Ciphertext-Simulatable HE from BFV with Randomized Evaluation

Intak Hwang, Seonhong Min, and Yongsoo Song

Seoul National University, Seoul, Republic of Korea
{intak.hwang,minsh,y.song}@snu.ac.kr

Abstract. Homomorphic Encryption (HE) is a privacy-enhancing technology that enables computation over encrypted data without the need for decryption. A primary application of HE is in the construction of communication-efficient Two-Party Computation (2PC) protocols between a client and a server, serving as the key owner and the evaluator, respectively. However, the 2PC protocol built on an HE scheme is not necessarily secure, as the standard IND-CPA security of HE does not guarantee the privacy of the evaluation circuit. Several enhanced security notions for HE, such as circuit privacy and sanitization, have been proposed to address this issue, but they require significant overhead in terms of parameter size or time complexity.
In this work, we introduce a novel security notion for HE, called ciphertext simulatability, which precisely captures the security requirements of HE in the construction of 2PC. Then, we provide a concrete construction of ciphertext-simulatable HE from the BFV scheme by modifying its evaluation algorithm. We provide theoretical analysis and demonstrate experimental results to ensure that our solution has insignificant overhead in terms of parameter size and error growth. As a matter of independent interest, we demonstrate how our approach of designing ciphertext-simulatable BFV can be further extended to satisfy stronger security notions such as sanitization.

Keywords: Homomorphic Encryption, Circuit Privacy, Ciphertext Simulatability

1 Introduction

Homomorphic Encryption (HE) is a cryptographic scheme that supports computations over encrypted data. A typical application of HE is a communication-efficient Two-Party Computation (2PC) protocol between a client and a server. In such a protocol, the client generates a key pair consisting of a secret key and an evaluation key, and shares the evaluation key and the encryptions of private data with the server. Upon receiving the evaluation key and ciphertexts from the client, the server evaluates a circuit over the ciphertexts and sends the resulting ciphertexts back to the client. Finally, the client decrypts the output ciphertext to obtain the computation result, while the server learns nothing from the protocol. This approach has been applied to numerous 2PC protocols, such as

Private Information Retrieval [29], Private Set Intersection [14], and Privacy-Preserving Machine Learning [26]. Moreover, this 2PC framework also leads to constructions of numerous cryptographic primitives based on HE, such as Oblivious PRF [1] and Non-Interactive Blind Signatures [6], particularly due to its round-optimality.

Unfortunately, one cannot blindly guarantee the privacy of the aforementioned 2PC framework built on top of HE, even if the underlying HE is IND-CPA secure. To be precise, while the server's view can be simulatable from the IND-CPA security of HE, simulating the client's view is generally not feasible. This is because the output ciphertext may contain non-trivial information about the server's private input beyond the desired output of the computation.

To address this intrinsic issue in HE, two stronger security notions have been primarily studied in the literature. Gentry [20] introduced circuit privacy, which ensures that the output ciphertext from a circuit evaluation reveals no information about the circuit beyond the result. On the other hand, Ducas and Stehlé [16] proposed the sanitization of ciphertexts, which is a public operation that rerandomizes any two ciphertexts that encrypt the same message so that they are indistinguishable. Note that sanitization implies circuit privacy, as one can build a circuit-private evaluation algorithm by sanitizing the output ciphertext.

Two generic approaches were proposed to achieve circuit privacy/sanitization. The first method is noise flooding [20,4], which randomizes the ciphertext by adding an encryption of zero with an exponentially larger error, drowning the information about the existing error in the ciphertext. The second is the soak-spin-repeat method [16], which instead repeats a cycle of adding a moderate amount of error and bootstrapping until the ciphertext is sufficiently randomized.

However, these approaches have significant disadvantages in terms of performance: they either require exponentially larger parameters or multiple bootstrappings. In particular, these limitations are critical for 2PC protocols evaluating small circuits using BFV/BGV-style leveled HE schemes, as they are often instantiated with compact parameters to reduce performance and communication costs. Even with bootstrappable parameters, bootstrapping in leveled schemes is order of magnitudes slower than other operations and requires several GBs of public keys, often making even a single bootstrapping infeasible in practice. While there are works that overcome these problems for GSW/AP-style HE schemes [8,9,25], their approach is generally not applicable to leveled schemes. This is mainly because they usually only focus on single operations, such as external product, while leveled schemes consist of a mix of much more complex operations.

These overheads are partially derived from the robustness of circuit privacy and sanitization. Their main functionality is to remove any remaining information about the server's private input, or even the evaluated circuit, from the output ciphertext. While this naturally leads to a secure 2PC protocol, it may be overkill in some situations, including the case when the circuit is publicly known to both the client and the server.

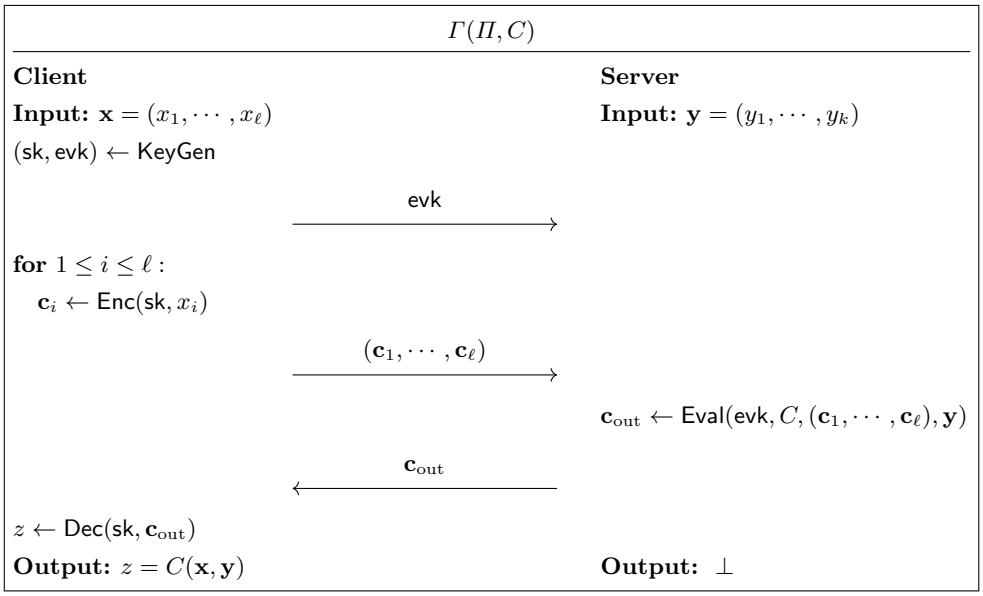

Fig. 1. A two-party protocol Γ computing a circuit C from Π.

1.1 Our Contributions

In this work, we define a new security notion for HE, called *ciphertext simulatability*, which effectively captures the security requirements for simulation-based 2PC protocols built upon it. We then present a concrete construction of ciphertext-simulatable HE based on the BFV scheme, which demonstrates that this relaxed notion can be achieved at a minimal cost compared to previously proposed security notions such as circuit privacy and sanitization.

In many practical scenarios where multiple parties participate, the evaluated circuit C is publicly known to all parties, while the inputs of each party remain private. However, existing HE security notions may be overkill for scenarios where the circuit is known. Therefore, we aim to capture the essence of this scenario with ciphertext simulatability. In a nutshell, an HE scheme is said to be ciphertext-simulatable for a circuit C if the output ciphertext derived from the evaluation of C can be simulated by an efficient algorithm that has no access to the server's private input. Here, the simulator receives the secret and evaluation keys, input ciphertexts, output message, and the circuit. Our notion of ciphertext simulatability can be viewed as a relaxation of circuit privacy in two aspects. Firstly, the evaluation algorithm does not have to hide any information about C, such as its depth or structure, as long as the client cannot extract useful information about the server's input from the output ciphertext. Secondly, the secret key is given to the ciphertext simulator. These two assumptions are reasonable in 2PC scenarios, where the circuit is public to both parties and the client's view includes the secret key. Therefore, even with these relaxations,

we can naturally extend the ciphertext simulatability of an HE scheme to the simulation-based security of the 2PC protocol built on top of it.

Then, we provide a concrete construction of a ciphertext-simulatable HE from the BFV scheme. Specifically, we introduce another security notion called *error simulatability*, where the simulator is asked to simulate only the error of the output ciphertext of a circuit C, given the secret and evaluation keys, errors of the input ciphertexts, and the circuit. We prove that there exists an efficient reduction from error simulatability to ciphertext simulatability for the BFV scheme.

Finally, we construct an error-simulatable variant of the BFV scheme. We first construct a randomized variant for each of the basic operations of the BFV scheme. Then by composing the modified evaluation algorithms, we can evaluate a circuit in an error-simulatable way. Our modified BFV scheme achieves near-optimal error growth, comparable to that of the usual BFV scheme.

To experimentally verify our claim, we provide benchmark results using the proof-of-concept implementation of the standard BFV scheme and our ciphertext-simulatable BFV scheme written in Julia. For real-world examples, we implement a subprotocol of SANNS [12], a k-Nearest Neighbors Search protocol, which consists of computing L2 distance between two vectors. Compared to the traditional approach of noise flooding, our ciphertext-simulatable BFV scheme shows up to 7.20 times speedup. Also, it is only 30% slower than the standard BFV scheme with the same parameters.

As a side contribution, we also show that there exists a reduction from our error-simulatable BFV scheme to stronger security notions, such as sanitization. We also consider *strong ciphertext simulatability*, where the simulator is defined in the same way as ciphertext simulatability but is not given the secret key. We stress that these stronger security notions are overkill for constructing semi-honest 2PC protocols, as we mentioned earlier. Nonetheless, we believe that it may be of independent interest, as these reductions may show the usefulness of our new notion and construction.

1.2 Technical Overview

In this subsection, we give a technical overview of our construction of a ciphertext-simulatable HE scheme based on BFV.

Randomization Techniques. We introduce two ciphertext randomization techniques that serve as the main building blocks of our constructions. First, for multiplicative operations such as plaintext multiplication or ciphertext multiplication, we adapt the randomized lifting trick from [3]. To be more specific, when computing $\mu \cdot a$, we first randomize and lift μ to a Gaussian random variable $\hat{\mu}$ over some coset $\mu + \Lambda$. Then, we add a sufficiently large smoothing Gaussian random variable e. The convolution property of Gaussian distributions guarantees that the output $\hat{\mu}a + e$ also follows some Gaussian distribution independent of μ.

Second, for randomizing masking, we use a specially crafted public key encryption of zero called *ciphertext randomizer*, borrowing the idea from [9]. In

particular, the masking of ciphertext randomizers is uniformly random, independent of the secret key. Therefore, we can always rerandomize the masking of a ciphertext by simply creating a ciphertext randomizer and adding it to the ciphertext.

Randomized Evaluation. The first step of our construction is to randomize each operation of the BFV scheme using the randomization techniques mentioned above. Our goal is to make the error of the output ciphertext independent of its input except for the error of the ciphertext, so that the error of the output ciphertext is simulatable only using the error of the input ciphertexts. We note that since we allow the leakage of the information about operations, some operations do not need randomization at all. For instance, in plaintext-ciphertext or ciphertext-ciphertext additions, the error of the output ciphertext does not interfere with other components of the input ciphertexts other than the errors, so we perform them as usual.

In case of plaintext-ciphertext multiplication, we adapt the oblivious linear evaluation algorithm from [11]. For a plaintext μ and a ciphertext $\mathbf{c}$ with error e, instead of computing $\mu \cdot \mathbf{c}$ we compute $\hat{\mu} \cdot \mathbf{c} + (\tilde{e}, 0)$, where $\hat{\mu}$ is the Gaussian lifting of μ and $\tilde{e}$ is a smoothing Gaussian random variable. Then, the error of the output ciphertext becomes $\hat{\mu} \cdot e + \tilde{e}$, which can be shown to be independent of μ by the aforementioned property of randomized lifting. Although our algorithm is similar to that of [11] in hindsight, the previous work only covered the case where $\mathbf{c}$ is a fresh public-key encryption. In this work, we expand their proofs to support arbitrary ciphertexts.

For ciphertext-ciphertext multiplication, we use a trick similar to Beaver's Triple [7] to circumvent expensive randomizations. To be specific, for input ciphertexts $\mathbf{c}, \mathbf{c}'$ encrypting m, m' respectively, we rerandomize them by adding a ciphertext randomizer and uniformly random plaintext u, u'. In this way, every component of the ciphertexts, including the masking, message, and error, becomes a known constant or random variable following known distributions. Therefore, we can simply multiply two ciphertexts in the usual way and obtain a simulatable ciphertext error. However, the output ciphertext encrypts $(m + u)(m' + u')$ instead of mm'. To correct this, we subtract an encryption of $um' + u'm + uu'$ from the resulting ciphertext, which can be generated with randomized plaintext-ciphertext multiplications described above.

For independent interest, we give an alternative construction based on direct randomization of the suboperations of ciphertext-ciphertext multiplication. While this approach has higher error growth and slower performance compared to the aforementioned approach, it has an interesting property that the distribution of the error of the output ciphertext is Gaussian.

Finally, for automorphisms, the main issue is that the information about the masking of the input ciphertext is added to the error of the output ciphertext after key-switching. As a similar issue arises in computing the external product in GSW/AP-style HE schemes, prior works that achieved circuit privacy in those schemes [8,9,25] used randomized gadget decomposition. However, they were highly inefficient due to the excessive amount of Gaussian sampling. In this

work, we propose a much simpler solution: we just add a ciphertext randomizer before key-switching, so that the mask is rerandomized to a uniform distribution.

Once every operation is randomized, we can chain them to construct a randomized evaluation algorithm for any circuit. A crucial detail is that the Gaussian parameters involved in randomizations of each operation are dependent on the evaluation order. Concretely, they depend on the error bound of their input ciphertexts. Therefore, before evaluation, we compute the optimal parameters for each operation.

Ciphertext-Simulatable BFV. Using our randomized evaluation algorithm for BFV, we finally reduce it to ciphertext-simulatable BFV. Recall that the error of the output ciphertext is simulatable only with the errors of the input ciphertext, which the simulator knows since it has access to the secret key and the input ciphertexts. Moreover, the message of the output ciphertext is known to the simulator. Therefore, to satisfy ciphertext simulatability, we only need to transform the evaluation algorithm so that the masking of the output ciphertext is simulatable. This can be efficiently done by adding a ciphertext randomizer at the end of the evaluation. This concludes our construction of ciphertext-simulatable BFV.

1.3 Related Works

Relaxed Circuit Privacy. Relaxation of circuit privacy by allowing partial leakage of the circuit has also been considered in previous works. For example, Gentry [20] defined circuit privacy with a known circuit level, and Bourse et al. [8] presented a variant of the GSW scheme that is circuit-private for branching programs with a known branching length. In the context of [20,8], revealing only the depth of the circuit was sufficient as they considered evaluation algorithms with a single operation. However, since BFV circuits are usually composed of multiple types of gates, we expand on these works to allow leakage of the circuit's structure.

There are also more general security notions that capture the leakage of the circuit. In [15], the authors defined Φ-circuit privacy, in which the simulator knows the leakage $\Phi(C)$ of the circuit C. Even further, the authors defined input privacy, where the simulator has access to the entire circuit in [30]. These definitions are closely related to ciphertext simulatability, and we will discuss more about them in Section 6.

Malicious Circuit Privacy. There is another line of work [30,15] that studies malicious circuit privacy, where the ciphertexts and the evaluation keys can be generated maliciously. We note that [30] gives a generic transformation from the circuit-private HE scheme to the maliciously circuit-private one. Using our reduction from ciphertext-simulatable BFV to circuit-private BFV in Section 6, we can also construct maliciously circuit-private BFV using their results. Note that it is also possible to consider ciphertext simulatability in malicious settings, which we discuss in Section 6.

