{z}$, namely the key K being obfuscated, and the circuit $f := C(\cdot,\cdot)$ being evaluated. This implies that the vectors $\mathbf{m}_{\mathbf{z}}$ and $\mathbf{m}_{\mathbf{z},\mathbf{x}_L}$ are replaced as follows:[24]

$$\mathbf{m}_{\mathbf{t}} := (1, \mathbf{0}_L, \mathbf{t}) \in \{0,1\}^{L+n+2},$$
$$\mathbf{m}_{\mathbf{t},\mathbf{x}_L} := (1, \mathbf{x}_L, \mathbf{t}) \in \{0,1\}^{L+n+2},$$

Second, instead of encoding $\mathbf{m}_{\mathbf{t},\mathbf{x}_L}$ into a single LWE sample $\mathbf{c}_{\mathbf{t},\mathbf{x}_L}$, we provide $L+1$ LWE samples, each of which depends on either $\mathbf{t}$ or a single bit of $\mathbf{x}_L$.

[24] The FHE decryption key $\mathbf{t}$ still remains in the ciphertext because the PFFE scheme itself does not hide the circuit f. In other words, we can hardcode only public data such as FHE encryption into the circuit f.

Concretely, the optimized construction defines the ciphertext $\mathsf{ct}_{\mathbf{x}}$ as follows:

$$\mathsf{ct}_{\mathbf{x}} = \begin{pmatrix} \mathbf{c}_{\mathbf{x}_L,0} := \left((1,\mathbf{t}) \otimes \mathbf{s}_{\mathbf{x}_L}\right) \mathbf{B}_{L,0} + \mathbf{e}_{\mathbf{x}_L,0}, \\ \left(\mathbf{c}_{\mathbf{x}_L,j} := \left((1,x_{L,j}) \otimes \mathbf{s}_{\mathbf{x}_L}\right) \mathbf{B}_j + \mathbf{e}_{\mathbf{x}_L,j}\right)_{j\in[L]} \end{pmatrix} \in \mathbb{Z}_q^{(n+1)(n+2+2L)m'_B},$$

where $x_{L,j} \in \{0,1\}$ is the j-th bit of $\mathbf{x} = \mathbf{x}_L$, $\mathbf{B}_{L,0} \in \mathbb{Z}_q^{(n+1)(n+2)\times(n+1)(n+2)m'_B}$ and $\mathbf{B}_j \in \mathbb{Z}_q^{2(n+1)\times 2(n+1)m'_B}$ are sampled with their trapdoors independently. This way, the number of rows of the matrices $\mathbf{B}_{L,0}$ and $\mathbf{B}_i$ is $O(n)$, which is constant with respect to L.

Accordingly, the definitions of the functional secret key $\mathsf{sk}_f := (\mathbf{K}_{\mathsf{att}}, \mathbf{K}_t, \mathbf{G})$ are also modified as follows.

$$\begin{aligned} \mathbf{K}_{\mathsf{att}} &:= \mathsf{diag}\begin{pmatrix} \mathbf{K}_{L,0} \leftarrow\!\$\, \mathbf{B}_{L,0,\gamma_B}^{-1}\left(\mathbf{u}_1 \otimes \left(\mathbf{A}_{\mathsf{att},0} - \mathbf{G}\right)\right), \\ \left(\mathbf{K}_{L,j} \leftarrow\!\$\, \mathbf{B}_{L,j,\gamma_B}^{-1}\left(\mathbf{u}_1 \otimes \mathbf{A}_{\mathsf{att},j} - \mathbf{u}_2 \otimes \mathbf{G}\right)\right)_{j\in[L]} \end{pmatrix}, \\ \mathbf{K}_t &\leftarrow\!\$\, \mathbf{B}_{L,0,\gamma_B}^{-1}\left(\mathbf{u}_1 \otimes \mathbf{A}_t - \begin{pmatrix} 0 \\ \mathbf{I}_{n+1} \end{pmatrix} \otimes \mathbf{G}\right), \\ \mathbf{K}_G &\leftarrow\!\$\, \mathbf{B}_{L,0,\gamma_B}^{-1}\left(\mathbf{u}_1 \otimes \mathbf{A}'_G\right), \end{aligned}$$

where $\mathbf{A}_{\mathsf{att}} \in \mathbb{Z}_q^{(n+1)\times(1+L)m}$ is parsed as $\left(\mathbf{A}_{\mathsf{att},0} \in \mathbb{Z}_q^{(n+1)\times m}, \ldots, \mathbf{A}_{\mathsf{att},L} \in \mathbb{Z}_q^{(n+1)\times m}\right)$.

To generate the above modified ciphertext $\mathsf{ct}_{\mathbf{x}}$ by matrix products in the iPRO scheme, we modify the GGH15 encoding $\mathbf{p}_\epsilon$ and the preimages provided in the obfuscation, namely $\{\mathbf{K}_{i,b}\}_{i\in[L],b\in\{0,1\}}$, as follows.

$$\begin{aligned} &\mathbf{p}_\epsilon := ((1,\mathbf{t}) \otimes \mathbf{s}_\epsilon)\, \mathbf{B}_{0,0} + \mathbf{e}_{\epsilon,0}, \\ \forall i \in [L], b \in \{0,1\} \quad &\mathbf{K}_{i,b} := \mathsf{diag}\begin{pmatrix} \mathbf{K}_{i,b,0} \leftarrow\!\$\, \mathbf{B}_{i-1,0,\gamma_B}^{-1}\left(\left(\mathbf{I}_{n+2} \otimes \mathbf{S}_{i,b}\right) \mathbf{B}_{i,0} + \mathbf{E}_{i,0}\right), \\ \left(\mathbf{K}_{i,b,j} \leftarrow\!\$\, \mathbf{B}_{i-1,j,\gamma_B}^{-1}\left(\left(\mathbf{I}_2 \otimes \mathbf{S}_{i,b}\right) \mathbf{B}_{i,j} + \mathbf{E}_{i,j}\right)\right)_{j\in[i-1]}, \\ \mathbf{K}_{i,b,i} \leftarrow\!\$\, \mathbf{B}_{i-1,0,\gamma_B}^{-1}\left(\left(\begin{pmatrix} 1, & b \\ \mathbf{0}, & \mathbf{0} \end{pmatrix} \otimes \mathbf{S}_{i,b}\right) \mathbf{B}_{i,i} + \mathbf{E}_{i,i}\right) \end{pmatrix} \\ &\qquad \in \mathbb{Z}_q^{(n+2i)(n+1)m'_B \times (n+2i+2)(n+1)m'_B} \end{aligned}$$

For every $i \in [L]$ and $\mathbf{x}_i \in \{0,1\}^i$, define

$$\mathbf{p}_{\mathbf{x}_i} := \begin{pmatrix} \mathbf{p}_{\mathbf{x}_i,0} := \left((1,\mathbf{t}) \otimes \mathbf{s}_{\mathbf{x}_i}\right) \mathbf{B}_{i,0} + \mathbf{e}_{\mathbf{x}_i,0}, \\ \left(\mathbf{p}_{\mathbf{x}_i,j} := \left((1,x_{i,j}) \otimes \mathbf{s}_{\mathbf{x}_i}\right) \mathbf{B}_{i,j} + \mathbf{e}_{\mathbf{x}_i,j}\right)_{j\in[i]} \end{pmatrix} \in \mathbb{Z}_q^{(n+1)(n+2+2i)m'_B},$$

which is identical to $\mathbf{p}_\epsilon$ for $i = 0$. Then, we can confirm that the products of preimages selected by bits of $\mathbf{x}_i$ yield the ciphertext $\mathsf{ct}_{\mathbf{x}}$, up to a small noise, as

follows:

$$\begin{aligned}
\mathbf{p}_\epsilon \prod_{i\in[L]} \mathbf{K}_{i,\mathbf{x}_i} &\approx \left(((1,\mathbf{t})\otimes \mathbf{s}_\epsilon)\,\mathbf{B}_{0,0}\right)\cdot \mathsf{diag}\left(\mathbf{K}_{1,x_1,0},\mathbf{K}_{1,x_1,1}\right)\cdot \prod_{i\in[2,L]} \mathbf{K}_{i,\mathbf{x}_i} \\
&\approx \mathbf{p}_{\mathbf{x}_1}\cdot \prod_{i\in[2,L]} \mathbf{K}_{i,\mathbf{x}_i}, \\
&\vdots \\
&\approx \mathbf{p}_{\mathbf{x}_{L-1}}\cdot \mathbf{K}_{L,\mathbf{x}_L}, \\
&\approx \mathbf{p}_{\mathbf{x}_L} \\
&\approx \mathsf{ct}_{\mathbf{x}}.
\end{aligned}$$

We can prove the security of this modified construction, and in particular the pseudorandomness of $\{\mathbf{p}_{\mathbf{x}_i}\}_{i\in[0,L],\mathbf{x}_i\in\{0,1\}^i}$, in the same way as in the security proof in Appendix E.2. The only difference is in the proof of the indistinguishability between Hyb_1 and Hyb_2. In the original proof, Eq. (30) simulates each vector $\mathbf{p}_{\mathbf{x}_i}$ by computing a linear combination of the $L'+n+1$ LWE samples $\mathbf{c}_{B,\mathbf{x}_i,j}$ provided by the all-product LWE challenger. In the modified construction, it suffices to simulate the same linear combination separately for each $\mathbf{p}_{\mathbf{x}_i,j}$.

G.2 Supporting Lookup-Table Evaluation in the PFFE Scheme

We explain how to support lookup-table (LUT) evaluation introduced in [76] in our PFFE scheme. The original construction is defined over polynomial rings, whereas we define it over integer rings.

Definition of LUT gates and circuits Let $T\in\mathbb{N}$ denote the size of the LUT, and define a LUT by a vector $\mathbf{t}\in\mathbb{Z}_q^T$. For each $k\in[0,T-1]$, $\mathbf{t}[k]$ returns the $(k+1)$-th element $t_{k+1}\in\mathbb{Z}_q$ of the LUT $\mathbf{t}$, where the index starts from 0 and ends at $T-1$.

We define a LUT evaluation gate. Each gate g is associated with a LUT $\mathbf{t}_g$ and has one input wire and one output wire. Concretely, on input $x\in[0,T-1]$, the gate outputs $\mathbf{t}_g[x]$.

Using this gate, we define bounded-integer arithmetic circuits equipped with LUT evaluation. Each wire w of the circuit carries an integer $x_w\in\mathbb{Z}$. Each gate g of the circuit performs one of the following operations:

- **Addition gate:** on input two integers $x_u,x_v\in\mathbb{Z}$, outputs $x_w := x_u+x_v \bmod q\in\mathbb{Z}$ if $x_w\in\mathbb{Z}_q$ and $\bot$ otherwise.
- **Multiplication gate:** on input two integers $x_u,x_v\in\mathbb{Z}$, outputs $x_w := x_u\cdot x_v \mod q\in\mathbb{Z}$ if $x_w\in\mathbb{Z}_q$ and $\bot$ otherwise.
- **LUT evaluation gate:** on input an integer $x_u\in\mathbb{Z}$, outputs $x_w := \mathbf{t}_g[x_u]$ if $x_u\in[0,T-1]$, and $\bot$ otherwise.

Let L be the input size of the circuit. For a circuit C, we define the admissible input space $\mathcal{I}_C\subseteq\{0,1\}^L$ as the set of inputs on which no gate outputs $\bot$.

LUT evaluation over BGG+ encodings To support LUT evaluation in the PFFE scheme, we introduce a method to evaluate LUT gates over BGG+ encodings. This is a variant of the method introduced in [76].

Algorithms 11 and 12 describe the evaluation of a LUT gate g associated with a LUT $\mathbf{t}_g$. The former is used for the $\mathsf{VMEvalC}$ algorithm, and the latter for the $\mathsf{VMEvalCX}$ algorithm; both are modified to support the evaluation of arithmetic circuits with LUT gates. These outputs satisfy the following equation:

$$(\mathbf{B}_{\mathsf{LUT}}, \mathbf{A}_u - x_u\mathbf{G})\,\mathbf{H}_w = \mathbf{A}_w - \mathbf{t}_g[x_u]\mathbf{G}.$$

This holds because it holds that

$$\begin{aligned}(\mathbf{B}_{\mathsf{LUT}}, \mathbf{A}_u - x_u\mathbf{G})\,\mathbf{H}_w &= \mathbf{B}_{\mathsf{LUT}}\mathbf{K}_{w,x_u+1} + (\mathbf{A}_u - x_u\mathbf{G})\,\mathbf{G}^{-1}(\mathbf{R}_{w,x_u+1}) \\ &= \big(\mathbf{A}_w - \mathbf{t}_g[x_u+1]\mathbf{G} - \mathbf{A}_u\mathbf{G}^{-1}(\mathbf{R}_{w,x_u+1}) + x_u\mathbf{R}_{w,x_u+1}\big) \\ &+ \big(\mathbf{A}_u\mathbf{G}^{-1}(\mathbf{R}_{w,x_u+1}) \ - x_u\mathbf{R}_{w,x_u+1}\big) \\ &= \mathbf{A}_w - \mathbf{t}_g[x_u]\mathbf{G}.\end{aligned}$$

- $\mathsf{EvalC}(\mathbf{A}, C) \to \mathbf{A}_C$: it takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times(L+1)m}$ and the circuit C and outputs a matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n\times Mm}$.
- $\mathsf{EvalCX}(\mathbf{A}, C, \mathbf{x}) \to \mathbf{H}_{C,x}$: it takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times(L+1)m}$, the circuit C, and the input $\mathbf{x}$, and outputs a matrix $\mathbf{H}_{C,x} \in \mathbb{Z}_q^{(L+1)m\times Mm}$.

The outputs of these algorithms satisfy the following properties:

- For a circuit C of depth dep, $\left\|\mathbf{H}_{C,x}\right\|_\infty \le m^{\mathcal{O}(\mathsf{dep})}$ holds.
- For any $\mathbf{A}$, C, and $\mathbf{x}$, it holds that

$$(\mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G})\mathbf{H}_{C,x} = \mathbf{A}_C - C(\mathbf{x}) \otimes \mathbf{G},$$

  where $\mathbf{A}_C \leftarrow \mathsf{EvalC}(\mathbf{A}, C)$ and $\mathbf{H}_{C,x} \leftarrow \mathsf{EvalCX}(\mathbf{A}, C, \mathbf{x})$.

Additionally, we adopt evaluation algorithms for an inner product proposed in [54]. The original algorithms are defined for an inner product between a hidden vector $\mathbf{t} \in \mathbb{Z}_q^t$ and the output of the Boolean circuit $C(\mathbf{x}) \in \{0,1\}^t$; ours in Lemma 7 extends the output of the circuit to a matrix $\mathbf{Y} \in \mathbb{Z}_q^{t\times M}$ by combining a technique introduced in Section 4.1 of [28].

Lemma 13 (Evaluation of BGG+ encodings for an arithmetic circuit with LUT gates). For all parameters $q, n, L, M, T \in \mathbb{N}$, $m, m'_B \ge n\lceil\log_2 q\rceil$, and a Gaussian parameter $\gamma B = \mathsf{poly}(\lambda)(m)$ satisfying the lower bound required for trapdoor preimage sampling, the following two deterministic algorithms are defined:

- $\mathsf{EvalC}(\mathbf{A}, C, \mathbf{B}^{-1}_{\mathsf{LUT},\gamma_B}) \to (\mathbf{A}_C, \mathcal{L}_C)$: it takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times(L+1)m}$, the bounded-integer arithmetic circuit C with LUT gates, and a trapdoor $\mathbf{B}^{-1}_{\mathsf{LUT},\gamma_B}$, and outputs a matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n\times Mm}$ and a set of matrices $\mathcal{L}_G$.

– $\mathsf{EvalCX}(\mathbf{A}, C, \mathbf{x}, \mathcal{L}_C) \to \mathbf{H}_{C,x}$: it takes as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times (L+1)m}$, the circuit C, the input $\mathbf{x}$, and a set of matrices $\mathcal{L}_C$, and outputs a matrix $\mathbf{H}_{C,x} \in \mathbb{Z}_q^{((L+1)m+m'_B) \times Mm}$.

The outputs of these algorithms satisfy the following properties:

– For a circuit C of depth dep, $\left\|\mathbf{H}_{C,x}\right\|_\infty \leq m^{\mathcal{O}(\mathsf{dep})}$ holds.
– For any $\mathbf{A}, C, \mathbf{x}$ in the admissible input space $\mathcal{I}_C$, and any pair of $(\mathbf{B}_{\mathsf{LUT}}, \mathbf{B}^{-1}_{\mathsf{LUT},\gamma_B})$, it holds that

$$(\mathbf{B}_{\mathsf{LUT}},\ \mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G})\mathbf{H}_{C,x} = \mathbf{A}_C - C(\mathbf{x}) \otimes \mathbf{G}, \tag{33}$$

where $(\mathbf{A}_C, \mathcal{L}_C) \leftarrow \mathsf{EvalC}(\mathbf{A}, C, \mathbf{B}^{-1}_{\mathsf{LUT},\gamma_B})$ and $\mathbf{H}_{C,x} \leftarrow \mathsf{EvalCX}(\mathbf{A}, C, \mathbf{x}, \mathcal{L}_C)$.