2 Preliminaries

For additional preliminaries, we refer to Appendix A.

2.1 Notation

Let n be a power of two and q be an integer. We denote the $2n$-th cyclotomic ring $R = \mathbb{Z}[X]/(X^n+1)$, and similarly $R_q = \mathbb{Z}_q[X]/(X^n+1)$. We use $\mathbb{Z} \cap (-q/2, q/2]$ as a representative set of $\mathbb{Z}_q$, and denote $[a]_q$ as reduction of a modulo q.

We denote column vectors by lowercase bold letters, and matrices by uppercase bold letters. For a ring element $a = a_0 + a_1 X + \cdots + a_{n-1} X^{n-1} \in R$, we will often identify it with its coefficient vector or negacyclic matrix, defined respectively as

$$\mathbf{a} = (a_0, \ldots, a_{n-1}), \quad \mathbf{A} = \begin{bmatrix} a_0 & -a_{n-1} & \cdots & -a_1 \\ a_1 & a_0 & \cdots & -a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & \cdots & a_0 \end{bmatrix}.$$

Note that for any $a, b \in R$, the coefficient vector and negacyclic matrix of ab can be written as $\mathbf{Ab}$ and $\mathbf{AB}$, respectively.

2.2 Homomorphic Encryption

Definition 1 (Homomorphic Encryption). *A Homomorphic Encryption scheme Π is a tuple of PPT algorithms* Setup, KeyGen, Enc, Dec *and* Eval, *described as follows:*

- $\mathsf{Setup}(1^\lambda)$*: Given a security parameter λ, output a public parameter* pp.
- $\mathsf{KeyGen}(\mathsf{pp})$*: Given a public parameter* pp*, output a secret key* sk *and an evaluation key* evk.
- $\mathsf{Enc}(\mathsf{sk}, x)$*: Given a secret key* sk *and a plaintext m, output a ciphertext* $\mathbf{c}$.
- $\mathsf{Dec}(\mathsf{sk}, \mathbf{c})$*: Given a secret key* sk *and a ciphertext c, output a plaintext x.*
- $\mathsf{Eval}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), (y_1, \ldots, y_k))$*: Given a evaluation key* evk*, a circuit C on $(\ell+k)$ variables, ciphertexts $c_1, \ldots, c_\ell$, and plaintexts $y_1, \ldots, y_k$, output a ciphertext* $\mathbf{c}_{\text{out}}$.

Π is said to be correct for a class of circuits $\mathcal{C}$ if for any $x_1, \ldots, x_\ell, y_1, \ldots, y_k$, the following holds with an overwhelming probability:

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), (y_1, \ldots, y_k))) = C(x_1, \ldots, x_\ell, y_1, \ldots, y_k)$$

where $(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ and $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, x_i)$ for $1 \le i \le \ell$. Π is said to be compact if ciphertexts are of polynomial size.

Below, we summarize some of the security notions for HE.

Definition 2 (IND-CPA Security). *A homomorphic encryption scheme* $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *is (IND-CPA) secure if for any PPT adversary* $\mathcal{A}$*,*

$$\left| \Pr\left[b = b' \;\middle|\; \begin{array}{r} (\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}), \\ (m_0, m_1) \leftarrow \mathcal{A}(\mathsf{evk}), \\ b \xleftarrow{\$} \{0,1\}, \\ b' \leftarrow \mathcal{A}(\mathsf{evk}, \mathsf{Enc}(\mathsf{sk}, m_b)) \end{array} \right] - \frac{1}{2} \right| \leq \mathrm{negl}(\lambda).$$

Definition 3 (Circuit Privacy [9]). *A homomorphic encryption scheme* $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *is circuit private for a class of circuits* $\mathcal{C}$ *if there exists a PPT algorithm* Sim *such that for any* $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathcal{M}^\ell$ *and* $C \in \mathcal{C}$*, the following distributions are computationally indistinguishable:*

$$\{(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{c}_{\mathrm{out}}) \mid \mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Eval}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell))\},$$
$$\{(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{c}_{\mathrm{out}}) \mid z := C(\mathbf{x}), \mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Sim}(\mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)\}$$

where $(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ *and* $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, x_i)$ *for* $1 \leq i \leq \ell$*.*

Definition 4 (Sanitization [16]). *Let* $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *be a homomorphic encryption scheme. A PPT algorithm* $\mathsf{Sanitize}$ *is said (computationally) sanitizing if for* $(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ *and any ciphertexts* $\mathbf{c}$ *and* $\mathbf{c}'$ *such that* $\mathsf{Dec}(\mathsf{sk}, \mathbf{c}) = \mathsf{Dec}(\mathsf{sk}, \mathbf{c}')$*, the following distributions are computationally indistinguishable:*

$$\{\mathsf{sk}, \mathsf{evk}, \mathsf{Sanitize}(\mathsf{evk}, \mathbf{c})\}, \;\; \{\mathsf{sk}, \mathsf{evk}, \mathsf{Sanitize}(\mathsf{evk}, \mathbf{c}')\}.$$

Moreover, $\mathsf{Sanitize}$ *is called message-preserving if for any* $\mathbf{c}$*,*

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{Sanitize}(\mathsf{evk}, \mathbf{c})) = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$$

with an overwhelming probability.

2.3 Useful Lemmas

Definition 5 ([27, Smoothing Parameter]). *For a* n*-dimensional lattice* Λ *and* $\varepsilon > 0$*, the smoothing parameter* $\eta_\varepsilon(\Lambda)$ *is defined as the smallest* s *such that* $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \varepsilon$*.*

Definition 6 ([27, Definition 3.2]). *Let* $\mathbf{\Sigma}$ *be a positive definite matrix. We say* $\eta_\varepsilon(\Lambda) \leq \sqrt{\mathbf{\Sigma}}$ *if* $\eta_\varepsilon(\sqrt{\mathbf{\Sigma}}^{-1}\Lambda) \leq 1$*.*

Lemma 1 ([27, Lemma 3.3]). *For a* n*-dimensional lattice* Λ *and* $\varepsilon > 0$*,*

$$\eta_\varepsilon(\Lambda) \leq \lambda_n(\Lambda) \cdot \sqrt{\frac{\ln(2n(1+1/\varepsilon))}{\pi}}$$

where $\lambda_n(\Lambda)$ *is the* n*-th successive minimum of* Λ*.*

Lemma 2 ([19, Lemma 2.3, Theorem 3.1]). *For $0 \leq \varepsilon < 1$, a positive definite matrix $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$, a lattice coset $\mathbf{a} + \Lambda \subset \mathbb{R}^n$, and a matrix $\mathbf{T}$ such that $\ker(\mathbf{T})$ is a Λ-subspace and $\eta_\varepsilon(\ker(\mathbf{T}) \cap \Lambda) \leq \sqrt{\mathbf{\Sigma}}$, then*

$$\Delta_{\mathrm{ML}}(\mathbf{T}\mathcal{D}_{\mathbf{a}+\Lambda,\sqrt{\mathbf{\Sigma}}}, \mathcal{D}_{\mathbf{Ta}+\mathbf{T}\Lambda,\mathbf{T}\sqrt{\mathbf{\Sigma}}}) \leq \log \frac{1+\varepsilon}{1-\varepsilon}.$$

When $\mathbf{T}$ is injective on $\mathbf{a} + \Lambda$, the two distributions are identical without conditions, i.e. $\mathbf{T}\mathcal{D}_{\mathbf{a}+\Lambda,\sqrt{\mathbf{\Sigma}}} \equiv \mathcal{D}_{\mathbf{Ta}+\mathbf{T}\Lambda,\mathbf{T}\sqrt{\mathbf{\Sigma}}}$.

Lemma 3 ([19, Simplified Theorem 4.5]). *Let $\mathbf{\Sigma}_1, \mathbf{\Sigma}_2$ be positive definite matrices, and let $\mathbf{\Sigma}^{-1} := \mathbf{\Sigma}_1^{-1} + \mathbf{\Sigma}_2^{-1}$. If $\eta_\varepsilon(\mathbb{Z}^n) \leq \sqrt{\mathbf{\Sigma}}$ for $0 < \varepsilon < 1$, then for any $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^n$,*

$$\Delta_{\mathrm{ML}}(\mathcal{D}_{\mathbb{Z}^n,\mathbf{c}_1,\sqrt{\mathbf{\Sigma}_1}} + \mathcal{D}_{\mathbb{Z}^n,\mathbf{c}_2,\sqrt{\mathbf{\Sigma}_2}}, \mathcal{D}_{\mathbb{Z}^n,\mathbf{c}_1+\mathbf{c}_2,\sqrt{\mathbf{\Sigma}_1+\mathbf{\Sigma}_2}}) \leq 2\log \frac{1+\varepsilon}{1-\varepsilon}.$$

3 Ciphertext Simulatability

We begin by recalling a well-known limitation of homomorphic encryption (HE) in the context of multi-party protocols: the standard IND-CPA security notion of HE does not always imply the simulation-based security of multi-party protocols built on top of it. Specifically, consider an HE-based two-party computation (Fig. 1), where the client and the server act as the encryptor and evaluator, respectively, to compute a circuit C on their private inputs $\mathbf{x}$ and $\mathbf{y}$. In this scenario, the privacy of the client's data $\mathbf{x} = (x_1, \ldots, x_\ell)$ is directly guaranteed by the security of the underlying HE scheme. However, the server's private input $\mathbf{y} = (y_1, \ldots, y_k)$ is not fully protected against the client, as the resulting ciphertext $\mathbf{c}_{\mathrm{out}}$ may reveal additional information about $\mathbf{y}$ beyond the desired output $z = C(\mathbf{x}, \mathbf{y})$.

To solve this problem, we introduce a new security notion for HE, called *ciphertext simulatability*. Roughly, an HE scheme is called ciphertext-simulatable when the ciphertext $\mathbf{c}_{\mathrm{out}}$ obtained by evaluating a circuit C can be simulated from the input ciphertexts generated by the client and the output $z = C(\mathbf{x}, \mathbf{y})$ of the computation. This security notion is generally applicable to any HE scheme for constructing secure 2PC protocols.

Definition 7 (Ciphertext Simulatability). *A homomorphic encryption scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ over a message space $\mathcal{M}$ is ciphertext-simulatable for a circuit $C : \mathcal{M}^\ell \times \mathcal{M}^k \to \mathcal{M}$ if there exists a PPT algorithm $\mathsf{Sim}^{\mathsf{C}}$ such that for any $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathcal{M}^\ell$ and $\mathbf{y} = (y_1, \ldots, y_k) \in \mathcal{M}^k$, the following distributions are computationally indistinguishable:*

$$\{(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{c}_{\mathrm{out}}) \mid \mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Eval}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), (y_1, \ldots, y_k))\},$$

$$\{(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{c}_{\mathrm{out}}) \mid z = C(\mathbf{x}, \mathbf{y}), \mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Sim}^{\mathsf{C}}(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)\}$$

where $(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ and $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, x_i)$ for $1 \leq i \leq \ell$.

Ciphertext simulatability precisely captures the aforementioned security issue of HE. If an HE scheme is ciphertext-simulatable for a circuit C, then the output ciphertext of homomorphic evaluation of C on inputs $\mathbf{c}_1, \ldots, \mathbf{c}_\ell$ and $y_1, \ldots, y_k$ is (computationally) indistinguishable from a simulated ciphertext from $\mathbf{c}_1, \ldots, \mathbf{c}_\ell$ and $z = C(\mathbf{x}, \mathbf{y})$, which obviously leaks no information about the evaluator's input $y_1, \ldots, y_k$ beyond the computing result $z = C(\mathbf{x}, \mathbf{y})$ to the client. Therefore, it is possible to simulate the client's view only using its input and output, thus prove the security of 2PC. We formally describe this reduction in the following theorem.

Theorem 1. *Let Π be a homomorphic encryption scheme, and $\Gamma(\Pi, C)$ be the 2PC protocol as defined in Fig. 1 for a circuit C. If Π is IND-CPA secure and ciphertext-simulatable for C, then Γ securely computes C in the presence of semi-honest adversaries.*

Proof. Let Sim be a ciphertext simulator for a circuit C. We build two simulators, $\mathsf{Sim}^{\mathrm{Client}}$ and $\mathsf{Sim}^{\mathrm{Server}}$, each simulating the view of the client and the server as follows.

$\mathsf{Sim}^{\mathrm{Client}}(\mathbf{x}, z)$

$(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}$
for $1 \le i \le \ell$
  $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, x_i)$
$\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Sim}(\mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)$
return $(\mathbf{x}, (\mathsf{sk}, \mathsf{evk}, \mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{c}_{\mathrm{out}})$

$\mathsf{Sim}^{\mathrm{Server}}(\mathbf{y})$

$(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}$
for $1 \le i \le \ell$
  $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, 0)$
$\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Eval}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y})$
return $(\mathbf{y}, \mathbf{c}_{\mathrm{out}}, (\mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell)))$

The output of $\mathsf{Sim}^{\mathrm{Client}}$ differs from the view of the client only in the generation of $\mathbf{c}_{\mathrm{out}}$: In the client's view, it is generated with Eval, while in the simulator's output, it is generated with Sim. However, since Π is ciphertext-simulatable for C, the distribution of those two are computationally indistinguishable. Therefore, $\mathsf{Sim}^{\mathrm{Client}}$ correctly simulates the view of the client.

On the other hand, the output of $\mathsf{Sim}^{\mathrm{Server}}$ differs from the view of the server only in the generation of $\mathbf{c}_i$: In the server's view, they are encryptions of x_i, while in the simulator's output, they are encryptions of zero. However, since sk is not included in the view and Π is IND-CPA secure, the distributions of these two are computationally indistinguishable. Therefore, $\mathsf{Sim}^{\mathrm{Server}}$ correctly simulates the view of the server. We conclude that Γ securely computes C under the presence of semi-honest adversaries. □

While circuit privacy (Definition 3) gives similar security guarantees, our definition can be viewed as a relaxation of circuit privacy in two ways. First, the ciphertext simulator in our definition gets the secret key as an input. This is natural in most HE-based 2PC scenarios, as the client typically generates the secret key. Second, we do not consider the circuit as private information. This is implicitly captured in our definition, as in ciphertext simulatability the simulator must simulate the output of a *specific* circuit, while in circuit privacy it must

simulate the output of arbitrary circuits. This allows us to build a simulator using the internal structure of a given circuit.

Although the concept of ciphertext simulatability stems from 2PC, we believe that our definition can be applied in a more generic way, replacing circuit privacy in both practical and theoretical applications. For example, consider a recent construction of Non-Interactive Blind Signatures (NIBS) based on HE by Baldimtsi et al. [6]. In their construction, the signer computes the signature of $F_K(r)$ homomorphically, where F is a PRF, K is a PRF key (given as a ciphertext), and r is some randomness sampled by the signer. Since the information about the signing key must not leak in the final ciphertext, they require that the underlying HE is circuit-private. However, since F and the signature scheme is both a public algorithm, we can simply replace the circuit privacy requirement with ciphertext simulatability without any hassle.

4 Error Simulatability of BFV

Ciphertext simulatability is a general security concept applicable to arbitrary HE schemes. However, we will now focus specifically on the BFV scheme [10,17] and outline a concrete approach to achieve this security notion. It is important to note that the standard BFV scheme does not inherently provide ciphertext simulatability. Therefore, we will construct variants of BFV with alternative evaluation algorithms that satisfy specific properties necessary for achieving advanced security goals. We will continue to refer to these modified constructions as 'BFV', while explicitly describing the security guarantee that each construction can achieve.