Proof. For each gate g, let w be the output wire of g, and let C_w be the subcircuit that has the same input wires as C and outputs the value carried by the wire w. We prove the following claim by induction over the topological order of the circuit: if u, v are the input wires of a gate g (or u is the single input wire when g is a LUT gate) and Eq. (33) holds for the subcircuits C_u and C_v, then it also holds for the subcircuit C_w.

For an input wire, the precondition of the induction is immediate. Consider an addition or multiplication gate g with input wires u, v and output wire w. By the standard BGG+ evaluation algorithms for arithmetic gates [16], EvalC produces $\mathbf{A}_w$ and EvalCX produces a local matrix $\mathbf{H}_{g,x}$ such that

$$(\mathbf{A}_u - x_u\mathbf{G},\ \mathbf{A}_v - x_v\mathbf{G})\,\mathbf{H}_{g,x} = \mathbf{A}_w - x_w\mathbf{G},$$

where $x_w = x_u + x_v \bmod q$ for an addition gate and $x_w = x_u x_v \bmod q$ for a multiplication gate. The output x_w is equivalent to the integer output of the gate g because $\mathbf{x} \in \mathcal{I}_C$.

By the induction hypothesis,

$$(\mathbf{B}_{\mathsf{LUT}},\ \mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G})\mathbf{H}_{C_u,x} = \mathbf{A}_u - x_u\mathbf{G},$$
$$(\mathbf{B}_{\mathsf{LUT}},\ \mathbf{A} - (1, \mathbf{x}) \otimes \mathbf{G})\mathbf{H}_{C_v,x} = \mathbf{A}_v - x_v\mathbf{G}.$$

Thus, by setting

$$\mathbf{H}_{C_w,x} := \left(\mathbf{H}_{C_u,x},\ \mathbf{H}_{C_v,x}\right)\mathbf{H}_{g,x},$$

we obtain Eq. (33) for C_w.

It remains to consider a LUT gate g with input wire u and output wire w. Since $\mathbf{x} \in \mathcal{I}_C$, the value x_u is in $[0, T-1]$. Algorithms 11 and 12 produce $\mathbf{A}_w$, store the helper matrices $\{\mathbf{R}_{w,k} \in \mathbb{Z}_q^{(n+1)\times m}\}_{k\in[T]}$ and $\{\mathbf{K}_{w,k} \in \mathbb{Z}_q^{m'_B \times m}\}_{k\in[T]}$ in $\mathcal{L}_C$, and output a local matrix $\mathbf{H}_{g,x}$ satisfying

$$(\mathbf{B}_{\mathsf{LUT}},\ \mathbf{A}_u - x_u\mathbf{G})\,\mathbf{H}_{g,x} = \mathbf{A}_w - \mathbf{t}_g[x_u]\mathbf{G}.$$

Write $\mathbf{H}_{C_u,x}$ according to the row partition induced by $(\mathbf{B}_{\mathsf{LUT}},\ \mathbf{A}-(1,\mathbf{x})\otimes\mathbf{G})$ as

$$\mathbf{H}_{C_u,x}=\begin{pmatrix}\mathbf{H}_{\mathsf{LUT},C_u,x}\\ \mathbf{H}_{\mathsf{in},C_u,x}\end{pmatrix}.$$

Combining this equation with the induction hypothesis for C_u, and setting

$$\mathbf{H}_{C_w,x}:=\begin{pmatrix}\mathbf{I}_{m'_B} & \mathbf{H}_{\mathsf{LUT},C_u,x}\\ \mathbf{0} & \mathbf{H}_{\mathsf{in},C_u,x}\end{pmatrix}\mathbf{H}_{g,x},$$

gives

$$(\mathbf{B}_{\mathsf{LUT}},\ \mathbf{A}-(1,\mathbf{x})\otimes\mathbf{G})\mathbf{H}_{C_w,x}=\mathbf{A}_w-\mathbf{t}_g[x_u]\mathbf{G}=\mathbf{A}_w-C_w(\mathbf{x})\mathbf{G}.$$

Therefore Eq. (33) holds for every output wire of C.

The local matrix $\mathbf{H}_{g,x}$ for a LUT gate has norm bounded by $\mathsf{poly}(\lambda)(m)$, since $\mathbf{G}^{-1}(\mathbf{R})$ is bounded by the gadget decomposition bound, and the preimage $\mathbf{K}$ has norm bounded by the Gaussian parameter γ_B up to a polynomial factor (Lemma 2). The local matrices for addition and multiplication gates are also bounded by $\mathsf{poly}(\lambda)(m)$ by the standard BGG+ evaluation algorithms for arithmetic gates. Since $\mathbf{H}_{C,x}$ is obtained by multiplying such local matrices along paths of length at most the depth dep, and by concatenating matrices for parallel subcircuits, we obtain

$$\left\|\mathbf{H}_{C,x}\right\|_\infty\le m^{\mathcal{O}(\mathsf{dep})}.$$

Modification to the PFFE scheme To support the evaluation of LUT gates in the PFFE scheme modified in Appendix G.1, we add the following preimage to the functional secret key sk_f.

$$\mathbf{K}_{\mathsf{LUT}}\leftarrow\!\!\$\ \mathbf{B}^{-1}_{L,0,\gamma_B}(\mathbf{u}_1\otimes\mathbf{B}_{\mathsf{LUT}})$$

This derives from $\mathbf{c}_{\mathbf{x}_L,0}$

$$\begin{aligned}\mathbf{b}_{\mathbf{x}_L,\mathsf{LUT}}&:=\mathbf{c}_{\mathbf{x}_L,0}\mathbf{K}_{\mathsf{LUT}}\\ &\approx\mathbf{s}_{\mathbf{x}_L}\mathbf{B}_{\mathsf{LUT}}.\end{aligned}$$

Given $\mathbf{b}_{\mathbf{x}_L,\mathsf{LUT}}$ and the BGG+ encodings

$$\begin{aligned}\mathbf{c}_{\mathbf{x}_L,\mathsf{att}}&\approx\mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_{\mathsf{att}}-(1,\mathbf{x}_L)\otimes\mathbf{G}\right),\\ \mathbf{c}_{\mathbf{x}_L,t}&\approx\mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_t-\mathbf{t}\otimes\mathbf{G}\right),\end{aligned}$$

this enables the key-homomorphic evaluation for a vector-matrix circuit with LUT gates.

$$\begin{aligned}&(\mathbf{B}_{\mathsf{LUT}},\ \mathbf{A}-(1,\mathbf{x})\otimes\mathbf{G},\ \mathbf{A}_t-\mathbf{t}\otimes\mathbf{G})\,\mathbf{H}_{G,x}\\ &\quad=\mathbf{A}_G-(\mathbf{t}\mathbf{Y})\otimes\mathbf{G},\end{aligned}$$

where $\mathbf{Y} := G(\mathbf{x})$, and $\mathbf{H}_{G,x}$ is a low-norm matrix deterministically derived from

$$\mathbf{H}_{G,x} \leftarrow \mathsf{EvalCX}(\mathbf{A}_{\mathsf{att}}, G, \mathbf{x}, \mathcal{L}_C).$$

To claim the security of the modified PFFE scheme, we need to ensure that the additional preimages introduced for LUT evaluation preserve the pseudorandomness of the ciphertexts. We prove this under a generalized version of the all-product LWE assumption formalized below. Compared with the all-product LWE assumption (Assumption 4), this generalized version allows a public function $f : \{0,1\}^L \to \mathbb{Z}_q^M$ to determine, for each input vector $\mathbf{x}_L \in \{0,1\}^L$, the constants to be tensored with the gadget matrix, rather than fixing them to the entries of $\mathbf{x}_L$ themselves.

Assumption 5 (Generalized All-Product LWE Assumption). Let n, $m_A \geq (n+1)\lceil \log_2 q \rceil$, $\{m_{B,i}\}_{i\in[0,L]}$, q, $L \in \mathbb{N}$ be parameters, and $\lambda \in \mathbb{N}$ be a security parameter. Let $M \in \mathbb{N}$ be an output dimension, and let $f : \{0,1\}^L \to \mathbb{Z}_q^M$ be a public function of polynomial size. Let σ_A and $\{\sigma_{B,i}\}_{i\in[0,L]}$ be Gaussian parameters.