As a first step, we introduce the notion of *error simulatability* in this section, which is a security concept for HE schemes where the error of a ciphertext can be defined explicitly. We then build a reduction from error simulatability to ciphertext simulatability for BFV, simplifying our goal in the following section.

4.1 The BFV Scheme

We provide a description of the standard RLWE-based BFV scheme below.

- $\mathsf{Setup}(1^\lambda)$: For a security parameter λ, generate a public parameter as follows:
  - Choose a power-of-two integer n, a ciphertext modulus q, and a plaintext modulus t.
  - Choose a secret key distribution χ_s and an error distribution χ_e over R.
  - Choose a gadget vector $\mathbf{g} \in R_q^d$ and a gadget decomposition $h : R_q \to R^d$.

  Output $\mathsf{pp} = (n, q, t, \chi_s, \chi_e, \mathbf{g}, h)$.

We set χ_s as the uniform distribution over the set of polynomials in R whose coefficients are in the set $\{0, \pm 1\}$, and χ_e as the discrete Gaussian distribution $\mathcal{D}_{\mathbb{Z}^n, \sigma_{\mathrm{err}}}$ where $\sigma_{\mathrm{err}} = 3.2\sqrt{2\pi}$.

In this work, we require that the plaintext modulus t divides the ciphertext modulus q and denote the scaling factor as $\Delta := q/t \in \mathbb{Z}$. While this condition

is not essential for the BFV scheme in general, it is necessary in our solution to formally define the notion of ciphertext error.

• $\mathsf{KeyGen}(\mathsf{pp}, J)$: Generate a secret key, a public key, a relinearization key, and automorphism keys as follows. The generation of automorphism keys is optional, and each automorphism key is associated with an element $j \in \mathbb{Z}_{2n}^{\times}$ of the index set $J \subseteq \mathbb{Z}_{2n}^{\times}$ chosen by the key owner.

- Sample $s \leftarrow \chi_s$ and set the secret key as $\mathsf{sk} = s$.
- Sample $p_1 \xleftarrow{\$} R_q$ and $e_{\mathsf{pk}} \leftarrow \chi_e$, and let $p_0 = -p_1 s + e_{\mathsf{pk}} \pmod q$. Set the public key as $\mathsf{pk} = (p_0, p_1) \in R_q^2$.
- Sample $\mathbf{u}_1 \xleftarrow{\$} R_q^d$ and $\mathbf{e}_{\mathsf{rlk}} \leftarrow \chi_e^d$, and let $\mathbf{U} = (\mathbf{u}_0, \mathbf{u}_1) \in R_q^{d\times 2}$ where $\mathbf{u}_0 = -\mathbf{u}_1 s + \mathbf{g}s^2 + \mathbf{e}_{\mathsf{rlk}} \pmod q$. Set the relinearization key as $\mathsf{rlk} = \mathbf{U}$.
- For each $j \in J$, sample $\mathbf{e}_{\mathsf{atk},j} \leftarrow \chi_e^d$ and $\mathbf{v}_{j,1} \xleftarrow{\$} R_q^d$. Let $\mathbf{v}_{j,0} = -\mathbf{v}_{j,1} s + \mathbf{g}\psi_j(s) + \mathbf{e}_{\mathsf{atk},j} \pmod q$ and $\mathbf{V}_j = (\mathbf{v}_{j,0}, \mathbf{v}_{j,1}) \in R_q^{d\times 2}$ where $\psi_j : X \mapsto X^j$ is the automorphism of index $j \in \mathbb{Z}_{2n}^{\times}$. Set the automorphism key as $\mathsf{atk}_j = \mathbf{V}_j$ and $\mathsf{atk} = \{\mathsf{atk}_j : j \in J\}$.

Output $(\mathsf{sk}, \mathsf{evk})$, where $\mathsf{evk} = (\mathsf{pk}, \mathsf{rlk}, \mathsf{atk})$ denotes the evaluation key.

• $\mathsf{Enc}(\mathsf{sk}, m)$: Given a plaintext $m \in R_t$ and a secret key $\mathsf{sk} = s \in R$, sample $c_1 \leftarrow R_q$ and $e \leftarrow \chi_e$. Compute $c_0 = -c_1 s + \Delta m + e \pmod q$ and return $\mathbf{c} = (c_0, c_1) \in R_q^2$.

• $\mathsf{Dec}(\mathsf{sk}, \mathbf{c})$: Given a ciphertext $\mathbf{c} = (c_0, c_1) \in R_q^2$ and a secret $\mathsf{sk} = s \in R$, output $m = \lfloor (t/q) \cdot (c_0 + c_1 s) \rceil \pmod t \in R_t$.

We only describe the symmetric-key encryption, as the public-key encryption is not necessary for this work.

• $\mathsf{PAdd}(\mathbf{c}, \mu)$: Given a ciphertext $\mathbf{c} = (c_0, c_1)$ and a plaintext $\mu \in R_t$, output $\mathbf{c}_{\text{padd}} = (c_0 + \Delta\mu, c_1) \pmod q$.

• $\mathsf{Add}(\mathbf{c}, \mathbf{c}')$: Given two ciphertexts $\mathbf{c}, \mathbf{c}' \in R_q^2$, output $\mathbf{c}_{\text{add}} = \mathbf{c} + \mathbf{c}' \pmod q$.

• $\mathsf{PMult}(\mathbf{c}, \mu)$: Given a ciphertext $\mathbf{c} \in R_q^2$ and a plaintext $\mu \in R_t$, output $\mathbf{c}_{\text{pmul}} = [\mu]_t \cdot \mathbf{c} \pmod q$.

• $\mathsf{Mult}(\mathsf{rlk}, \mathbf{c}, \mathbf{c}')$: Given two ciphertexts $\mathbf{c} = (c_0, c_1), \mathbf{c}' = (c_0', c_1') \in R_q^2$ and a relinearization key $\mathsf{rlk} \in R_q^{d\times 2}$, do the following:

- Let $\hat{c}_i = [c_i]_q$ and $\hat{c}_i' = [c_i']_q$ for $i = 0, 1$. Compute $\hat{d}_0 = \hat{c}_0\hat{c}_0' \pmod{\Delta q}$, $\hat{d}_1 = \hat{c}_0\hat{c}_1' + \hat{c}_0'\hat{c}_1 \pmod{\Delta q}$, and $\hat{d}_2 = \hat{c}_1\hat{c}_1' \pmod{\Delta q}$[1].
- Compute $d_i = \lfloor \hat{d}_i/\Delta \rceil \pmod q$ for $i = 0, 1, 2$.
- Output $\mathbf{c}_{\text{mult}} = (d_0, d_1) + d_2 \boxdot \mathsf{rlk} \pmod q$.

[1] In the existing works, the notation $[\cdot]_q$ is often opted out when describing $\hat{c}_i$'s. This is an abuse of notation, and we need to use $[c_i]_q$ rather than directly use c_i to perform operations modulo Δq. This fact will play an important rule in our construction in later sections.

- $\mathsf{Aut}(\mathsf{atk}_j, \mathbf{c}, j)$: Given a ciphertext $\mathbf{c} = (c_0, c_1) \in R_q^2$, an automorphism index $j \in \mathbb{Z}_{2n}^\times$, and an automorphism key $\mathsf{atk}_j \in R_q^{d\times 2}$, do the following:
  - Compute $d_0 = \psi_j(c_0)$ and $d_1 = \psi_j(c_1)$.
  - Output $\mathbf{c}_{\mathrm{aut}} = (d_0, 0) + d_1 \boxdot \mathsf{atk}_j \pmod q$.

4.2 Error Simulatability for BFV

In this subsection, we present a method to achieve ciphertext simulatability for the BFV scheme. In general, a BFV encryption of m is of the form $\mathbf{c} = (c_0 = -c_1 s + \Delta m + e, c_1) \in R_q^2$ for some $c_1 \in R_q$ and $e \in R$. We call $c_1 \in R_q$ the *masking*, and $e \in R$ the *error* of $\mathbf{c}$, respectively. We will formally define and denote the error of a ciphertext $\mathbf{c}$ as $\mathsf{Err}_{\mathsf{sk}}(\mathbf{c}) := [c_0 + c_1 s]_\Delta \in R$. Then, a BFV ciphertext can be fully identified by its masking component c_1, error e, and message m. Among the three components of a ciphertext, analyzing the error is the most challenging part in the construction of a ciphertext simulator. In this context, we define a relaxed security notion of ciphertext simulatability for BFV, called *error simulatability* below.

Definition 8 (Error Simulatability). *Let $\mathsf{Eval}^{\mathsf{E}}$ be an evaluation algorithm for the BFV scheme. We say that $\mathsf{BFV}^{\mathsf{E}} := (\mathsf{Setup}, \ldots, \mathsf{Eval}^{\mathsf{E}})$, the BFV scheme associated with the evaluation algorithm $\mathsf{Eval}^{\mathsf{E}}$, is error-simulatable for a circuit $C : R_t^\ell \times R_t^k \to R_t$ if there exists a PPT algorithm $\mathsf{Sim}^{\mathsf{E}}$ such that for any $x_1, \ldots, x_\ell, y_1, \ldots, y_k \in R_t$, the following distributions are computationally indistinguishable:*

$$\left\{ (\mathsf{sk}, \mathsf{evk}, (e_1, \ldots, e_\ell), e_{\mathrm{out}}) \;\middle|\; \begin{array}{r} \mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Eval}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), (y_1, \ldots, y_k)) \\ e_{\mathrm{out}} \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{out}}) \end{array} \right\},$$

$$\{ (\mathsf{sk}, \mathsf{evk}, (e_1, \ldots, e_\ell), e_{\mathrm{out}}) \mid e_{\mathrm{out}} \leftarrow \mathsf{Sim}^{\mathsf{E}}(\mathsf{sk}, \mathsf{evk}, (e_1, \ldots, e_\ell)) \}$$

where $(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, x_i)$ and $e_i \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_i)$ for $1 \le i \le \ell$.

Roughly speaking, a BFV scheme is considered error-simulatable for a circuit C if the error of the resulting ciphertext from homomorphic evaluation can be efficiently simulated using only the errors of the input ciphertexts. Not surprisingly, the standard BFV scheme is not error-simulatable in general, as the error of a ciphertext from homomorphic evaluation depends not only on the errors of the input ciphertext but many other factors such as the maskings of the input ciphertexts and the input plaintexts. In the next section, we will show that we can replace the usual homomorphic evaluation algorithm with a new randomized one to build an error-simulatable variant of the BFV scheme.

Before that, we demonstrate in the rest of this section that if a BFV scheme is error-simulatable for a circuit, then it is also possible to achieve ciphertext simulatability by combining a simple post-processing step with the evaluation algorithm with insignificant overheads in terms of computational complexity and noise growth. Technically, we construct a reduction from error simulatability to ciphertext simulatability for BFV using the following algorithm, generating a random encryption of zero:

• $\mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})$: Given a public key $\mathsf{pk} = (p_0, p_1) \in R_q^2$, sample $e_1, e_2 \leftarrow \mathcal{D}_{\mathbb{Z}^n,\sigma}$ and $e_0 \leftarrow \lfloor \mathcal{D}_{\mathbb{R}^n,\tau} \rceil$. Return the ciphertext $\mathbf{c}_{\mathrm{rand}} = e_2 \cdot (p_0, p_1) + (e_0, e_1) \pmod q$.

Below we analyze the distribution of $\mathbf{c}_{\mathrm{rand}} \leftarrow \mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})$ and show that its masking is computationally indistinguishable from a uniformly random variable over R_q and also independent from the error $\mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{rand}})$. This explains why we do not define masking simulatability as a separate notion, as we can always achieve it by adding $\mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})$ to the ciphertext. We note that a similar result was presented in [9]. However, we adapt their result to the rounded Gaussian case, which improves the efficiency of sampling random variables.

Lemma 4 (Ciphertext Randomizer). *For $\sigma, \tau > 0$, let $\kappa > 0$ be such that*

$$\frac{1}{\kappa^2} = \frac{1}{\sigma^2} + \frac{n^2(1 + K^2\sigma_{\mathrm{err}}^2)}{\tau^2}.$$

If $\kappa \geq 2\eta_\varepsilon(\mathbb{Z}^{2n})$ for some negligible $\varepsilon > 0$ and $\mathsf{RLWE}_{1,n,q,\kappa/\sqrt{2}}$ is hard, then the following holds:

$$\{(\mathsf{sk}, \mathsf{evk}, \mathbf{c}_{\mathrm{rand}}) \mid \mathbf{c}_{\mathrm{rand}} \leftarrow \mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})\} \approx_c \left\{(\mathsf{sk}, \mathsf{evk}, (-us + e, u)) \mid u \xleftarrow{\$} R_q, e \leftarrow \mathcal{D}_{\mathrm{rand}}^{\sigma,\tau}\right\}$$

where $(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, and

$$\mathcal{D}_{\mathrm{rand}}^{\sigma,\tau}(\mathsf{sk}, \mathsf{pk}) := \left\lfloor \mathcal{D}_{\mathbb{R}^n, \sqrt{\sigma^2(\mathbf{S}\mathbf{S}^\top + \mathbf{E}_{\mathsf{pk}}\mathbf{E}_{\mathsf{pk}}^\top) + \tau^2\mathbf{I}}} \right\rceil$$

for $\mathsf{sk} = s$, $\mathsf{pk} = (-p_1 s + e_{\mathsf{pk}}, p_1)$ and $\mathbf{S}$, $\mathbf{E}_{\mathsf{pk}}$, the negacyclic matrices corresponding to s and e_{pk}, respectively. Moreover, $e \leftarrow \mathcal{D}_{\mathrm{rand}}^{\sigma,\tau}$ is bounded by

$$\|e\|_\infty \leq B_{\mathrm{rand}}^{\sigma,\tau} = K\sqrt{\sigma^2(n + K^2 n\sigma_{\mathrm{err}}^2) + \tau^2}$$

with an overwhelming probability.

Proof. See Appendix B. □

Finally, based on the ciphertext randomizer, we provide a generic transformation from an error-simulatable BFV into a ciphertext-simulatable variant.

Theorem 2 (From Error Simulatability to Ciphertext Simulatability). *Suppose $\mathsf{BFV}^{\mathsf{E}} = (\mathsf{Setup}, \ldots, \mathsf{Eval}^{\mathsf{E}})$ is error-simulatable for a circuit C. Then, there exists an efficient evaluation algorithm $\mathsf{Eval}^{\mathsf{C}}$ for the BFV scheme such that $\mathsf{BFV}^{\mathsf{C}} = (\mathsf{Setup}, \ldots, \mathsf{Eval}^{\mathsf{C}})$ is ciphertext-simulatable for C, and the output error bound after evaluating C is at most $B_{\mathrm{rand}}^{\sigma,\tau}$ greater than that of $\mathsf{Eval}^{\mathsf{E}}$ for σ, τ satisfying the conditions in Lemma 4.*

Proof (Sketch). Let $\mathsf{Sim}^{\mathsf{E}}$ be an error simulator for $\mathsf{Eval}^{\mathsf{E}}$ and a circuit $C : \mathcal{M}^\ell \times \mathcal{M}^k \to \mathcal{M}$. We construct a ciphertext-simulatable evaluation algorithm $\mathsf{Eval}^{\mathsf{C}}$ for C, and the corresponding ciphertext simulator $\mathsf{Sim}^{\mathsf{C}}$ from $\mathsf{Eval}^{\mathsf{E}}$ and $\mathsf{Sim}^{\mathsf{E}}$ as follows:

$\mathsf{Eval}^{\mathsf{C}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y})$

$\mathbf{c}_{\text{out}} \leftarrow \mathsf{Eval}^{\mathsf{E}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y})$

$\mathbf{c}_{\text{rand}} \leftarrow \mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})$

$\mathbf{c}'_{\text{out}} \leftarrow \mathbf{c}_{\text{out}} + \mathbf{c}_{\text{rand}} \pmod q$

return $\mathbf{c}'_{\text{out}}$

$\mathsf{Sim}^{\mathsf{C}}(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)$

$(e_1, \ldots, e_\ell) \leftarrow (\mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_1), \ldots, \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_\ell))$

$e_{\text{out}} \leftarrow \mathsf{Sim}^{\mathsf{E}}(\mathsf{sk}, \mathsf{evk}, (e_1, \ldots, e_\ell))$

$\mathbf{c}_{\text{rand}} \leftarrow \mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})$

$\mathbf{c}'_{\text{out}} \leftarrow (\Delta z + e_{\text{out}}, 0) + \mathbf{c}_{\text{rand}} \pmod q$

return $\mathbf{c}'_{\text{out}}$

We claim that for any $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathcal{M}^\ell$ and $\mathbf{y} = (y_1, \ldots, y_k) \in \mathcal{M}^k$, $\mathsf{Eval}^{\mathsf{C}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y})$ and $\mathsf{Sim}^{\mathsf{C}}(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)$ produce computationally indistinguishable distributions where $\mathbf{c}_i \leftarrow \mathsf{Enc}_{\mathsf{sk}}(x_i)$ for $1 \le i \le \ell$ and $z = C(\mathbf{x}, \mathbf{y})$. For the full proof, we refer to Appendix B. □

5 Constructing Error-simulatable BFV

In this section, we construct a variant of the BFV scheme that is error-simulatable for general circuits. More specifically, we randomize the evaluation algorithm of the original BFV scheme while keeping the other algorithms unchanged. In the following sections, we will introduce randomized variants of individual BFV operations, prove their correctness, and provide error analysis to construct error simulators and select optimal parameters.

5.1 Description of New Evaluation Algorithms

Our goal in this section is to redesign the basic operations of the BFV scheme to meet the necessary condition of error simulatability. Specifically, the distribution of the output ciphertext error must depend only on the errors of the input ciphertexts or evaluation keys, while being independent of input plaintexts and other components of the input ciphertexts such as maskings. As we achieve this property with careful randomizations, most of our algorithms accept additional Gaussian parameters such as σ, τ. We note that they must be chosen based on the evaluated circuit, or dynamically during computation, based on the error bound of an input ciphertext.

Additions. There is no need to modify the previous addition operations of BFV described below, as they already satisfy the desired property.

- $\mathsf{PAdd}(\mathbf{c}, \mu)$: Given a ciphertext $\mathbf{c} = (c_0, c_1)$ and a plaintext $x \in R_t$, output $\mathbf{c}_{\text{padd}} = (c_0 + \Delta\mu, c_1) \pmod q$.
- $\mathsf{Add}(\mathbf{c}, \mathbf{c}')$: Given two ciphertexts $\mathbf{c}, \mathbf{c}' \in R_q^2$, output $\mathbf{c}_{\text{add}} = \mathbf{c} + \mathbf{c}' \pmod q$.

Plaintext Multiplication. Our plaintext multiplication algorithm is based on the ideas presented in [11], which involve sampling a *lifting*[2] of μ from a discrete Gaussian distribution. While [11] used a similar technique to construct OLE

[2] An element $\hat{\mu}$ of R is called a lifting of $\mu \in R_t$ if $\hat{\mu} = \mu \pmod t$.

protocol, their security proof only covered fresh, public-key encryptions. In this paper, we expand their work to cover arbitrary ciphertexts.

- $\mathsf{PMult}^{\mathsf{E}}_{\sigma,\tau}(\mathbf{c}, \mu)$: Given a ciphertext $\mathbf{c} \in R_q^2$ and a plaintext $\mu \in R_t$, sample $\hat{\mu} \leftarrow \mathcal{D}_{\mu+t\mathbb{Z}^n,\sigma}$ and $\tilde{e} \leftarrow \lfloor \mathcal{D}_{\mathbb{R}^n,\tau} \rceil$. Output $\mathbf{c}_{\text{pmult}} = \hat{\mu} \cdot \mathbf{c} + (\tilde{e}, 0) \pmod q$.

Randomization. In Section 4.2, we introduced a public-key encryption algorithm and utilized it as a major building block in our reduction from error simulatability to ciphertext simulatability for BFV. We again employ this algorithm as a subroutine of our new ciphertext-ciphertext multiplication and automorphism operations to randomize the underlying key-switching process.

- $\mathsf{Rand}_{\sigma_r,\tau_r}(\mathsf{pk})$: Given a public key $\mathsf{pk} = (p_0, p_1) \in R_q^2$, sample $e_1, e_2 \leftarrow \mathcal{D}_{\mathbb{Z}^n,\sigma_r}$ and $e_0 \leftarrow \lfloor \mathcal{D}_{\mathbb{R}^n,\tau_r} \rceil$. Return the ciphertext $\mathbf{c}_{\text{rand}} = e_2 \cdot (p_0, p_1) + (e_0, e_1) \pmod q$.

We will fix the parameter (σ_r, τ_r) during the setup phase to satisfy the correctness and security conditions, and include it as part of the public parameters. Hence, we will simply write $\mathsf{Rand} = \mathsf{Rand}_{\sigma_r,\tau_r}$ when there is no confusion in the context.

Ciphertext-ciphertext Multiplication. For ciphertext-ciphertext multiplication, we use a randomization trick similar to Beaver's triple [7]. Given an input ciphertext $\mathbf{c}, \mathbf{c}'$ encrypting m, m' respectively, we first rerandomize them as $\hat{\mathbf{c}} \leftarrow \mathbf{c} + \mathbf{r} + (\Delta u, 0)$ and $\hat{\mathbf{c}}' \leftarrow \mathbf{c}' + \mathbf{r}' + (\Delta u', 0)$, where $\mathbf{r}, \mathbf{r}' \leftarrow \mathsf{Rand}(\mathsf{pk})$ and $u, u' \xleftarrow{\$} R_t$. Then, the masking and the message of $\hat{\mathbf{c}}, \hat{\mathbf{c}}'$ become uniformly random over R_q and R_t, respectively. Since the error simulator knows the error of $\mathbf{c}, \mathbf{c}'$ and the secret key, this implies that it can fully simulate $\hat{\mathbf{c}}, \hat{\mathbf{c}}'$. As a result, the error of deterministic multiplication $\mathsf{Mult}(\mathsf{rlk}, \hat{\mathbf{c}}, \hat{\mathbf{c}}')$ also becomes simulatable.

However, the output ciphertext is an encryption of $(m+u)(m'+u') = mm' + um' + u'm + uu'$, instead of the desired mm'. To solve this problem, we correct the output by subtracting an encryption of $um' + u'm + uu$, which can be computed as $\mathsf{PMult}^{\mathsf{E}}(\mathbf{c}, u') + \mathsf{PMult}^{\mathsf{E}}(\mathbf{c}', u) + (\Delta uu', 0)$. Note that PMult operation should be error-simulatable, since otherwise the information about u, u' would leak on the error, causing $m + u$ and $m' + u'$ to be nonuniform to the distinguisher.

As a result, our randomized ciphertext-ciphertext multiplication only requires two more $\mathsf{PMult}^{\mathsf{E}}$ and Rand operations compared to the standard ciphertext-ciphertext multiplication, yielding comparable performance and only additive error growth compared to the deterministic counterpart. We summarize our multiplication algorithm as follows.

- $\mathsf{Mult}^{\mathsf{E}}_{\sigma_1,\tau_1,\sigma_2,\tau_2}(\mathsf{pk}, \mathsf{rlk}, \mathbf{c}, \mathbf{c}')$: Given two ciphertexts $\mathbf{c}, \mathbf{c}' \in R_q^2$, a public key $\mathsf{pk} \in R_q^2$, and a relinearization key $\mathsf{rlk} \in R_q^{d\times 2}$, sample $u, u' \xleftarrow{\$} R_t$, and $\mathbf{r}, \mathbf{r}' \leftarrow \mathsf{Rand}(\mathsf{pk})$. Compute $\hat{\mathbf{c}} \leftarrow \mathbf{c} + \mathbf{r} + (\Delta u, 0) \pmod q$, $\hat{\mathbf{c}}' \leftarrow \mathbf{c}' + \mathbf{r}' + (\Delta u', 0) \pmod q$. Output $\mathbf{c}_{\text{mult}} = \mathsf{Mult}(\mathsf{rlk}, \hat{\mathbf{c}}, \hat{\mathbf{c}}') - \mathsf{PMult}^{\mathsf{E}}_{\sigma_1,\tau_1}(\mathbf{c}, u') - \mathsf{PMult}^{\mathsf{E}}_{\sigma_2,\tau_2}(\mathbf{c}', u) - (\Delta uu', 0) \pmod q$.

Alternative Construction of Ciphertext-ciphertext Multiplication. We also give an alternative construction of ciphertext-ciphertext multiplication based on direct Gaussian lifting, denoted as $\mathsf{MultG}^{\mathsf{E}}$. While $\mathsf{MultG}^{\mathsf{E}}$ has worse error

growth and performance compared to $\mathsf{Mult}^{\mathsf{E}}$, it has an interesting property that the distribution of the error of the output ciphertext follows a Gaussian distribution, which may be of independent theoretical interest.

1. Tensor Product: the usual BFV multiplication algorithm starts with lifting the ciphertext components, i.e., $c_i \mapsto \hat{c}_i = [c_i]_q \in R$. Instead, we offer a randomized lifting that samples each $\hat{c}_i$ from a discrete Gaussian distribution defined over the coset $c_i + q\mathbb{Z}^n$ while still satisfying the correctness condition. We also add an extra Gaussian error for smoothing the error distribution.

2. Modulus Switching: in the following step, we replace the deterministic rounding-off operation $\hat{d}_i \mapsto d_i = \lfloor \hat{d}_i/\Delta \rceil$ with a sampling from a Gaussian distribution centered at $\hat{d}_i/\Delta$. Additionally, we introduce an extra Gaussian error to smooth the distribution.

3. Relinearization: the last step can be viewed as a key-switching procedure from s^2 to s acting on the ciphertext component d_2. Consequently, d_2 also affects the error induced by the gadget product with rlk. To solve this problem and simultaneously make the output error Gaussian, we adapt the randomized gadget decomposition technique from [8,9,25]. We remark that there is a much more efficient way to randomized key-switching if the output error need not to be Gaussian, which we explain in detail below.

We summarize our multiplication algorithm as follows.

• $\mathsf{MultG}^{\mathsf{E}}_{\sigma_1,\tau_1,\sigma_2,\tau_2,\sigma_3,\tau_3}(\mathsf{pk}, \mathsf{rlk}, \mathbf{c}, \mathbf{c}')$: Given two ciphertexts $\mathbf{c}, \mathbf{c}' \in R_q^2$ and a relinearization key $\mathsf{rlk} \in R_q^{d\times 2}$, do the following:
  – $\mathsf{Tensor}^{\mathsf{E}}_{\sigma_1,\tau_1}(\mathbf{c}, \mathbf{c}')$: Sample $\hat{c}_i \leftarrow \mathcal{D}_{c_i+q\mathbb{Z}^n,\sigma_1}$ and $\hat{c}'_i \leftarrow \mathcal{D}_{c'_i+q\mathbb{Z}^n,\sigma_1}$ for $i = 0, 1$. Sample $\hat{e} \leftarrow \lfloor \mathcal{D}_{\mathbb{R}^n,\tau_1} \rceil_{\Delta} \equiv \Delta \big\lfloor \mathcal{D}_{\mathbb{R}^n,\tau_1/\Delta} \big\rceil$. Compute $\hat{d}_0 = \hat{c}_0\hat{c}'_0 + \hat{e} \pmod{\Delta q}$, $\hat{d}_1 = \hat{c}_0\hat{c}'_1 + \hat{c}'_0\hat{c}_1 \pmod{\Delta q}$ and $\hat{d}_2 = \hat{c}_1\hat{c}'_1 \pmod{\Delta q}$. Output $(\hat{d}_0, \hat{d}_1, \hat{d}_2)$.
  – $\mathsf{ModSwitch}^{\mathsf{E}}_{\sigma_2,\tau_2}(\hat{d}_0, \hat{d}_1, \hat{d}_2)$: Sample $d'_i \leftarrow \mathcal{D}_{\mathbb{Z}^n,\frac{1}{\Delta}\hat{d}_i,\sigma_2}$ for $i = 0, 1, 2$. Sample $\tilde{e} \leftarrow \lfloor \mathcal{D}_{\mathbb{R}^n,\tau_2} \rceil$. Compute $d_0 = d'_0 + \tilde{e}$, and set $d_i = d'_i$ for $i = 1, 2$. Output $(d_0, d_1, d_2) \pmod q$.
  – $\mathsf{Relin}^{\mathsf{E}}_{\sigma_3,\tau_3}(\mathsf{rlk}, (d_0, d_1, d_2))$: Sample $h_{\mathrm{rand}}(d_2) \leftarrow \mathcal{D}_{h(d_2)+\Lambda_q^{\perp}(\mathbf{g}\otimes\mathbf{I}_n),\sigma_3}$, $\overline{e} \leftarrow \lfloor \mathcal{D}_{\mathbb{R}^n,\tau_3} \rceil$. Output $\mathbf{c}_{\mathrm{mult}} = (d_0, d_1 + r_0) + \langle h_{\mathrm{rand}}, \mathsf{rlk} \rangle + (\overline{e}, 0) \pmod q$.

Automorphism. Similar to relinearization, the automorphism operation can be understood as a key-switching operation from $\psi_j(s)$ to s. While we can use the randomized gadget decomposition just as $\mathsf{Relin}^{\mathsf{E}}$, it suffers a significant performance overhead in practice. This is because randomized decomposition requires nd Gaussian samples, which are significantly higher than other operations. Instead, we propose a much simpler and more efficient way: we simply add a ciphertext randomizer before key-switching. Then, the distribution of $\psi_j(c_1)$ is rerandomized to a uniform distribution over R_q, independent of other ciphertext components. Finally, we can perform the deterministic gadget product between $\psi_j(c_1)$ and atk_j.

• $\mathsf{Aut}^{\mathsf{E}}(\mathsf{pk}, \mathsf{atk}_j, \mathbf{c}, j)$: Given a ciphertext $\mathbf{c} = (c_0, c_1) \in R_q^2$, a public key $\mathsf{pk} \in R_q^2$, an index $j \in \mathbb{Z}_{2n}^\times$ and a corresponding automorphism key $\mathsf{atk}_j \in R_q^{d\times 2}$, sample $(r_0, r_1) \leftarrow \mathsf{Rand}(\mathsf{pk})$ and compute $d_0 = \psi_j(c_0 + r_0) \pmod q$ and $d_1 = \psi_j(c_1 + r_1) \pmod q$. Output $\mathbf{c}_{\mathrm{aut}} = (d_0, 0) + d_1 \boxdot \mathsf{atk}_j \pmod q$.

5.2 Correctness and Error Analysis

In this section, we precisely analyze the new BFV operations to elaborate on parameter requirements, provide correctness proofs, and establish error distributions along with their bounds. Note that all proofs are deferred to Appendix B.

We begin by introducing two useful lemmas which will be used repeatedly in our proofs.