We define the following two distributions:

$$\mathcal{D}_{\mathsf{GAPLWE},0} := \left(\begin{array}{l} \mathbf{A}, \{\mathbf{B}_i\}_{i\in[0,L]}, \\ \left\{ \mathbf{c}_{A,\mathbf{x}_L} := \mathbf{s}_{\mathbf{x}_L} \left(\mathbf{A} - f(\mathbf{x}_L) \otimes \mathbf{G}\right) + \mathbf{e}_{A,\mathbf{x}_L} \right\}_{\mathbf{x}_L \in \{0,1\}^L} \\ \left\{ \mathbf{c}_{B,\mathbf{x}_i} := \mathbf{s}_{\mathbf{x}_i} \mathbf{B}_i + \mathbf{e}_{B,\mathbf{x}_i} \right\}_{i\in[0,L],\, \mathbf{x}_i \in \{0,1\}^i} \end{array} \right),$$

$$\mathcal{D}_{\mathsf{GAPLWE},1} := \left(\begin{array}{l} \mathbf{A}, \{\mathbf{B}_i\}_{i\in[0,L]}, \\ \left\{ \mathbf{c}^\star_{A,\mathbf{x}_L} \leftarrow\$ \mathbb{Z}_q^{Mm_A} \right\}_{\mathbf{x}_L \in \{0,1\}^L} \\ \left\{ \mathbf{c}^\star_{B,\mathbf{x}_i} \leftarrow\$ \mathbb{Z}_q^{m_{B,i}} \right\}_{i\in[0,L],\, \mathbf{x}_i \in \{0,1\}^i} \end{array} \right),$$

where

- $\mathbf{A} \leftarrow\$ \mathbb{Z}_q^{(n+1)\times Mm_A}$,
- $\mathbf{B}_i \leftarrow\$ \mathbb{Z}_q^{(n+1)\times m_{B,i}}$ for every $i \in [0,L]$,
- $\mathbf{e}_{A,\mathbf{x}_L} \leftarrow\$ \mathcal{D}_{\mathbb{Z},\sigma_A}^{Mm_A}$ for every $\mathbf{x}_L \in \{0,1\}^L$,
- $\mathbf{e}_{B,\mathbf{x}_i} \leftarrow\$ \mathcal{D}_{\mathbb{Z},\sigma_{B,i}}^{m_{B,i}}$ for every $i \in [0,L]$ and $\mathbf{x}_i \in \{0,1\}^i$,
- $\bar{\mathbf{s}}_\epsilon \leftarrow\$ \{0,\pm 1\}^n$,
- $\bar{\mathbf{S}}_{i,b} \leftarrow\$ \{0,\pm 1\}^{n\times n}$ for every $i \in [L]$ and $b \in \{0,1\}$,
- $\bar{\mathbf{s}}_{\mathbf{x}_i} := \bar{\mathbf{s}}_\epsilon \prod_{j\in[i]} \bar{\mathbf{S}}_{j,x_j}$ and $\mathbf{s}_{\mathbf{x}_i} := (\bar{\mathbf{s}}_{\mathbf{x}_i}, -1)$ for every $i \in [0,L]$ and $\mathbf{x}_i \in \{0,1\}^i$.

For a function $\kappa = \kappa(\lambda)$, the generalized all-product LWE assumption is said to hold if for every non-uniform adversary $\mathcal{A}$ such that $\mathrm{Size}(\mathcal{A}) \leq \mathsf{poly}(\kappa)$, it holds that

$$\left|\Pr\left[\mathcal{A}\left(\mathcal{D}_{\mathsf{GAPLWE},0}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathcal{D}_{\mathsf{GAPLWE},1}\right) = 1\right]\right| \leq \mathsf{negl}(\kappa).$$

The following lemma states that the modified PFFE scheme is secure using the generalized all-product LWE assumption.

Lemma 14 (Security of the modified PFFE scheme). Let $\kappa = \lambda^{L^2 \log \lambda}$. Assume that the following holds:

- the private-coin binding non-uniform κ-evasive LWE assumption (Assumption 2).
- the generalized all-product LWE assumption (Assumption 5).
- the subexponential semantic security of the FHE scheme for the security parameter Λ (Definition 1).
- the subexponential security of the PRF scheme denoted by $\mathsf{PRF}_B : \{0,1\}^\lambda \times \{0,1\}^\lambda \to [-q/4 + B, q/4 - B]^{1 \times \ell}$ for the security parameter Λ.

Then, the PFFE scheme modified in Appendix G is secure in the sense of Definition 7.

Proof. **Proof sketch.** First, it suffices to prove the following indistinguishability; once this is established, the rest of the security proof is almost identical to the

proof in Appendix E.2.

$$\left(\begin{array}{l}\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t, \mathbf{A}'_G, \mathbf{B}_{\mathsf{LUT}} \\ \left\{\mathbf{c}_{\mathbf{x}_L,\mathsf{att}} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_{\mathsf{att}} - \mathbf{m}_{\mathbf{x}_L,\mathsf{att}} \otimes \mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att}}\right\}, \\ \left\{\mathbf{c}_{\mathbf{x}_L,t} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_t - \mathbf{t} \otimes \mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,t}\right\}, \\ \left\{\mathbf{d}_{\mathbf{x}_L} := \mathbf{s}_{\mathbf{x}_L}\mathbf{A}'_G + \mathbf{e}_{\mathbf{x}_L,d}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{b}_{\mathbf{x}_L,\mathsf{LUT}} := \mathbf{s}_{\mathbf{x}_L}\mathbf{B}_{\mathsf{LUT}} + \mathbf{e}_{\mathbf{x}_L,\mathsf{LUT}}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{p}_{\mathbf{x}_i} := \begin{pmatrix}\mathbf{p}_{\mathbf{x}_i,0} := \left((1,\mathbf{t}) \otimes \mathbf{s}_{\mathbf{x}_i}\right)\mathbf{B}_{i,0} + \mathbf{e}_{\mathbf{x}_i,0}, \\ \left(\mathbf{p}_{\mathbf{x}_i,j} := \left((1,x_{i,j}) \otimes \mathbf{s}_{\mathbf{x}_i}\right)\mathbf{B}_{i,j} + \mathbf{e}_{\mathbf{x}_i,j}\right)_{j\in[i]},\end{pmatrix}\right\}_{i\in[0,L],\,\mathbf{x}_i\in\{0,1\}^i}, \\ \mathsf{ct}_{\mathbf{z}} := \mathsf{FHE.Enc}(\mathsf{pk},\mathbf{z}), f, \mathsf{aux}_{\mathbf{z}}, \mathcal{L}_C\end{array}\right)$$

$$\approx_c \left(\begin{array}{l}\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t, \mathbf{A}'_G, \mathbf{B}_{\mathsf{LUT}} \\ \left\{\mathbf{c}^\star_{\mathbf{x}_L,\mathsf{att}} \leftarrow\$\, \mathbb{Z}_q^{(1+L)m}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{c}^\star_{\mathbf{x}_L,t} \leftarrow\$\, \mathbb{Z}_q^{(n+1)m}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{d}^\star_{\mathbf{x}_L} \leftarrow\$\, \mathbb{Z}_q^{M}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{b}^\star_{\mathbf{x}_L,\mathsf{LUT}} \leftarrow\$\, \mathbb{Z}_q^{m'_B}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{p}^\star_{\mathbf{x}_i} := \begin{pmatrix}\mathbf{p}^\star_{\mathbf{x}_i,0} \leftarrow\$\, \mathbb{Z}_q^{(n+1)(n+2)m'_B}, \\ \left(\mathbf{p}^\star_{\mathbf{x}_i,j} \leftarrow\$\, \mathbb{Z}_q^{2(n+1)m'_B}\right)_{j\in[i]},\end{pmatrix}\right\}_{i\in[0,L],\,\mathbf{x}_i\in\{0,1\}^i}, \\ \mathsf{ct}^\star_{\mathbf{z}} := \mathsf{FHE.Enc}(\mathsf{pk},\mathbf{0}), f, \mathsf{aux}_{\mathbf{z}}, \mathcal{L}_C\end{array}\right). \tag{34}$$

The set of matrices $\mathcal{L}_C$, prepared for evaluating LUT gates, depends on the trapdoor $\mathbf{B}^{-1}_{\mathsf{LUT},\gamma_B}$. Therefore, to prove the pseudorandomness of $\mathbf{b}_{\mathbf{x}_L,\mathsf{LUT}}$ in the presence of $\mathcal{L}_C$, we invoke Lemma 4.