Lemma 5 (Gaussian Convolution). *Let $\mathbf{E} \in \mathbb{R}^{m\times n}$ a matrix and $\sigma, \tau > 0$ be reals. Let $\mathbf{c} \in \mathbb{R}^n, \mathbf{c}' \in \mathbb{R}^m$ and $\mathbf{a} + \Lambda \subset \mathbb{R}^n$ be a full-rank lattice coset such that $\mathbf{E}(\mathbf{a} + \Lambda) \subseteq \mathbb{Z}^m$. If $\sigma^{-2} + \tau^{-2}\|\mathbf{E}\|_2^2 \le \eta_\varepsilon(\Lambda)^{-2}$ for some $0 < \varepsilon < 1$, then*

$$\Delta_{\mathrm{ML}}\left(\mathbf{E}\mathcal{D}_{\mathbf{a}+\Lambda,\mathbf{c},\sigma} + \lfloor \mathcal{D}_{\mathbb{R}^m,\mathbf{c}',\tau}\rceil, \left\lfloor \mathcal{D}_{\mathbb{R}^m,\mathbf{E}\mathbf{c}+\mathbf{c}',\sqrt{\sigma^2\mathbf{E}\mathbf{E}^\top+\tau^2\mathbf{I}}}\right\rceil\right) \le \log\frac{1+\varepsilon}{1-\varepsilon}.$$

Lemma 6 (Rounded Gaussian Bound). *Let $\mathbf{\Sigma} \in \mathbb{R}^{n\times n}$ be a positive definite matrix. Then for any $k > 0$,*

$$\Pr_{\mathbf{x}\leftarrow\lfloor\mathcal{D}_{\mathbb{R}^n,\sqrt{\mathbf{\Sigma}}}\rceil}[\|\mathbf{x}\|_\infty > k \cdot \max_{1\le i\le n}\sqrt{\mathbf{\Sigma}_{i,i}}] \le 2ne^{-\pi\left(k-\frac{1}{2}\right)^2}.$$

Notations. We use $K > 0$ to denote a global constant such that the probability in Lemma 6 is negligible. For example, when $K = 6$, $2ne^{-\pi\left(K-\frac{1}{2}\right)^2} \le 2^{-100}$ for practical n used in the BFV scheme. We also write $B_{\mathrm{prod}} := KB_{\mathbf{g}}dn\sigma_{\mathrm{err}}$, which is an upper bound of the gadget product error with fresh gadget ciphertexts. Finally, for elements of R (e.g., s, e, e'), we will use lowercase bold letters (e.g. $\mathbf{s}, \mathbf{e}, \mathbf{e}'$) to denote the corresponding coefficient vectors, uppercase bold letters (e.g., $\mathbf{S}, \mathbf{E}, \mathbf{E}'$) to denote the corresponding negacyclic matrices.

Lemma 7 (Addition). *Let $\mathbf{c}, \mathbf{c}' \in R_q^2$ be BFV encryption of $m, m' \in R_t$, respectively, with $e = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c})$, $e' = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}')$. Then, $\mathbf{c}_{\mathrm{padd}} \leftarrow \mathsf{PAdd}(\mathbf{c}, \mu)$ is a BFV encryption of $m + \mu$ with $\mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{padd}}) = e$, and $\mathbf{c}_{\mathrm{add}} \leftarrow \mathsf{Add}(\mathbf{c}, \mathbf{c}')$ is a BFV encryption of $m + m'$ with $\mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{add}}) = e + e'$.*

Proof. The reason is obvious from the definition. □

Lemma 8 (Plaintext Multiplication). *Let $\mathbf{c} \in R_q^2$ be a BFV encryption of $m \in R_t$ with $e = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c})$ such that $\|e\|_\infty \le B$. If $\sigma, \tau > 0$ satisfy $\sigma^{-2} + n^2B^2\tau^{-2} \le (t\eta_\varepsilon(\mathbb{Z}^n))^{-2}$ for some negligible $\varepsilon > 0$, then $\mathbf{c}_{\mathrm{pmult}} \leftarrow \mathsf{PMult}^{\mathsf{E}}_{\sigma,\tau}(\mathbf{c}, \mu)$ is a BFV encryption of μm where the distribution of $e_{\mathrm{pmult}} := \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{pmult}})$ is within max-log distance $\hat{\varepsilon} = \log\frac{1+\varepsilon}{1-\varepsilon}$ from*

$$\mathcal{D}^{\sigma,\tau}_{\mathrm{pmult}}(e) := \left\lfloor \mathcal{D}_{\mathbb{R}^n,\sqrt{\sigma^2\mathbf{E}\mathbf{E}^\top+\tau^2\mathbf{I}}}\right\rceil.$$

Moreover, $\|e_{\mathrm{pmult}}\|_\infty$ *is bounded by* $B^{\sigma,\tau}_{\mathrm{pmult}}(B) := K\sqrt{n\sigma^2B^2+\tau^2}$ *with overwhelming probability.*

Lemma 9 (Multiplication). *Let* $\mathbf{c}, \mathbf{c}' \in R_q^2$ *be BFV encryptions of* $m, m' \in R_t$*, respectively, with* $e = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c})$*,* $e' = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}')$ *such that* $\|e\|_\infty \le B$ *and* $\|e\|_\infty \le B'$*. If* $\sigma_1, \tau_1, \sigma_2, \tau_2 > 0$ *satisfy the conditions of Lemma 8, respectively, then* $\mathbf{c}_{\mathrm{mult}} \leftarrow \mathsf{Mult}^{\mathsf{E}}_{\sigma_1,\tau_1,\sigma_2,\tau_2}(\mathsf{pk}, \mathsf{rlk}, \mathbf{c}, \mathbf{c}')$ *is a BFV encryption of* mm' *where the distribution of* $e_{\mathrm{mult}} := \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{mult}})$ *is computationally indistinguishable from*

$$\mathcal{D}^{\sigma_1,\tau_1,\sigma_2,\tau_2}_{\mathrm{mult}}(s, \mathsf{pk}, \mathsf{rlk}, e, e') := \left\{ e_0 + e_1 + e_2 \;\middle|\; \begin{array}{r} \hat{\mathbf{c}} \leftarrow \mathcal{D}_{\mathrm{multr}}(s, \mathsf{pk}, e) \\ \hat{\mathbf{c}}' \leftarrow \mathcal{D}_{\mathrm{multr}}(s, \mathsf{pk}, e') \\ e_0 \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathsf{Mult}(\mathsf{rlk}, \hat{\mathbf{c}}, \hat{\mathbf{c}}')) \\ e_1 \leftarrow \mathcal{D}_{\mathrm{pmult}}(e) \\ e_2 \leftarrow \mathcal{D}_{\mathrm{pmult}}(e') \end{array} \right\}.$$

where $\mathcal{D}_{\mathrm{multr}}(s, \mathsf{pk}, e) := \{(-as + \Delta u + e + e_{\mathrm{rand}}, a) \mid a \xleftarrow{\$} R_q, u \xleftarrow{\$} R_t, e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}(s, \mathsf{pk})\}$.

Moreover, this distribution is bounded by $B^{\sigma_1,\tau_1,\sigma_2,\tau_2}_{\mathrm{mult}}(B, B') := B_{\mathrm{multd}}(B + B_{\mathrm{rand}}, B' + B_{\mathrm{rand}}) + B^{\sigma_1,\tau_1}_{\mathrm{pmult}}(B) + B^{\sigma_2,\tau_2}_{\mathrm{pmult}}(B')$ *with overwhelming probability, where* $B_{\mathrm{multd}}(B_1, B_2)$ *is the error bound for* $\mathsf{Mult}(\mathsf{rlk}, \mathbf{c}_1, \mathbf{c}_2)$ *with* $\|\mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_1)\|_\infty \le B_1$ *and* $\|\mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_2)\|_\infty \le B_2$*.*

Corollary 1 (Multiplication (Alternative)). *Let* $\mathbf{c}, \mathbf{c}' \in R_q^2$ *be BFV encryptions of* $m, m' \in R_t$*, respectively, with* $e = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c})$*,* $e' = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}')$ *such that* $\|e\|_\infty \le B$ *and* $\|e\|_\infty \le B'$*. If* $\sigma_1, \tau_1, \sigma_2, \tau_2, \sigma_3, \tau_3 > 0$ *satisfy the conditions of Lemma 13, Lemma 14 and Lemma 15, respectively, then* $\mathbf{c}_{\mathrm{mult}} \leftarrow \mathsf{MultG}^{\mathsf{E}}_{\sigma_1,\tau_1,\sigma_2,\tau_2,\sigma_3,\tau_3}(\mathsf{pk}, \mathsf{rlk}, \mathbf{c}, \mathbf{c}')$ *is a BFV encryption of* mm' *where the distribution of* $e_{\mathrm{mult}} := \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{mult}})$ *is computationally indistinguishable from*

$$\mathcal{D}^{\sigma_1,\tau_1,\sigma_2,\tau_2}_{\mathrm{multg}}(s, \mathsf{pk}, \mathsf{rlk}, e, e') := \left\{ e_3 \;\middle|\; \begin{array}{r} e_1 \leftarrow \mathcal{D}^{\sigma_1,\tau_1}_{\mathrm{tensor}}(s, e, e') \\ e_2 \leftarrow \mathcal{D}^{\sigma_2,\tau_2}_{\mathrm{scale}}(s, e_1) \\ e_3 \leftarrow \mathcal{D}^{\sigma_3,\tau_3}_{\mathrm{relin}}(s, \mathsf{rlk}, e_2) \end{array} \right\}.$$

Moreover, this distribution is bounded by $B^{\sigma_1,\tau_1,\sigma_2,\tau_2,\sigma_3,\tau_3}_{\mathrm{multg}}(B, B') := B_{\mathrm{relin}} \circ B^{\sigma_2,\tau_2}_{\mathrm{switch}} \circ B^{\sigma_1,\tau_1}_{\mathrm{tensor}}(B, B')$ *with an overwhelming probability.*

Lemma 10 (Automorphism). *Let* $\mathbf{c} \in R_q^2$ *be a BFV encryption of* $m \in R_t$ *with* $e = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c})$ *such that* $\|e\|_\infty \le B$*. Then,* $\mathbf{c}_{\mathrm{aut}} \leftarrow \mathsf{Aut}^{\mathsf{E}}(\mathsf{pk}, \mathsf{atk}_j, \mathbf{c}, j)$ *is a BFV encryption of* $\psi_j(m)$*, where the distribution of* $e_{\mathrm{aut}} := \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{aut}})$ *is computationally indistinguishable from*

$$\mathcal{D}_{\mathrm{aut}}(s, \mathsf{pk}, \mathsf{atk}_j, e) := \{(\psi_j(e + e_{\mathrm{rand}}) + \langle h(u), \mathbf{e}_{\mathsf{atk}_j}\rangle) \mid e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}(s, \mathsf{pk}), u \xleftarrow{\$} R_q\},$$

which is bounded by $B_{\mathrm{aut}}(B) := B_{\mathrm{prod}} + B_{\mathrm{rand}} + B$ *with overwhelming probability.*

5.3 Optimal Parameters for Error-Simulatable BFV operations

In this subsection, we discuss how to select optimal Gaussian parameters in our evaluation algorithms. Our goal is to minimize the error bound while satisfying the necessary conditions in our lemmas. We first note that this parameter

selection problem can be formulated as follows for some $S, T, \eta > 0$:

$$\text{Find the minimum of } \sqrt{Sx^2 + y^2}, \text{ given the condition } \frac{1}{x^2} + \frac{ST}{y^2} \leq \frac{1}{\eta^2}.$$

For instance, to find the optimal parameters for randomized plaintext multiplication in Lemma 8, we can set $S = nB^2$, $T = n$, and $\eta = t\eta_\varepsilon(\mathbb{Z}^n)$. It is easy to show using the Cauchy–Schwarz inequality that the minimum value of $\sqrt{Sx^2 + y^2}$ is $\sqrt{S}(1 + \sqrt{T})\eta$ when $x = \left(\sqrt{1 + \sqrt{T}}\right)\eta$ and $y = \left(\sqrt{ST + S\sqrt{T}}\right)\eta$. Based on this formula, we provide the optimal parameters and the best error bound for each of the BFV operations in Table 1.

Rand	σ_r	$\sigma_{\mathrm{err}}\sqrt{2\sqrt{n}+2}$
	τ_r	$\sigma_{\mathrm{err}}\sqrt{2(n+\sqrt{n})(1+K^2\sigma_{\mathrm{err}}^2)}$
	B_{rand}	$K\sigma_{\mathrm{err}}(n+\sqrt{n})\sqrt{2+2K^2\sigma_{\mathrm{err}}^2}$
$\mathsf{PMult}^{\mathsf{E}}$	σ	$t\eta\sqrt{\sqrt{n}+1}$
	τ	$Bt\eta\sqrt{n^2+n\sqrt{n}}$
	$B_{\mathrm{pmult}}(B)$	$KBt\eta(n+\sqrt{n})$
$\mathsf{Mult}^{\mathsf{E}}$	σ_1	$t\eta\sqrt{\sqrt{n}+1}$
	τ_1	$Bt\eta\sqrt{n^2+n\sqrt{n}}$
	σ_2	$t\eta\sqrt{\sqrt{n}+1}$
	τ_2	$B't\eta\sqrt{n^2+n\sqrt{n}}$
	$B_{\mathrm{mult}}(B,B')$	$\frac{1}{2}(1+n+n^2)+\frac{1}{\Delta}n(B+B_{\mathrm{rand}})(B'+B_{\mathrm{rand}})$ $+\frac{1}{2}tn(n+4)(B+B'+2B_{\mathrm{rand}})$ $+B_{\mathrm{prod}}+Kt\eta(n+\sqrt{n})(B+B')$
$\mathsf{Aut}^{\mathsf{E}}$	$B_{\mathrm{aut}}(B)$	$B_{\mathrm{prod}}+B_{\mathrm{rand}}+B$

Table 1. Optimal parameters and corresponding output error bounds for each evaluation algorithm. η denotes $\eta_\varepsilon(\mathbb{Z}^n)$ for some negligible $\varepsilon > 0$. We take the error bound for Mult from [23].

5.4 Proving Error Simulatability of BFV for Arbitrary Circuits

In this section, we combine our basic BFV operations to describe a new evaluation algorithm $\mathsf{Eval}^{\mathsf{E}}$ (Fig. 3) which is error-simulatable for general circuits. One distinctive feature of our evaluation algorithm is that the parameter set of each gate is chosen dynamically depending on the structure of a given circuit.

The parameter selection algorithm $\mathsf{SetParams}$ (Fig. 2) aims to assign an error bound to each wire that a ciphertext passes through and to determine the

$\mathsf{SetParams}(C)$

Let $G_1, \ldots, G_L$ be the gates of C, sorted in topological order.

For each ciphertext input wire w, set the error bound as $B_w = K\sigma_{\mathrm{err}}$.

for $1 \leq i \leq L$:

Use Table 1 and the error bound(s) $B_{w_{\mathrm{in}}}$ of ciphertext input wire(s) w_{in} to compute the public parameter $\mathsf{pp}_i^{\mathsf{E}}$ for G_i and the error bound w_{out} for the output wire.

Output $\{\mathsf{pp}_i^{\mathsf{E}} \mid 1 \leq i \leq L\}$.

Fig. 2. Parameter selection algorithm for a given circuit C.

parameters for individual unit gates. More specifically, it begins with initializing the error bounds of input ciphertext wires as the upper bound of freshly encrypted ciphertexts. Then, following the topological order of the circuit, it determines the parameters of each gate and assigns an error bound to the output wire iteratively based on the predetermined error bound(s) of input ciphertext wire(s) using Table 1. This procedure is applied to all gates and wires, thus creating an optimal chain of parameters. We also note that this operation is entirely deterministic and does not depend on actual input plaintexts or ciphertexts of the circuit.

We remark that our construction of $\mathsf{Eval}^{\mathsf{E}}$ is focused on theoretical simplicity rather than practicality. In other words, we may construct a much more efficient evaluation algorithm by using the structure of the circuit more effectively. For instance, we may change the representation of the circuit to have fewer randomized operations, or compute a part of the circuit in a deterministic manner if no private inputs are involved. We give an example of such optimization in Section 7.

Theorem 3. *For a circuit C, let B_{out} be the error bound of the output wire of C, where the bound is computed as $\mathsf{SetParams}(C)$. If $B_{\mathrm{out}} \leq \Delta/2$, then $\mathsf{BFV}^{\mathsf{E}} = (\mathsf{Setup}, \ldots, \mathsf{Eval}^{\mathsf{E}})$ is error-simulatable for C.*

Proof. Let $G_1, \ldots, G_L$ be the gates of C, topologically ordered from the input wire. For $0 < j < L$, we define hybrid games $\mathbf{Hyb}_j(\mathsf{sk}, \mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y})$ as follows. First, $\mathbf{Hyb}_j$ computes gates $G_1, \ldots, G_j$ in the same way as $\mathsf{Eval}^{\mathsf{E}}$. Then, it replaces each ciphertext on any wire in C with its error e. Then, it evaluates the rest of the gates $G_{j+1} \ldots, G_L$ in the same way as $\mathsf{Sim}^{\mathsf{E}}$. We also define $\mathbf{Hyb}_0 := \mathsf{Sim}^{\mathsf{E}}(\mathsf{sk}, \mathsf{evk}, e_1, \ldots, e_\ell)$ and $\mathbf{Hyb}_L := \mathsf{Err}_{\mathsf{sk}}(\mathsf{Eval}^{\mathsf{E}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y}))$ where $e_i \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_i)$.

Now we show that $\mathbf{Hyb}_0$ and $\mathbf{Hyb}_L$ are computationally indistinguishable using hybrid argument. To prove our claim by contradiction, we suppose that there exists an efficient adversary $\mathcal{A}$ which can distinguish $\mathbf{Hyb}_j$ and $\mathbf{Hyb}_{j+1}$ with a non-negligible advantage for some $0 \leq j < L$. Since the only difference between $\mathbf{Hyb}_j$ and $\mathbf{Hyb}_{j+1}$ is in the evaluation of the gate G_{j+1}, the adversary should be able to distinguish the output distributions of G_{j+1} in two hybrid

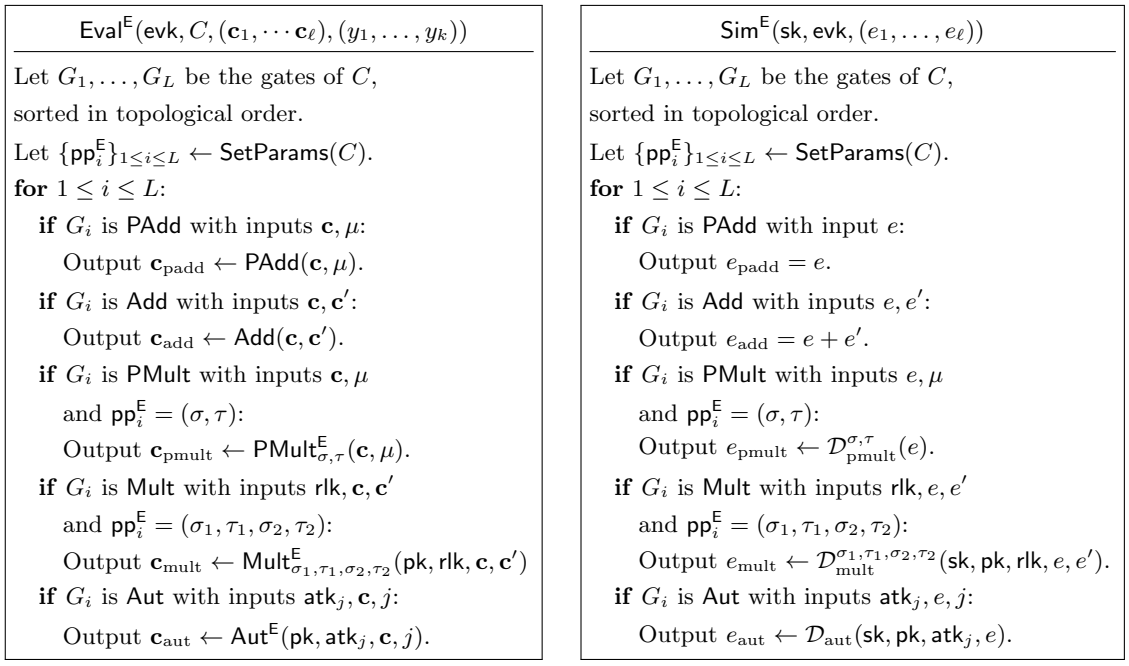

Fig. 3. Error-simulatable evaluation algorithm $\mathsf{Eval}^{\mathsf{E}}$ for a circuit C, and the corresponding error simulator $\mathsf{Sim}^{\mathsf{E}}$.

games. In other words, the adversary can distinguish with a non-negligible probability whether the error of the output wire of G_{j+1} is obtained by homomorphically evaluating G_{j+1} on input ciphertext(s) and extracting the error of an output ciphertext, or it is simulated using the corresponding error distribution determined by the error(s) of input ciphertext(s). However, this contradicts to Lemmas 7 to 10. □

Theorem 4. *For a circuit C, let B_{out} be the error bound of the output wire of C, where the bound is computed as* $\mathsf{SetParams}(C)$*. Moreover, let* $\mathsf{Eval}^{\mathsf{C}}$ *be an evaluation algorithm for the BFV scheme where* $\mathsf{Eval}^{\mathsf{C}}(\mathsf{evk}, C, (\mathbf{c}_1, \cdots, \mathbf{c}_\ell), \mathbf{y}) = (\mathsf{Eval}^{\mathsf{E}}(\mathsf{evk}, C, (\mathbf{c}_1, \cdots, \mathbf{c}_\ell), \mathbf{y}), 0) + \mathsf{Rand}(\mathsf{pk})$*. If $B_{\text{out}} + B_{\text{rand}} \le \Delta/2$, then* $\mathsf{BFV}^{\mathsf{C}} = (\mathsf{Setup}, \ldots, \mathsf{Eval}^{\mathsf{C}})$ *is ciphertext-simulatable for C.*

Proof. Obvious from Theorems 2 and 3. □

5.5 Asymptotic Comparison

We now present a comparison of the asymptotic overhead which the randomized operations of our ciphertext-simulatable variant have, in contrast to the conventional BFV scheme. Additionally, we outline the asymptotic complexity of our

approach to achieve circuit privacy, comparing it with generic methods like noise flooding or the soak-spin-repeat technique [16].

We start by examining the noise overhead of our approach in comparison to the standard BFV scheme. Recall that we do not add any randomness to the additive operations, hence our focus is solely on assessing the overhead introduced by multiplication operations and the automorphism operation. For concrete noise bounds, we refer to Table 1.

In case of plaintext multiplication, in our randomized operation, the error grows by a factor of $Kt\eta(n+\sqrt{n})$, while in the standard scheme it grows by a factor of tn. Hence, the overhead is $Kt\eta(n+\sqrt{n})/tn = K\eta(1+1/\sqrt{n}) \approx K\eta$. While K and η are $O(\mathrm{poly}\,\lambda)$, in practice we can consider them as constants. Concretely, we can set $K = \eta = 6$, which implies that $K\eta \approx 2^5$.

Our ciphertext multiplication consists of a deterministic multiplication of two randomized ciphertexts and two randomized plaintext multiplications. First, notice that the dominating factor in the error bound of standard ciphertext multiplication is $\frac{1}{2}tn(n+4)(B+B')$, which solely depends on the error bounds B and B' of the input ciphertexts. Since $B, B' \gtrapprox B_{\mathrm{rand}}$, the noise growth from the standard ciphertext multiplication remains almost the same, and the additional error only comes from two randomized plaintext multiplications, which is bounded by $Kt\eta(n+\sqrt{n})(B+B')$. As this is almost negligible compared to the error growth from the ciphertext multiplication, the overhead of our randomized ciphertext multiplication is near 1.

Similarly, in the automorphism operation, our randomized automorphism introduces an additive error denoted as B_{rand}. Since $B_{\mathrm{prod}} + B \gtrapprox B_{\mathrm{rand}}$, the error growth in our automorphism operation is nearly equivalent to that of the deterministic automorphism, with an overhead close to 1.

In Table 2, we provide a summary of these discussions. As shown above, our scheme maintains a nearly constant overhead relative to the standard BFV scheme, which is close to being optimal. In contrast, the noise flooding method [20] requires parameters to be exponentially large to support at least λ-bits of overhead. Meanwhile, the soak-spin-repeat method [16] necessitates parameters that enable bootstrapping. Therefore, for circuits with a shallower depth compared to the bootstrapping circuit, our scheme is asymptotically better than soak-spin-repeat, as it sustains a nearly constant overhead. Additionally, it is important to note that in practice, the soak-spin-repeat method could result in substantial communication costs because the bootstrapping keys for BFV can be several hundred GBs in size.

6 Stronger Security Notions

This work primarily focused on the notion of ciphertext simulatability and how to achieve it from the BFV scheme. As a matter of independent interest, we demonstrate how our approach of designing error-simulatable (and thus ciphertext-simulatable) BFV can be further extended to satisfy stronger security notions in this section.

Operation	Standard	Ours	Overhead
PAdd	B	B	1
Add	$B + B'$	$B + B'$	1
PMult	Btn	$KBt\eta(n + \sqrt{n})$	$\approx K\eta$
Mult	$\frac{1}{2}(1 + n + n^2) + \frac{1}{\Delta}nBB'$ $+\frac{1}{2}tn(n + 4)(B + B')$ $+B_{\mathrm{prod}}$	$\frac{1}{2}(1 + n + n^2)$ $+\frac{1}{\Delta}n(B + B_{\mathrm{rand}})(B' + B_{\mathrm{rand}})$ $+\frac{1}{2}tn(n + 4)(B + B' + 2B_{\mathrm{rand}})$ $+B_{\mathrm{prod}} + Kt\eta(n + \sqrt{n})(B + B')$	≈ 1
Aut	$B_{\mathrm{prod}} + B$	$B_{\mathrm{prod}} + B_{\mathrm{rand}} + B$	≈ 1

Table 2. Comparison of error growth of each operations on standard BFV scheme and our ciphertext-simulatable BFV scheme. 'Overhead' denotes the ratio between the error growth of our scheme and the standard scheme. Error bound of the input ciphertext(s) are denoted as B, B', and $\eta_\varepsilon(\mathbb{Z}^n)$ is denoted as η for some negelible $\varepsilon > 0$. We take the error bound for standard BFV scheme from [23].

6.1 Strong Ciphertext Simulatability

Below we introduce a stronger security notion of ciphertext simulatability, called *strong ciphertext simulatability*. A primary difference in strong ciphertext simulatability is that the simulator does not have access to the secret key of the HE scheme, whereas a simulator for the standard ciphertext simulatability does. Therefore, it is more challenging to build a valid simulator for strong ciphertext simulatability as it must simulate the output ciphertext using only input ciphertexts and the computational result, while a simulator for ciphertext simulatability can decrypt input ciphertexts and observe their errors as shown in the proof of Theorem 2.

Definition 9 (Strong Ciphertext Simulatability). *A homomorphic encryption scheme* $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *over a message space* $\mathcal{M}$ *is strongly ciphertext-simulatable for a circuit* $C : \mathcal{M}^\ell \times \mathcal{M}^k \to \mathcal{M}$ *if there exists a PPT algorithm* $\mathsf{Sim}^{\mathsf{SC}}$ *such that for any* $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathcal{M}^\ell$, $\mathbf{y} = (y_1, \ldots, y_k) \in \mathcal{M}^k$, *the following distributions are computationally indistinguishable:*

$$\{(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{c}_{\mathrm{out}}) \mid \mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Eval}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), (y_1, \ldots, y_k))\},$$

$$\{(\mathsf{sk}, \mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{c}_{\mathrm{out}}) \mid z = C(\mathbf{x}, \mathbf{y}), \mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Sim}^{\mathsf{SC}}(\mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)\}$$

where $(\mathsf{sk}, \mathsf{evk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ *and* $\mathbf{c}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, x_i)$ *for* $1 \le i \le \ell$.

We note that a similar concept, referred to as *input privacy*, was introduced in [30] within a different context where the authors considered non-compact HE schemes, such as constructions based on garbled circuits. It is also worth noting that a HE scheme is strongly ciphertext-simulatable for a circuit $C : \mathcal{M}^\ell \times \mathcal{M}^k \to \mathcal{M}$ if and only if the scheme is computationally circuit-private for a class of circuits $\mathcal{F} := \{F_{\mathbf{y}} : \mathcal{M}^\ell \to \mathcal{M} \mid F_{\mathbf{y}}(\mathbf{x}) = C(\mathbf{x}, \mathbf{y})\}$.

Now, we show that there also exists a generic reduction from error-simulatable BFV scheme to a strongly ciphertext-simulatable one. Our reduction is slightly looser than the reduction from error simulatability to ciphertext simulatability presented in Theorem 2, but the overhead is still minimal.

Notice that the direct translation of the reduction does not work, as it requires the secret key to compute the error of the input ciphertexts. To address this issue, we directly use the error-simulatable BFV evaluation algorithm $\mathsf{Eval}^{\mathsf{E}}$, instead of the ciphertext simulator $\mathsf{Sim}^{\mathsf{E}}$, in the simulator. Since a strong ciphertext simulator receives only the final output $z = C(\mathbf{x}, \mathbf{y})$ and is not given the private input $\mathbf{y} = (y_1, \ldots, y_k)$, our simulator simply executes the evaluation algorithm on the trivial input $\mathbf{y} = \mathbf{0}$. Note that to ensure correctness, we require an additional PMult gate after the evaluation of C, causing additional overhead compared to the reduction of Theorem 2.

Theorem 5 (Reduction from Error to Strong Ciphertext Simulatability). *Suppose that* $\mathsf{Eval}^{\mathsf{E}}$ *is an evaluation algorithm for the BFV scheme. If* $\mathsf{Eval}^{\mathsf{E}}$ *is error-simulatable for a circuit* C*, then there exists an efficient strongly ciphertext-simulatable evaluation algorithm* $\mathsf{Eval}^{\mathsf{SC}}$ *for the same circuit whose error bound is at most* $B_{\mathrm{rand}} + B^{\sigma,\tau}_{\mathrm{pmult}}(B_{\mathrm{out}})$*, where* B_{out} *is the error bound of the output wire of* C *computed as* $\mathsf{SetParams}(C)$ *and* σ, τ *satisfies the conditions in Lemma 8 for input error bound* B_{out}*.*

Proof. See Appendix B. □

6.2 Sanitization

The output distributions of a message-preserving sanitization algorithm are indistinguishable for any input ciphertexts $\mathbf{c}$ and $\mathbf{c}'$ with the same message. In this subsection, we show that BFV ciphertexts can be sanitized by a single randomized bootstrapping, without noise flooding. Compared to the soak-and-spin method [16], which performs an iterative bootstrapping with moderate noise flooding, our construction only requires a single bootstrapping and therefore enjoys a comparably small computational overhead.