Without loss of generality, we consider the case where the circuit contains only one LUT gate g with the input wire u and the output wire w. Then, we have

$$\mathcal{L}_C = \left\{\mathbf{A}_w, \{\mathbf{R}_{w,k}\}_{k\in[T]}, \{\mathbf{K}_{w,k}\}_{k\in[T]}\right\}.$$

By invoking Lemma 4 with respect to $\mathbf{B}_{\mathsf{LUT}}$, $\widetilde{\mathbf{S}} := \left(\mathbf{s}_{\mathbf{x}_L}\right)_{\mathbf{x}_L\in\{0,1\}^L} \in \mathbb{Z}_q^{2^L\times n}$, and $\widetilde{\mathbf{P}} := \left(\mathbf{A}_w - \mathbf{t}_g[k']\mathbf{G} - \mathbf{A}_u\mathbf{G}^{-1}(\mathbf{R}_{w,k}) + k'\mathbf{R}_{w,k}\right)$, we can prove the indistinguishability in Eq. (34) by showing that the following two distributions are

indistinguishable:

$$\left(\begin{array}{l}\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t, \mathbf{A}'_G, \mathbf{B}_{\mathsf{LUT}} \\ \left\{\mathbf{c}_{\mathbf{x}_L,\mathsf{att}} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_{\mathsf{att}} - \mathbf{m}_{\mathbf{x}_L,\mathsf{att}} \otimes \mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att}}\right\}, \\ \left\{\mathbf{c}_{\mathbf{x}_L,t} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_t - \mathbf{t} \otimes \mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,t}\right\}, \\ \left\{\mathbf{d}_{\mathbf{x}_L} := \mathbf{s}_{\mathbf{x}_L}\mathbf{A}'_G + \mathbf{e}_{\mathbf{x}_L,d}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{b}_{\mathbf{x}_L,\mathsf{LUT}} := \mathbf{s}_{\mathbf{x}_L}\mathbf{B}_{\mathsf{LUT}} + \mathbf{e}_{\mathbf{x}_L,\mathsf{LUT}}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{p}_{\mathbf{x}_i} := \left(\begin{array}{l}\mathbf{p}_{\mathbf{x}_i,0} := \left((1,\mathbf{t}) \otimes \mathbf{s}_{\mathbf{x}_i}\right)\mathbf{B}_{i,0} + \mathbf{e}_{\mathbf{x}_i,0}, \\ \left(\mathbf{p}_{\mathbf{x}_i,j} := \left((1,x_{i,j}) \otimes \mathbf{s}_{\mathbf{x}_i}\right)\mathbf{B}_{i,j} + \mathbf{e}_{\mathbf{x}_i,j}\right)_{j\in[i]},\end{array}\right)\right\}_{i\in[0,L],\mathbf{x}_i\in\{0,1\}^i}, \\ \mathsf{ct}_{\mathbf{z}} := \mathsf{FHE.Enc}(\mathsf{pk},\mathbf{z}), f, \mathsf{aux}_{\mathbf{z}}, \\ \left\{\mathbf{h}_{\mathbf{x}_L,k} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - \mathbf{t}_g[k']\mathbf{G} - \mathbf{A}_u\mathbf{G}^{-1}(\mathbf{R}_{w,k}) + k'\mathbf{R}_{w,k}\right) + \mathbf{e}_{\mathbf{x}_L,h,k}\right\}_{\mathbf{x}_L\in\{0,1\}^L,\,k\in[T]}, \\ \mathbf{A}_w, \left\{\mathbf{R}_{w,k}\right\}_{k\in[T]}\end{array}\right)$$

$$\approx_c \left(\begin{array}{l}\mathbf{B}_L, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t, \mathbf{A}'_G, \mathbf{B}_{\mathsf{LUT}} \\ \left\{\mathbf{c}^\star_{\mathbf{x}_L,\mathsf{att}} \leftarrow\$\, \mathbb{Z}_q^{(1+L)m}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{c}^\star_{\mathbf{x}_L,t} \leftarrow\$\, \mathbb{Z}_q^{(n+1)m}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{d}^\star_{\mathbf{x}_L} \leftarrow\$\, \mathbb{Z}_q^{M}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{b}^\star_{\mathbf{x}_L,\mathsf{LUT}} \leftarrow\$\, \mathbb{Z}_q^{m'_B}\right\}_{\mathbf{x}_L\in\{0,1\}^L}, \\ \left\{\mathbf{p}^\star_{\mathbf{x}_i} := \left(\begin{array}{l}\mathbf{p}^\star_{\mathbf{x}_i,0} \leftarrow\$\, \mathbb{Z}_q^{(n+1)(n+2)m'_B}, \\ \left(\mathbf{p}^\star_{\mathbf{x}_i,j} \leftarrow\$\, \mathbb{Z}_q^{2(n+1)m'_B}\right)_{j\in[i]},\end{array}\right)\right\}_{i\in[0,L],\mathbf{x}_i\in\{0,1\}^i}, \\ \mathsf{ct}^\star_{\mathbf{z}} := \mathsf{FHE.Enc}(\mathsf{pk},\mathbf{0}), f, \mathsf{aux}_{\mathbf{z}}, \\ \left\{\mathbf{h}_{\mathbf{x}_L,k} \leftarrow\$\, \mathbb{Z}_q^{m}\right\}_{\mathbf{x}_L\in\{0,1\}^L,\,k\in[T]}, \\ \mathbf{A}_w, \left\{\mathbf{R}_{w,k}\right\}_{k\in[T]}\end{array}\right).$$

Let $\mathbf{c}_{\mathbf{x}_L,\mathsf{att},u} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_u - x_u\mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,\mathsf{att},u}$ be the BGG+ encoding of the value x_u carried by the input wire u, which is deterministically derived from $\mathbf{c}_{\mathbf{x}_L,\mathsf{att}}$. Let $k^\star = x_u + 1$. Then it holds that

$$\begin{aligned}\mathbf{h}_{\mathbf{x}_L,k^\star} &:= \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - \mathbf{t}_g[x_u]\mathbf{G} - \mathbf{A}_u\mathbf{G}^{-1}(\mathbf{R}_{w,k^\star}) + x_u\mathbf{R}_{w,k^\star}\right) + \mathbf{e}_{\mathbf{x}_L,h,k^\star} \\ &\overset{s}{\approx} \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - \mathbf{t}_g[x_u]\mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,h,k^\star} - \mathbf{c}_{\mathbf{x}_L,\mathsf{att},u}\mathbf{G}^{-1}(\mathbf{R}_{w,k^\star}),\end{aligned}$$

where the statistical indistinguishability follows from the noise flooding. Therefore, if $\mathbf{c}_{\mathbf{x}_L,\mathsf{att},w} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - \mathbf{t}_g[x_u]\mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,h,k^\star}$ is pseudorandom, then $\mathbf{h}_{\mathbf{x}_L,k^\star}$ is also pseudorandom.

For $k \neq k^\star$, we have

$$\begin{aligned}
\mathbf{h}_{\mathbf{x}_L,k} &:= \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - \mathbf{t}_g[k-1]\mathbf{G} - \mathbf{A}_u\mathbf{G}^{-1}(\mathbf{R}_{w,k}) + (k-1)\mathbf{R}_{w,k}\right) + \mathbf{e}_{\mathbf{x}_L,h,k} \\
&\overset{\mathrm{s}}{\approx} \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{A}_w - \mathbf{t}_g[k-1]\mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,h,k} - \mathbf{c}_{\mathbf{x}_L,\mathsf{att},u}\mathbf{G}^{-1}(\mathbf{R}_{w,k^\star}) + (k-1-x_u)\mathbf{s}_{\mathbf{x}_L}\mathbf{R}_{w,k} \\
&\overset{\mathrm{s}}{\approx} \mathbf{c}_{\mathbf{x}_L,\mathsf{att},w} - \mathbf{c}_{\mathbf{x}_L,\mathsf{att},u}\mathbf{G}^{-1}(\mathbf{R}_{w,k^\star}) \\
&\quad + \left(\mathbf{s}_{\mathbf{x}_L}\left(\mathbf{R}'_{w,k} - \left(\mathbf{t}_g[k-1] - \mathbf{t}_g[x_u]\right)\mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,h,k}\right),
\end{aligned}$$

where $\mathbf{R}'_{w,k} := (k-1-x_u)\mathbf{R}_{w,k}$. Hence, if $\mathbf{r}_{\mathbf{x}_L,k} := \mathbf{s}_{\mathbf{x}_L}\left(\mathbf{R}'_{w,k} - \left(\mathbf{t}_g[k-1] - \mathbf{t}_g[x_u]\right)\mathbf{G}\right) + \mathbf{e}_{\mathbf{x}_L,h,k}$ is pseudorandom, then $\mathbf{h}_{\mathbf{x}_L,k}$ is also pseudorandom.