Below, we describe how our ciphertext-simulatable BFV evaluation algorithm can be combined with known bootstrapping methods [21,13,18]. For the bootstrapping, we set $t' = p^{r'}$ and $t = p^r$ for some prime p and positive integers $r' > r$ such that $t' \mid q$. Also, we assume that the baseline BFV scheme supports the packing for the point-wise arithmetic of integer vectors. Then, the BFV bootstrapping consists of four steps, namely Modulus Switching, Coeffs-to-Slots, Digit Extraction, and Slots-to-Coeffs.

• Modulus Switching : For the input encryption $(b, a) \in R_q^2$ of $m \in R_t$, the following value is homomorphically computed:

$$\left[\lfloor \frac{t'}{q} b \rceil + \lfloor \frac{t'}{q} a \rceil \cdot s \right]_{t'} = \frac{t'}{t} \cdot m + e \in R_{p^r}$$

for some $e \in R$. This can be easily computed with a linear combination, given the (trivial) encryption $(0, \frac{q}{t'})$ of the secret s. To be specific, we compute and

output the ciphertext $\left(\frac{q}{t'}b, \frac{q}{t'}a\right) \in R_q^2$. Then, it is indeed an error-free encryption of $\frac{t'}{t} \cdot m + e$ since $\frac{q}{t'}b + \frac{q}{t'}a \cdot s = \frac{q}{t'}(b + as) = \frac{q}{t'} \cdot \left(\frac{t'}{t} \cdot m + e\right) \pmod q$.

• Coeffs2Slots : In this step, we homomorphically perform a linear transformation to generate a ciphertext encrypting the coefficient vector of the original plaintext in plaintext slots. In other words, we obtain a ciphertext encrypting the plaintext $\mu \in R_{t'}$, which encodes a coefficient vector

$$\left(\frac{t'}{t}m_0 + e_0, \ldots, \frac{t'}{t}m_{n-1} + e_{n-1}\right) \in \mathbb{Z}_{t'}$$

of the polynomial $\frac{t'}{t} \cdot m + e \in R_{t'}$ in its slots. To compute this homomorphically, a linear sum of automorphisms with publicly known coefficients is evaluated over the input ciphertext. Note that we may have several ciphertexts when the slot size and the ring degree do not match.

• DigitExtract : In this step, the scaled rounding function $\lfloor t/t' \cdot x \rceil$ is homomorphically computed on the erroneous coefficients $\{\frac{t'}{t}m_i + e_i\}_{0 \le i < n}$, using polynomial evaluations and yet another homomorphic operation called the 'homomorphic division'. The homomorphic division by p is a special operation which homomorphically divides the plaintext $p \cdot m \in R_{p^{i+1}}$ by p, obtaining the new plaintext $m \in R_{p^i}$ in the new plaintext space, for some $i \ge 1$. This can be realized with a simple change of plaintext modulus, without any alteration on the ciphertext or the noise, since $b + as = p \cdot m \cdot \frac{q}{p^{i+1}} + e = m \cdot \frac{q}{p^i} + e \pmod q$.

During the homomorphic rounding, this homomorphic division operation is performed between the polynomial evaluations. Recall that we set $t = p^r$ and $t' = p^{r'}$, evaluating the scaled rounding function $\left\lfloor \frac{t}{t'}x \right\rceil$ is implemented by iteratively removing the lower $r' - r$ digits of x in base p representation. More precisely, given the input $x = x_{r'-1} \ldots x_0 \in \mathbb{Z}_{p^{r'}}$ in base-p representation, we first extract the last digit $x_0 = [x]_p \in \mathbb{Z}_{p^{r'}}$ with polynomial evaluation. Then, by homomorphically dividing $x - x_0$ by p as it is a multiple of p, we obtain $x_{r'-1} \ldots x_1 \in \mathbb{Z}_{p^{r'-1}}$. By repeating this procedure $r' - r$ times, we can obtain a ciphertext encrypting the coefficients $m_0, \ldots, m_{n-1}$ in plaintext slots. It should be noted that this method is more depth-optimized in practice [13,18], but the main strategy remains the same.

• Slots2Coeffs : In this step, the plaintext is homomorphically unpacked. In other words, the encoded values in the slot are moved to the coefficients. In other words, we obtain an encryption of $m = \sum_{i=0}^{n-1} m_i X^i \in R_t$. As with Coeffs-to-Slots, its homomorphic computation is instantiated by the linear transformation, which is essentially a linear sum of automorphisms with publicly known constants.

Note that the ciphertext after the modulus switching is an error-free encryption of $[b + as]_{t'}$, the whole bootstrapping process is error simulatable as long as the following steps after the modulus switching are error simulatable, as the simulator can simply set the coefficient matrix for the initial error to zero. Indeed, the Coeffs-to-Slots and Slots-to-Coeffs step can be simply randomized as a linear sum of randomized automorphisms with publicly known coefficients. Also, Digit Extraction is also error-simulatable, since the division by p operation does not

affect the error distribution at all, and the polynomial evaluation can be instantiated with the randomized ciphertext and plaintext multiplications. Hence, we obtain the error simulatability, and consequently ciphertext simulatability for the whole bootstrapping process. Then, this randomized bootstrapping satisfies sanitization, as the output ciphertext distributions for the input ciphertexts with the same message are indistinguishable.

6.3 Malicious Ciphertext Simulatability

Some works [30,15] consider malicious circuit privacy, which requires the circuit-private property to hold even if keys and ciphertexts are maliciously generated. We can define *malicious ciphertext simulatability* in a similar manner.

The most simple way to achieve malicious ciphertext simulatability is using Non-Interactive Zero Knowledge Proofs of Knowledge (NIZKPoK) of keys and ciphertexts. Note that our construction (Fig. 3) assumes two things: the well-formedness of ciphertext and keys, and the infinite norm bound of the secret key and the errors. In particular, whether the secret key and the errors follow a specific distribution is not important. Therefore, we can adapt NIZKPoK for HE ciphertexts and keys that provide such proofs, such as [22], to upgrade our construction to the malicious setting.

7 Experimental Results

In this section, we present benchmark results of our ciphertext-simulatable BFV scheme, and compare it to the standard BFV scheme, with and without noise flooding. Since randomized operations rely on Gaussian distribution with parameters depending on the circuit, it is hard to provide a general performance analysis. In this section, we give an example of SANNS [12], a 2PC protocol for k-Nearest Neighbors Search. In particular, we implement a subprotocol of SANNS, which computes the L2 distance between two vectors of length d. The evaluation of the circuit and the corresponding ciphertext-simulatable evaluation is given as follows.

$\mathsf{Eval}(\mathsf{evk}, C_{\ell_2}, \mathbf{c}, y)$

1 : $\mathbf{c}_{\text{sub}} \leftarrow \mathsf{PAdd}(\mathbf{c}, -y)$
2 : $\mathbf{c}_{\text{out}} \leftarrow \mathsf{Mult}(\mathsf{rlk}, \mathbf{c}_{\text{sub}}, \mathbf{c}_{\text{sub}})$
3 : **for** $i \in 1, \ldots, \log d$
4 : $\quad \mathbf{c}_{\text{rot}} \leftarrow \mathsf{Aut}(\mathsf{atk}_{n/2^i}, \mathbf{c}_{\text{out}}, n/2^i)$
5 : $\quad \mathbf{c}_{\text{out}} \leftarrow \mathsf{Add}(\mathbf{c}_{\text{out}}, \mathbf{c}_{\text{rot}})$
6 : **return** $\mathbf{c}_{\text{out}}$

$\mathsf{Eval}^{\mathsf{C}}(\mathsf{evk}, C_{\ell_2}, \mathbf{c}, y)$

1 : $\mathbf{c}_{\text{sub}} \leftarrow \mathsf{PAdd}(\mathbf{c}, -y)$
2 : $\mathbf{c}_{\text{out}} \leftarrow \mathsf{Mult}^{\mathsf{E}}(\mathsf{rlk}, \mathbf{c}_{\text{sub}}, \mathbf{c}_{\text{sub}})$
3 : **for** $i \in 1, \ldots, \log d$
4 : $\quad \mathbf{c}_{\text{rot}} \leftarrow \mathsf{Aut}^{\mathsf{E}}(\mathsf{atk}_{n/2^i}, \mathbf{c}_{\text{out}}, n/2^i)$
5 : $\quad \mathbf{c}_{\text{out}} \leftarrow \mathsf{Add}(\mathbf{c}_{\text{out}}, \mathbf{c}_{\text{rot}})$
6 : $\mathbf{c}_{\text{out}} \leftarrow \mathbf{c}_{\text{out}} + \mathsf{Rand}(\mathsf{pk})$
7 : **return** $\mathbf{c}_{\text{out}}$

Note that the last for loop, which is for computing the summation of the slots homomorphically, is not present in SANNS. Instead, the client performs on the

plain after decryption. However, for the sake of comparison, we include it in our circuit to accommodate a wider range of operations.

7.1 Parameter Selection

In this subsection, we describe the parameter choices of our implementation. Our base parameters for the standard and randomized BFV scheme are summarized in Table 3. First, we set $t = 2^{23}$, following the scenario presented in [12]. This allows us to compute the distance between 8-bit vectors of length at most 128. The parameters with ring degree $n = 2^{12}$ are used for our scheme, and the standard BFV without noise flooding. For the standard BFV with noise flooding, we set the flooding noise to achieve 2^{-80} statistical distance. However, we could not find a suitable parameter for noise flooding when $n = 2^{12}$, so we increase the ring degree to $n = 2^{13}$ and increase q accordingly. Such restrictions are common among protocols using BFV, such as in [5].

The parameters we selected achieve 128-bit security according to lattice-estimator [2], with the key distribution χ_s as a uniform distribution over polynomials in R_q with coefficients $\{0, \pm 1\}$, and χ_e as $\mathcal{D}_{\sigma_{\text{err}}}$ where $\sigma_{\text{err}} = 3.2\sqrt{2\pi}$. For efficient computation using RNS representation, the ciphertext modulus is set with a multiple of a few word-size integers. Consequently, the RNS gadget decomposition is employed for the key-switching.

Finally, following Table 1, we select the Gaussian parameters for Rand and $\mathsf{Mult}^{\mathsf{E}}$.

n	$\log t$	$\log q$
2^{12}	23	54×2
2^{13}		47×4

Rand		$\mathsf{Mult}^{\mathsf{E}}$	
σ_r	τ_r	$\sigma_1 = \sigma_2$	$\tau_1 = \tau_2$
$2^{6.51}$	$2^{12.85}$	$2^{28.60}$	$2^{44.18}$

Table 3. Parameters for the standard and ciphertext-simulatable BFV scheme.

Using the $n = 2^{12}$ parameters for the base BFV scheme and the Gaussian parameters from Table 3, the estimated error bound for the output ciphertext of our ciphertext-simulatable BFV scheme is $\approx 2^{80.76}$, which gives a sufficiently low decryption failure rate. We also note that for the standard evaluation using the same $n = 2^{12}$ parameters results in the error bound of $\approx 2^{80.58}$, which is nearly identical to the randomized counterpart. In other words, randomized operations have negligible overhead on the error growth. On the other hand, the standard BFV evaluation with $n = 2^{13}$ parameters results in the error bound of $\approx 2^{75.58}$. Thus, around 80 bits of noise flooding is possible.

7.2 Performance Analysis

In this subsection, we analyze the performance of our randomized BFV scheme and compare it to the standard BFV scheme, with and without the noise flood-

ing. We implemented the standard BFV scheme, as well as our ciphertext-simulatable variant, in Julia. All experiments were performed in a single thread on a machine with an Intel Xeon Platinum 8268 CPU running at 2.90GHz. Our source code is available at `https://github.com/SNUCP/simct`.

Operation	Standard	Ours	Noise Flooding
Add	0.02ms	0.02ms	0.12ms
PMult	0.29ms	1.05ms	0.95ms
Mult	4.13ms	9.56ms	20.59ms
Aut	0.66ms	1.95ms	4.61ms
Rand	-	1.24ms	-

Table 4. Elapsed time for the standard BFV scheme with and without the noise flooding, our ciphertext-simulatable BFV scheme. Columns 'Standard' and 'Ours' used $n = 2^{12}$, and 'Noise Flooding' used $n = 2^{13}$.

The elapsed time for each operation of our scheme, standard BFV scheme with and without noise flooding is presented in Table 4. The operations $\mathsf{PMult}^{\mathsf{E}}, \mathsf{Mult}^{\mathsf{E}}$ and $\mathsf{Aut}^{\mathsf{E}}$ are 3.62, 2.31 and 2.95 times slower than $\mathsf{PMult}, \mathsf{Mult}$ and Aut with the same base parameters, respectively. On the other hand, $\mathsf{Mult}^{\mathsf{E}}$ and $\mathsf{Aut}^{\mathsf{E}}$ are 2.17 and 2.36 times faster than PMult and Aut with noise flooding parameters, while $\mathsf{PMult}^{\mathsf{E}}$ and PMult are nearly identical.

7.3 Optimized Circuit Representation

Before presenting the benchmark results for the full circuit evaluation, we explore methods to reduce the latency of randomized evaluation by selecting optimal circuit representation. In a nutshell, we carefully identify operations that do not need randomization and compute them in a deterministic way.

For example, when computing a sequence of operations without any plaintext or ciphertext multiplications (e.g. lines 3–5 of C_{ℓ_2}), it is unnecessary to randomize them. This is because plaintext and ciphertext multiplication is the only operation where the message or server's private inputs get entangled with other ciphertext components. Therefore, we can simply add a ciphertext randomizer and compute them in a deterministic way. Then, the input ciphertexts of this subcircuit become simulatable except for the message, and the simulator can simply run the operations with the simulated ciphertexts and extract the error.

Moreover, we can find an alternative circuit representation that contains less randomized plaintext or ciphertext multiplications. In the initial lines (1–2) of C_{ℓ_2}, the server calculates $(x - y)^2$ using the client's input x. When computed directly, this requires a randomized ciphertext-ciphertext multiplication, which consists of two Rand and $\mathsf{PMult}^{\mathsf{E}}$ operations, because $x - y$ includes information

from the server. However, evaluating $(x-y)^2$ as $x^2-2xy+y^2$ allows us to perform a single plaintext-ciphertext multiplication and a deterministic ciphertext-ciphertext multiplication instead. In other words, it is evident that these two evaluation paths lead to ciphertexts encrypting identical messages:

$$\mathsf{Mult}(\mathsf{Add}(\mathbf{c},-y),\mathsf{Add}(\mathbf{c},-y)), \quad \mathsf{PAdd}(\mathsf{Add}(\mathsf{Mult}(\mathbf{c},\mathbf{c}),\mathsf{PMult}(\mathbf{c},-2y)),y^2).$$

Given that the simulator has knowledge of $\mathbf{c}$ and sk, it can independently calculate the error in $\mathsf{Mult}(\mathbf{c},\mathbf{c})$. Thus, only a single $\mathsf{PMult}^{\mathsf{E}}$ operation is needed, and we can reduce the computation by 2 Rand and 1 $\mathsf{PMult}^{\mathsf{E}}$ operations. In conclusion, we are able to develop a ciphertext-simulatable evaluation algorithm that replicates the functionality of C_{ℓ_2}, together with its corresponding ciphertext simulator, as follows.