We can use the generalized all-product LWE assumption to prove the pseudorandomness of $\mathbf{c}_{\mathbf{x}_L,\mathsf{att},w}$ and $\mathbf{r}_{\mathbf{x}_L,k}$. Specifically, for every $\mathbf{x}_L \in \{0,1\}^L$, define a function $f_{G,\mathbf{x}_L} : \{0,1\}^L \to \mathbb{Z}_q^T$ as follows:[25]

$$\begin{aligned}
&f_{G,\mathbf{x}_L}(\mathbf{x}_L) := (y_k)_{k\in[T]}, \\
&y_k := \begin{cases} \mathbf{t}_g[x_u] & \text{if } k = x_u + 1, \\ \mathbf{t}_g[k-1] - \mathbf{t}_g[x_u] & \text{otherwise} \end{cases}.
\end{aligned}$$

Then, the encodings $\mathbf{c}_{A,\mathbf{x}_L}$ provided by the challenger of the generalized all-product LWE assumption for this function can be used to simulate $\mathbf{c}_{\mathbf{x}_L,\mathsf{att},w}$ and $\mathbf{r}_{\mathbf{x}_L,k}$ for every $k \in [T]$. Therefore, $\mathbf{h}_{\mathbf{x}_L,k}$ remains pseudorandom under this assumption. Lemma 4 then implies that $\mathbf{b}_{\mathbf{x}_L,\mathsf{LUT}}$ is also pseudorandom in the presence of $\mathcal{L}_C$. □

G.3 Low-Depth Circuit Design for Arithmetic Operations and Gadget Decomposition

In this subsection, we describe a design of modulo-q arithmetic and gadget decomposition circuits used in the implementation. This enables lower-depth circuits for the ring-GSW FHE simulation than the ones in [76].

Specifically, we use fast base conversion in the Chinese remainder theorem (CRT) representation introduced by Halevi, Polyakov, and Shoup [56]. This is an adaptation, for BGG+ encodings, of the FHE construction proposed by Boneh and Kim [19], which improves the efficiency of homomorphic modular reduction under moduli different from the ciphertext modulus.

Notations and representations. Let q be the modulus of the BGG+ encodings and of the FHE ciphertexts. For this simplified presentation, we treat q as a single modulus, rather than decomposing it into several coprime factors.

[25] This definition implies that the function f depends on the circuit G and the input vector $\mathbf{x}_L$.

One may assume that q is prime, although the construction only requires that q is coprime to the auxiliary moduli introduced below.

Let

$$p := \prod_{j=1}^{s} p_j,$$

where $p_1, \ldots, p_s$ are pairwise coprime integers. We choose each p_j to have a small bit-width k, for example $k = 8$, so that arithmetic modulo p_j can be implemented by a small LUT. The product p must be large enough to contain the intermediate integer values before reduction modulo q. For example, for one multiplication of two canonical representatives in $[0, q)$, it is natural to require

$$q^2 < p.$$

Given $x \in \mathbb{Z}_q$, let $[x]_q \in [0, q)$ be its canonical representative. We represent $[x]_q$ by its residues modulo $p_1, \ldots, p_s$:

$$\left([[x]_q]_{p_1}, \ldots, [[x]_q]_{p_s}\right).$$

Over BGG+ encodings under the secret $\mathbf{s}$, this means that we store the following vector of encodings:[26]

$$\begin{aligned}\mathbf{C}_{[x]_q} &:= \left(\mathbf{c}_{[[x]_q]_{p_1}}, \ldots, \mathbf{c}_{[[x]_q]_{p_s}}\right) \\ &:= \left(\mathbf{s}\left(\mathbf{A}_{[[x]_q]_{p_1}} - [[x]_q]_{p_1}\mathbf{G}\right) + \mathbf{e}_{[[x]_q]_{p_1}}, \ldots, \mathbf{s}\left(\mathbf{A}_{[[x]_q]_{p_s}} - [[x]_q]_{p_s}\mathbf{G}\right) + \mathbf{e}_{[[x]_q]_{p_s}}\right) \in \mathbb{Z}_q^{n \times ms}.\end{aligned}$$

Thus, the encoding modulus is still q, but the plaintext values inside the encodings are small residues modulo p_j. The large modulus q is used only as the ambient modulus of BGG+ encodings, while the nonlinear plaintext arithmetic is performed over the small moduli p_j.

Small-modulus arithmetic by LUTs. Assume that each p_j is represented with k bits. Addition modulo p_j can be implemented by a LUT that takes

$$x_1 + x_2 \in [0, 2^{k+1})$$

and outputs

$$(x_1 + x_2) \bmod p_j \in [0, p_j).$$

Similarly, multiplication modulo p_j can be implemented by a LUT that takes

$$x_1 x_2 \in [0, 2^{2k})$$

and outputs

$$x_1 x_2 \bmod p_j \in [0, p_j).$$

[26] Here, we omit $\mathbf{x}_L$ from the input-dependent secret $\mathbf{s}_{\mathbf{x}_L}$ for simplicity.

Since k is a small constant, these LUTs have constant size from the perspective of $\log_2 q$.

Therefore, a product in the p-RNS representation can be computed limbwise by applying, for every $j \in [s]$, the LUT

$$([[x]_q]_{p_j}, [[y]_q]_{p_j}) \mapsto [[x]_q[y]_q]_{p_j}.$$

If $X = [x]_q$ and $Y = [y]_q$ are the integer representatives and $XY < p$, the result is a valid CRT representation of XY modulo p:

$$([XY]_{p_1}, \ldots, [XY]_{p_s}).$$

Modulo-q reduction within the p-RNS representation. We next discuss how to simulate reduction modulo q while staying in the same p-RNS representation. That is, given the residues of an unreduced product modulo the auxiliary moduli $p_1, \ldots, p_s$, our goal is to convert them into residues of a bounded representative that is equivalent to the product modulo q, without first reconstructing the large integer product by a deep Boolean circuit.

At first glance, one might wonder why we need to reduce this value modulo q. After all, BGG+ encodings are defined over the encoding modulus q, and therefore adding a multiple of q does not change the encoded plaintext. In particular, a value of the form

$$[xy]_q + qe$$

is semantically equivalent to $[xy]_q$ over $\mathbb{Z}_q$, as long as the additional term qe remains small enough for correctness.

The real issue is not the presence of such a qe term itself. The issue is that the auxiliary p-RNS representation is used as an integer representation, not merely as a representation modulo q. In order for the residues modulo $p_1, \ldots, p_s$ to determine the intended integer uniquely, the represented value must remain within a prescribed range, typically strictly below $p = \prod_j p_j$.

If we keep multiplying unreduced integer representatives, the represented integer grows rapidly:

$$x, y = \mathcal{O}(q) \quad \Longrightarrow \quad xy = \mathcal{O}(q^2),$$

and after repeated multiplications this growth becomes exponential in the multiplicative depth. Once the represented value exceeds the range covered by p, the residues

$$([z]_{p_1}, \ldots, [z]_{p_s})$$

no longer uniquely identify the intended integer $z \in \mathbb{Z}$.

Therefore, the purpose of the modular-reduction step is to restore the representation invariant: after each multiplication, the circuit converts the p-RNS representation of the unreduced product xy into a p-RNS representation of a bounded representative

$$[xy]_q + qe,$$

which is equivalent to $[xy]_q$ modulo q, while still lying within the range where the p-RNS representation uniquely determines the underlying integer. This is what allows multiplication to be iterated without requiring p to grow exponentially with the multiplicative depth.

Fast modular reduction from p to q. We implement reduction modulo q via fast modular reduction as follows. Let X be an integer represented by its residues

$$[X]_{p_1}, \ldots, [X]_{p_s}.$$

Define

$$\hat{p}_l := \frac{p}{p_l}.$$

For each $l \in [s]$, define

$$y_l := \left[[X]_{p_l} \cdot \hat{p}_l^{-1} \right]_{p_l}.$$

By the CRT reconstruction formula over the integers, there exists an integer v such that

$$\sum_{l=1}^{s} y_l \hat{p}_l = [X]_p + pv.$$

Dividing both sides by p, we obtain

$$\sum_{l=1}^{s} \frac{y_l}{p_l} = \frac{[X]_p}{p} + v.$$

Therefore, if $[X]_p$ is sufficiently smaller than p, we can recover v as

$$v := \left\lfloor \sum_{l=1}^{s} \frac{y_l}{p_l} \right\rfloor.$$

The fast base-conversion step avoids reconstructing $[X]_p$ as a large integer. Instead, for each output limb p_j, it computes

$$r_j := \left[\sum_{l=1}^{s} y_l \cdot [[\hat{p}_l]_q]_{p_j} - v \cdot [[p]_q]_{p_j} \right]_{p_j}. \tag{35}$$

This value satisfies

$$r_j = [[X]_q + qe]_{p_j}$$

for some integer e. The additional term qe is harmless: BGG+ encodings are defined over $\mathbb{Z}_q$, and hence $[X]_q + qe$ represents the same value as $[X]_q$ modulo q.

To compute v over BGG+ encodings, we represent the real numbers

$$\frac{y_l}{p_l} \in [0, 1)$$

as fixed-point integers with sufficient precision. We then use LUTs to convert between the integer representation modulo p_l and the fixed-point representation used for the quotient computation. The precision is chosen so that the rounding step returns the correct quotient under the prescribed range bound for X.

Thus, the reduction from p to q consists of the limbwise computation of y_l, fixed-point quotient estimation for v, and limbwise computation of the corrected residues r_j. Each of these steps uses only small-modulus LUTs and additions over RNS limbs.

Depth of the arithmetic circuit. The important feature of this construction is how the dependence on $\log_2 q$ appears. The number of limbs s grows with the size of p, and hence with $\log_2 q$. For example, if each p_j has k bits and we require $p > q^2$, then

$$s \approx \frac{2 \log_2 q}{k}.$$

Thus, larger q increases the number of limbs and therefore the circuit size.

However, if we ignore the depth of additions over these limbs, it does not increase the nonlinear depth. The nonlinear part consists of a constant number of LUT layers as follows:

- small-modulus multiplication modulo p_j
- conversion to the CRT coefficients y_l
- fixed-point conversion and quotient estimation
- final modular correction modulo each p_j

Therefore, under this convention, the arithmetic circuit has constant nonlinear depth independent of $\log_2 q$. Equivalently, the cost of increasing q is shifted from multiplicative depth to RNS width.

Gadget decomposition compatible with the arithmetic circuit. Ring-GSW multiplication and related FHE operations require not only multiplication modulo q, but also gadget decomposition. Therefore, we need a gadget-decomposition circuit whose output is compatible with the same p-RNS representation.

Fortunately, the above representation naturally gives such a gadget. Define

$$\mathbf{g} := ([\hat{p}_1]_q, \ldots, [\hat{p}_s]_q, -[p]_q) \in \mathbb{Z}_q^{s+1}. \tag{36}$$

For a value $x' = [x]_q + qe$ represented by its residues modulo $p_1, \ldots, p_s$, define

$$[y_j]_{p_j} := \left[[x']_{p_j} \cdot \hat{p}_j^{-1} \right]_{p_j},$$

and

$$w := \left\lfloor \sum_{j=1}^{s} \frac{[y_j]_{p_j}}{p_j} \right\rfloor.$$

Then we can obtain a gadget decomposition of $[x]_q$ by outputting the vector of residues

$$\mathbf{g}^{-1}([x]_q) := ([y_1]_{p_1}, \ldots, [y_s]_{p_s}, w)\,. \tag{37}$$

By the same CRT reconstruction identity, we have

$$\begin{aligned}\mathbf{g}\left(\mathbf{g}^{-1}([x]_q)\right)^T &= \sum_{j=1}^{s} [\hat{p}_j]_q [y_j]_{p_j} - [p]_q w \pmod q \\ &= [x]_q \pmod q. \end{aligned} \tag{38}$$

Thus, $\mathbf{g}^{-1}([x]_q)$ is a gadget decomposition of $[x]_q$ with respect to the fixed gadget vector $\mathbf{g}$. Its coordinates are small: the first s coordinates are bounded by the small moduli p_j, and the last coordinate w lies in $[0, s)$.

The decomposition circuit uses the same operations as the fast base-conversion circuit. Therefore, it has the same constant nonlinear-depth property, again ignoring the depth of additions over RNS limbs.

Simulating ring-GSW multiplication. We can now combine the arithmetic circuit and the gadget-decomposition circuit to simulate ring-GSW multiplication over BGG+ encodings. At a high level, ring-GSW multiplication consists of repeated modular multiplications and gadget decompositions. In the usual FHE implementation, the ciphertext entries are elements modulo q, and gadget decomposition is applied to these entries before performing external products. In our setting, ciphertext entries are encoded by BGG+ encodings, so both the modular arithmetic and the gadget decomposition must themselves be evaluated over encodings.

The arithmetic circuit provides the operation

$$(x, y) \mapsto xy \bmod q$$

over encoded inputs, while keeping the result in the p-RNS representation. The gadget-decomposition circuit provides

$$x \mapsto \mathbf{g}^{-1}(x)$$

in the same representation. Therefore, the output of one component can be fed directly into the other without converting back to a Boolean representation of x.

This is the main advantage over a bit-level implementation of ring-GSW multiplication. The circuit still grows with the number of RNS limbs, but the nonlinear depth of the core arithmetic no longer grows with $\log_2 q$. Consequently, the noise growth in BGG+ encodings is governed by a small constant number of nonlinear layers rather than by a Boolean circuit whose depth scales with the bit-length of q.

Algorithm 9 Samp_η^h, where $h \in [L-1]$

Input: 1^λ.

Output: $\widetilde{\mathbf{S}}_{\eta,h} \in \mathbb{Z}_q^{2^{h-1}\times \hat{n}}$, $\widetilde{\mathbf{P}}'_{\eta,h} \in \mathbb{Z}_q^{\hat{n}\times 2\hat{m}}$, $\mathsf{aux}_{h,1}$, $\mathsf{aux}_{h,2}$.

$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{PFFE.Setup}(1^\lambda, 1^\ell, 1^L, 1^M)$.

$(K, \mathsf{aux}_K) \leftarrow \mathsf{PROSamp}(1^\lambda)$.

$\mathbf{z}_\eta := K^{(\eta)}$.

$(\mathsf{ct}_{\mathbf{z}}, \mathsf{st}) \leftarrow \mathsf{PFFE.EncPriv}(\mathsf{mpk}, \mathbf{z}_\eta)$.

$(\hat{\mathbf{s}}_{\mathbf{z}}, \{\hat{\mathbf{S}}_{i,b}\}_{i\in[L],b\in\{0,1\}}) \leftarrow \mathsf{PFFE.Extract}(\mathsf{st})$.

if $h = 1$ **then**

  $\mathcal{P}_{<h-1} = \emptyset$.

else

  For every $i \in [0, h-2]$, $\mathbf{B}_i \leftarrow\!\!\$\ \mathbb{Z}_q^{\hat{n}\times\hat{m}}$.

  $\mathcal{P}_{<h-1} := \left\{\mathbf{p}_{\mathbf{x}_i}\right\}_{i\in[0,h-2],\mathbf{x}_i\in\{0,1\}^i}$, where

$$\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i}\mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i},$$
$$\hat{\mathbf{s}}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{z}} \prod_{j=1}^{i} \hat{\mathbf{S}}_{j,x_j},$$
$$\mathbf{e}_{\mathbf{x}_i} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma_E}^{\hat{m}}.$$

end if

For every $i \in [h, L-1]$, $(\mathbf{B}_i, \mathbf{B}_{i,\gamma_B}^{-1}) \leftarrow \mathsf{TrapGen}(1^{\hat{n}}, 1^{\hat{m}}, q)$.

$\mathcal{K}_{>h} := \left(\left\{\mathbf{K}_{i,b}\right\}_{i\in[h+1,L-1],b\in\{0,1\}}, \left\{\mathbf{K}_{L,b}\right\}_{b\in\{0,1\}}\right)$, where

$$\mathbf{K}_{i,b} \leftarrow\!\!\$\ \mathbf{B}_{i-1,\gamma_B}^{-1}\left(\hat{\mathbf{S}}_{i,b}\mathbf{B}_i + \mathbf{E}_{K,i,b}\right) \quad \text{for } i \in [h+1, L-1],$$
$$\mathbf{K}_{L,b} \leftarrow\!\!\$\ \mathbf{B}_{L-1,\gamma_B}^{-1}\left(\hat{\mathbf{S}}_{L,b} + \mathbf{E}_{K,L,b}\right),$$
$$\mathbf{E}_{K,i,b} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z},\sigma_E}^{\hat{n}\times\hat{m}} \quad \text{for } i \in [h+1, L].$$

$$\widetilde{\mathbf{S}}_h := \begin{pmatrix} \hat{\mathbf{s}}_{\mathbf{0}_{h-1}} \\ \vdots \\ \hat{\mathbf{s}}_{\mathbf{1}_{h-1}} \end{pmatrix}.$$

$\widetilde{\mathbf{P}}'_h := \left(\hat{\mathbf{S}}_{h,0}\mathbf{B}_h,\ \hat{\mathbf{S}}_{h,1}\mathbf{B}_h\right)$.

$\mathsf{sk}_C \leftarrow \mathsf{PFFE.KeyGen}(\mathsf{msk}, C)$.

$\mathsf{aux}_{h,1} := (\mathsf{ct}_{\mathbf{z}}, \mathcal{P}_{<h-1}, \mathcal{K}_{>h})$.

$\mathsf{aux}_{h,2} := (C, \mathsf{mpk}, \mathsf{sk}_C, \mathsf{aux}_K)$.

return $\widetilde{\mathbf{S}}_h, \widetilde{\mathbf{P}}'_h, \mathsf{aux}_{h,1}, \mathsf{aux}_{h,2}$.

Algorithm 10 $\mathsf{Samp}_{\mathsf{PFFESamp}}$

Input: 1^λ.

Output: $\widetilde{\mathbf{S}} \in \mathbb{Z}_q^{2^L \times n_B}$, $\widetilde{\mathbf{P}} \in \mathbb{Z}_q^{n_B \times ((L'+n+1)m+M)}$, $\mathsf{aux}_{h,1}$, $\mathsf{aux}_{h,2}$.

$(f \in \mathcal{F}_{\ell,L,M}, \mathbf{z} \in \{0,1\}^\ell, \mathsf{aux}_{\mathbf{z}} \in \{0,1\}^\star) \leftarrow\!\$\ \mathsf{PFFESamp}_{\mathbf{z}}(1^\lambda)$.

$(\mathbf{t}, \mathsf{pk}) \leftarrow \mathsf{FHE.Setup}(1^\lambda)$, where $\mathbf{t} \in \{0, \pm 1\}^{n+1}$.

$K_B \leftarrow\!\$\ \{0,1\}^\lambda$.

$\mathsf{ct}_{\mathbf{z}} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, (\mathbf{z}, K_B))$.

$\mathbf{m}_{\mathbf{z}} := (1, \mathbf{z}', \mathbf{0}_L, \mathbf{t}) \in \{0,1\}^{L'+n+1}$, where $\mathbf{z}' \in \{0,1\}^{\ell'}$ is a bit string of $\mathsf{ct}_{\mathbf{z}}$.

$\bar{\mathbf{s}}_\epsilon \leftarrow\!\$\ \{0, \pm 1\}^n$ and $\mathbf{s}_\epsilon := (\bar{\mathbf{s}}_\epsilon, -1)$.

$\hat{\mathbf{s}}_{\mathbf{z}} := \mathbf{m}_{\mathbf{z}} \otimes \mathbf{s}_\epsilon$.

for $i \in [L]$ **do**

  $\bar{\mathbf{S}}_{i,b} \leftarrow\!\$\ \{0, \pm 1\}^{n \times n}$ and $\mathbf{S}_{i,b} := \begin{pmatrix} \bar{\mathbf{S}}_{i,b} & 0 \\ 0 & 1 \end{pmatrix}$.

  $\mathbf{M}_{i,b} := \begin{pmatrix} \mathbf{I}_{\ell'+i} & b & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{I}_{(L-i)+n+1} \end{pmatrix}$.

end for

For every $i \in [L-1]$ and $b \in \{0,1\}$, $\hat{\mathbf{S}}_{i,b} := \mathbf{M}_{i,b} \otimes \mathbf{S}_{i,b}$.

$\widetilde{\mathbf{S}} := \begin{pmatrix} \hat{\mathbf{s}}_{\mathbf{0}_L} \\ \vdots \\ \hat{\mathbf{s}}_{\mathbf{1}_L} \end{pmatrix}$, where $\hat{\mathbf{s}}_{\mathbf{x}_L} := \hat{\mathbf{s}}_{\mathbf{z}} \prod_{i=1}^L \hat{\mathbf{S}}_{i,x_i}$ for every $\mathbf{x}_L \in \{0,1\}^L$.

$\mathbf{A}_{\mathsf{att}} \leftarrow\!\$\ \mathbb{Z}_q^{(n+1) \times L'm}$ and $\mathbf{A}_t \leftarrow\!\$\ \mathbb{Z}_q^{(n+1) \times (n+1)m}$.

$G_F \leftarrow \mathsf{MakeVEvalCkt}(n, m, q, F)$, where F is defined as in Eq. (10).

$\mathbf{A}_G \leftarrow \mathsf{VMEvalC}((\mathbf{A}_{\mathsf{att}}, \mathbf{A}_t), G_F)$.

$\mathbf{A}'_G := \mathbf{A}_G \left(\mathbf{I}_M \otimes \mathbf{G}^{-1}(\mathbf{u}_{n+1})\right)$.

$\widetilde{\mathbf{P}} := \left(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_t, \mathbf{A}'_G\right)$.

for $i \in [0, L-1]$ **do**

  $\mathbf{B}_i \leftarrow\!\$\ \mathbb{Z}_q^{n_B \times m_B}$.

  For every $\mathbf{x}_i \in \{0,1\}^i$, $\mathbf{p}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{x}_i} \mathbf{B}_i + \mathbf{e}_{\mathbf{x}_i}$, where $\hat{\mathbf{s}}_{\mathbf{x}_i} := \hat{\mathbf{s}}_{\mathbf{z}} \prod_{j=1}^i \hat{\mathbf{S}}_{j,x_j}$ and $\mathbf{e}_{\mathbf{x}_i} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z},\sigma_E}^{\hat{m}}$.

end for

$\mathsf{aux}_{\mathsf{pffe}} := \{\mathbf{p}_{\mathbf{x}_i}\}_{i \in [0,L-1],\, \mathbf{x}_i \in \{0,1\}^i}$.

$\mathsf{aux}_1 := \left(\mathsf{ct}_{\mathbf{z}}, \mathsf{aux}_{\mathsf{pffe}}\right)$.

$\mathsf{aux}_2 := \left(f, \mathsf{aux}_{\mathbf{z}}, \mathbf{A}_{\mathsf{att}}, \mathbf{A}_t\right)$.

return $\widetilde{\mathbf{S}}, \widetilde{\mathbf{P}}, \mathsf{aux}_1, \mathsf{aux}_2$.

Algorithm 11 $\mathsf{EvalLUT}_{\mathbf{t}_g}$

Input: $\mathbf{A}_u \in \mathbb{Z}_q^{(n+1)\times m}$, $\mathbf{B}^{-1}_{\mathsf{LUT},\gamma_B}$.

Output: $\mathbf{A}_w \in \mathbb{Z}_q^{(n+1)\times m}$, $\{\mathbf{R}_{w,k} \in \mathbb{Z}_q^{(n+1)\times m}\}_{k\in[T]}$, $\{\mathbf{K}_{w,k} \in \mathbb{Z}_q^{m'_B\times m}\}_{k\in[T]}$.

1: $\mathbf{A}_w \leftarrow\!\!\$\ \mathbb{Z}_q^{(n+1)\times m}$.
2: **for** $k \in [T]$ **do**
3: $\quad \mathbf{R}_{w,k} \leftarrow\!\!\$\ \mathbb{Z}_q^{(n+1)\times m}$.
4: $\quad \mathbf{K}_{w,k} \leftarrow\!\!\$\ \mathbf{B}^{-1}_{\mathsf{LUT},\gamma_B}\left(\mathbf{A}_w - \mathbf{t}_g[k']\mathbf{G} - \mathbf{A}_u\mathbf{G}^{-1}(\mathbf{R}_{w,k}) + k'\mathbf{R}_{w,k}\right)$, where $k' := k-1$.
5: **end for**
6: **return** $\mathbf{A}_w, \{\mathbf{R}_{w,k}\}_{k\in[T]}, \{\mathbf{K}_{w,k}\}_{k\in[T]}$.

Algorithm 12 $\mathsf{EvalLUTX}_{\mathbf{t}_g}$

Input: $\{\mathbf{R}_{w,k} \in \mathbb{Z}_q^{(n+1)\times m}\}_{k\in[T]}$, $\{\mathbf{K}_{w,k} \in \mathbb{Z}_q^{m'_B\times m}\}_{k\in[T]}$, $x_u \in \mathbb{Z}$.

Output: $\mathbf{H}_w \in \mathbb{Z}_q^{(m'_b+m)\times m}$ or $\perp$.

1: **if** $x_u > T-1$ **then**
2: $\quad$ **return** $\perp$.
3: **else**
4: $\quad \mathbf{H}_w := \begin{pmatrix} \mathbf{K}_{w,x_u+1} \\ \mathbf{G}^{-1}(\mathbf{R}_{w,x_u+1}) \end{pmatrix}$.
5: $\quad$ **return** $\mathbf{H}_w$.
6: **end if**