$\mathsf{Eval}^{\mathsf{C}}(\mathsf{evk}, C'_{\ell_2}, \mathbf{c}, y)$

$\mathbf{c}_{\mathrm{pmult}} \leftarrow \mathsf{PMult}^{\mathsf{E}}_{\sigma,\tau}(\mathbf{c}, -2y)$
$\mathbf{c}_{\mathrm{mult}} \leftarrow \mathsf{Mult}(\mathsf{rlk}, \mathbf{c}, \mathbf{c})$
$\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Add}(\mathbf{c}_{\mathrm{mult}}, \mathbf{c}_{\mathrm{pmult}})$
$\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{PAdd}(\mathbf{c}_{\mathrm{out}}, y^2)$
$\mathbf{c}_{\mathrm{out}} \leftarrow \mathbf{c}_{\mathrm{out}} + \mathsf{Rand}(\mathsf{pk})$
for $i \in 1, \ldots, \log d$
  $\mathbf{c}_{\mathrm{rot}} \leftarrow \mathsf{Aut}(\mathsf{atk}_{n/2^i}, \mathbf{c}_{\mathrm{out}}, n/2^i)$
  $\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Add}(\mathbf{c}_{\mathrm{out}}, \mathbf{c}_{\mathrm{rot}})$
return $\mathbf{c}_{\mathrm{out}}$

$\mathsf{Sim}^{\mathsf{C}}(\mathsf{sk}, \mathsf{evk}, \mathbf{c}, z)$

$e \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathbf{c})$
$e_{\mathrm{pmult}} \leftarrow \mathcal{D}^{\sigma,\tau}_{\mathrm{pmult}}(e)$
$e_{\mathrm{mult}} \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathsf{Mult}(\mathsf{rlk}, \mathbf{c}, \mathbf{c}))$
$e_{\mathrm{out}} \leftarrow e_{\mathrm{mult}} + e_{\mathrm{pmult}}$
$\mathbf{c}_{\mathrm{out}} \leftarrow (e_{\mathrm{out}}, 0) + \mathsf{Rand}(\mathsf{pk})$
for $i \in 1, \ldots, \log d$
  $\mathbf{c}_{\mathrm{rot}} \leftarrow \mathsf{Aut}(\mathsf{atk}_{n/2^i}, \mathbf{c}_{\mathrm{out}}, n/2^i)$
  $\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{Add}(\mathbf{c}_{\mathrm{out}}, \mathbf{c}_{\mathrm{rot}})$
return $\mathbf{c}_{\mathrm{out}} + (\Delta z, 0)$

Table 5 presents a summary of the benchmark results for the full circuit. The ciphertext-simulatable BFV scheme using a naïve evaluation method takes 24.47 ms, making it 2.52 times slower than the standard BFV scheme, yet still 2.86 times quicker than noise flooding. In contrast, the optimized ciphertext-simulatable evaluation requires only 12.90 ms, almost twice as fast as the naïve method. This optimized approach is just 1.32 times slower than the standard scheme and is 7.20 times faster than noise flooding.

	Standard	Ours (Optimized)	Ours	Noise Flooding
Total	9.72ms	12.90ms	24.47ms	70.01ms

Table 5. Elapsed time for the standard BFV scheme with and without the noise flooding, our ciphertext-simulatable BFV scheme.

References

1. Albrecht, M.R., Davidson, A., Deo, A., Gardham, D.: Crypto dark matter on the torus - oblivious PRFs from shallow PRFs and TFHE. In: Joye, M., Leander, G.

(eds.) Advances in Cryptology – EUROCRYPT 2024, Part VI. Lecture Notes in Computer Science, vol. 14656, pp. 447–476. Springer, Cham, Switzerland, Zurich, Switzerland (May 26–30, 2024). https://doi.org/10.1007/978-3-031-58751-1_16
2. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3), 169–203 (2015)
3. Alperin-Sheriff, J., Peikert, C.: Faster bootstrapping with polynomial error. In: Advances in Cryptology–CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I 34. pp. 297–314. Springer (2014)
4. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold fhe. In: Advances in Cryptology–EUROCRYPT 2012: 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings 31. pp. 483–501. Springer (2012)
5. Asi, H., Boemer, F., Genise, N., Mughees, M.H., Ogilvie, T., Rishi, R., Rothblum, G.N., Talwar, K., Tarbe, K., Zhu, R., et al.: Scalable private search with wally. arXiv preprint arXiv:2406.06761 (2024)
6. Baldimtsi, F., Cheng, J., Goyal, R., Yadav, A.: Non-interactive blind signatures: Post-quantum and stronger security. In: Chung, K.M., Sasaki, Y. (eds.) Advances in Cryptology – ASIACRYPT 2024, Part II. Lecture Notes in Computer Science, vol. 15485, pp. 70–104. Springer, Singapore, Singapore, Kolkata, India (December 9–13, 2024). https://doi.org/10.1007/978-981-96-0888-1_3
7. Beaver, D.: Efficient multiparty protocols using circuit randomization. In: Advances in Cryptology—CRYPTO'91: Proceedings 11. pp. 420–432. Springer (1992)
8. Bourse, F., Del Pino, R., Minelli, M., Wee, H.: Fhe circuit privacy almost for free. In: Annual International Cryptology Conference. pp. 62–89. Springer (2016)
9. Bourse, F., Izabachène, M.: Plug-and-play sanitization for tfhe. Cryptology ePrint Archive (2022)
10. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapsvp. In: Annual Cryptology Conference. pp. 868–886. Springer (2012)
11. de Castro, L., Kim, D., Kim, M., Lee, K., Min, S., Song, Y.: More efficient lattice-based OLE from circuit-private linear HE with polynomial overhead. Cryptology ePrint Archive, Paper 2024/1534 (2024), https://eprint.iacr.org/2024/1534
12. Chen, H., Chillotti, I., Dong, Y., Poburinnaya, O., Razenshteyn, I., Riazi, M.S.: {SANNS}: Scaling up secure approximate {k-Nearest} neighbors search. In: 29th USENIX Security Symposium (USENIX Security 20). pp. 2111–2128 (2020)
13. Chen, H., Han, K.: Homomorphic lower digits removal and improved fhe bootstrapping. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 315–337. Springer (2018)
14. Cong, K., Moreno, R.C., da Gama, M.B., Dai, W., Iliashenko, I., Laine, K., Rosenberg, M.: Labeled psi from homomorphic encryption with reduced computation and communication. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. pp. 1135–1150 (2021)
15. Döttling, N., Dujmovic, J.: Maliciously circuit-private fhe from information-theoretic principles. Cryptology ePrint Archive (2022)
16. Ducas, L., Stehlé, D.: Sanitization of fhe ciphertexts. In: Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I 35. pp. 294–310. Springer (2016)

17. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive (2012)
18. Geelen, R., Iliashenko, I., Kang, J., Vercauteren, F.: On polynomial functions modulo pe and faster bootstrapping for homomorphic encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 257–286. Springer (2023)
19. Genise, N., Micciancio, D., Peikert, C., Walter, M.: Improved discrete gaussian and subgaussian analysis for lattice cryptography. In: IACR International Conference on Public-Key Cryptography. pp. 623–651. Springer (2020)
20. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the forty-first annual ACM symposium on Theory of computing. pp. 169–178 (2009)
21. Halevi, S., Shoup, V.: Bootstrapping for helib. Journal of Cryptology **34**(1), 7 (2021)
22. Hwang, I., Lee, H., Seo, J., Song, Y.: Practical zero-knowledge PIOP for public key and ciphertext generation in (multi-group) homomorphic encryption. Cryptology ePrint Archive, Paper 2024/1879 (2024), `https://eprint.iacr.org/2024/1879`
23. Kim, A., Polyakov, Y., Zucca, V.: Revisiting homomorphic encryption schemes for finite fields. In: Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part III 27. pp. 608–639. Springer (2021)
24. Kim, D., Lee, D., Seo, J., Song, Y.: Toward practical lattice-based proof of knowledge from hint-mlwe. Cryptology ePrint Archive (2023)
25. Kluczniak, K.: Circuit privacy for fhew/tfhe-style fully homomorphic encryption in practice. Cryptology ePrint Archive (2022)
26. Lee, J.W., Kang, H., Lee, Y., Choi, W., Eom, J., Deryabin, M., Lee, E., Lee, J., Yoo, D., Kim, Y.S., et al.: Privacy-preserving machine learning with fully homomorphic encryption for deep neural network. iEEE Access **10**, 30039–30054 (2022)
27. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing **37**(1), 267–302 (2007)
28. Micciancio, D., Walter, M.: Gaussian sampling over the integers: Efficient, generic, constant-time. In: Advances in Cryptology–CRYPTO 2017: 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20–24, 2017, Proceedings, Part II 37. pp. 455–485. Springer (2017)
29. Mughees, M.H., Chen, H., Ren, L.: Onionpir: Response efficient single-server pir. In: Proceedings of the 2021 ACM SIGSAC conference on computer and communications security. pp. 2292–2306 (2021)
30. Ostrovsky, R., Paskin-Cherniavsky, A., Paskin-Cherniavsky, B.: Maliciously circuit-private fhe. In: Annual Cryptology Conference. pp. 536–553. Springer (2014)

A Additional Preliminaries

A.1 Probability Distributions

For a probability distribution $\mathcal{X}$, we denote sampling x from $\mathcal{X}$ as $x \leftarrow \mathcal{X}$. When x is sampled from a uniform distribution over a set S, we denote $x \xleftarrow{\$} S$.

For any two distributions $\mathcal{X}_1$ and $\mathcal{X}_2$ with the same support Ω, we define the max-log distance [28] as $\Delta_{\mathrm{ML}}(\mathcal{X}_1, \mathcal{X}_2) = \sup_{x\in\Omega}|\log \mathcal{X}_1(x) - \log \mathcal{X}_2(x)|$, and the statistical distance as $\Delta_{\mathrm{SD}}(\mathcal{X}_1, \mathcal{X}_2) = \frac{1}{2}\sum_{x\in\Omega}|\mathcal{X}_1(x) - \mathcal{X}_2(x)|$. Note that the max-log distance is a stronger metric than the statistical distance, that is, $\Delta_{\mathrm{ML}}(\mathcal{X}_1, \mathcal{X}_2) \le \log(1+\varepsilon)$ implies $\Delta_{\mathrm{SD}}(\mathcal{X}_1, \mathcal{X}_2) \le \varepsilon/2$ [19]. Therefore, we will exclusively use the max-log distance, as one can translate our results to statistical distance using this relationship.

We denote $\mathcal{X}_1 \overset{\varepsilon}{\approx} \mathcal{X}_2$ when $\Delta_{\mathrm{SD}}(\mathcal{X}_1, \mathcal{X}_2) \le \varepsilon$. If ε is negligible, we say $\mathcal{X}_1$ and $\mathcal{X}_2$ are statistically indistinguishable. When there is no PPT algorithm that can distinguish $\mathcal{X}_1$ and $\mathcal{X}_2$ with non-negligible probability, we say $\mathcal{X}_1$ and $\mathcal{X}_2$ are computationally indistinguishable, and we write as $\mathcal{X}_1 \approx_c \mathcal{X}_2$.

For a vector $\mathbf{c} \in \mathbb{R}^n$ and a positive definite matrix $\mathbf{\Sigma} \in \mathbb{R}^{n\times n}$, we define the Gaussian function as $\rho_{\mathbf{c},\sqrt{\mathbf{\Sigma}}} = \exp(-\pi(\mathbf{x}-\mathbf{c})^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{c}))$. We also define (discrete) Gaussian distribution over $\mathbf{a}+\Lambda$ and continuous Gaussian distribution with center $\mathbf{c}$ and parameter $\sqrt{\mathbf{\Sigma}}$ as

$$\mathcal{D}_{\mathbf{a}+\Lambda,\mathbf{c},\sqrt{\mathbf{\Sigma}}}(\mathbf{x}) = \frac{\rho_{\mathbf{c},\sqrt{\mathbf{\Sigma}}}(\mathbf{x})}{\sum_{\mathbf{v}\in\mathbf{a}+\Lambda}\rho_{\mathbf{c},\sqrt{\mathbf{\Sigma}}}(\mathbf{v})}, \quad \mathcal{D}_{\mathbb{R}^n,\mathbf{c},\sqrt{\mathbf{\Sigma}}}(\mathbf{x}) = \frac{\rho_{\mathbf{c},\sqrt{\mathbf{\Sigma}}}(\mathbf{x})}{\sqrt{\det\mathbf{\Sigma}}}$$

respectively. When writing Gaussian functions and distributions, if $\mathbf{\Sigma} = \sigma^2\mathbf{I}$, we simply write σ instead of $\sqrt{\mathbf{\Sigma}}$, and if $\mathbf{c} = 0$, we omit the center.

A.2 Linear Algebra

A symmetric matrix $\mathbf{M} \in \mathbb{R}^{n\times n}$ is called positive definite and denoted as $\mathbf{M} \succ 0$ if $\mathbf{x}^\top\mathbf{M}\mathbf{x} > 0$ for all $\mathbf{x} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$. A square root of $\mathbf{M}$ is a matrix $\mathbf{A}$ such that $\mathbf{A}\mathbf{A}^\top = \mathbf{M}$. Note that such $\mathbf{A}$ always exists, but may not be unique. We write $\mathbf{A} = \sqrt{\mathbf{M}}$ when the specific choice of square root is unimportant.

Similarly, a symmetric matrix $\mathbf{M}$ is positive semidefinite if $\mathbf{x}^\top\mathbf{M}\mathbf{x} \ge 0$ for all $\mathbf{x} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$, and we denote it as $\mathbf{M} \succeq 0$. We write $\mathbf{B} \ge \mathbf{A}$ when $\mathbf{B}\mathbf{B}^\top - \mathbf{A}\mathbf{A}^\top \succeq 0$.

For any vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$, we define the p-norm of $\mathbf{x}$ as

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n}|x_i|^p\right)^{\frac{1}{p}}, \quad \|\mathbf{x}\|_\infty = \max_i|x_i|.$$

We also define the p-norm of a ring element a as the p-norm of its coefficient vector. Similarly, for any matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$, we define the p-norm of $\mathbf{M}$ as

$$\|\mathbf{M}\|_p = \max_{\mathbf{x}\in\mathbb{R}^n\setminus\{\mathbf{0}\}}\frac{\|\mathbf{M}\mathbf{x}\|_p}{\|\mathbf{x}\|_p}, \quad \|\mathbf{M}\|_\infty = \max_i\sum_{j=1}^{n}|\mathbf{M}_{ij}|.$$

A.3 Ring Learning with Errors

Definition 10 (Ring-LWE). *Let m, n, q be positive integers such that N is a power of two. Let χ_s, χ_e be a distribution over R. Then, the Ring-LWE(RLWE) problem $\mathsf{RLWE}_{m,n,q,\chi_s,\chi_e}$ is to distinguish the following two distributions:*

$$\{(\mathbf{a}, \mathbf{a}s + \mathbf{e}) \mid \mathbf{a} \xleftarrow{\$} R_q^m, s \leftarrow \chi_s, \mathbf{e} \leftarrow \chi_e^m\},\ \{(\mathbf{a}, \mathbf{u}) \mid \mathbf{a}, \mathbf{u} \xleftarrow{\$} R_q^m\}.$$

When $\chi_s = \chi_e = \mathcal{D}_{\mathbb{Z}^n,\sigma}$, we simply write $\mathsf{RLWE}_{m,n,q,\sigma}$.

A.4 Gadget Decomposition

Definition 11 (Gadget Decomposition). *For a positive integer q, a function $h : R_q \to R^d$ is called a gadget decomposition with respect to a gadget vector $\mathbf{g} \in R_q^d$ and a bound $B_{\mathbf{g}} > 0$, if for every $a \in R_q$, the following holds:*

$$\langle h(a), \mathbf{g} \rangle = a \pmod q \quad \textit{and} \quad \|h(a)\|_\infty \leq B_{\mathbf{g}}.$$

We also call $h(a)$ the gadget decomposition of a.

Definition 12 (Gadget Product). *For $a \in R_q$ and $(\mathbf{u}_0, \mathbf{u}_1) \in R_q^{d\times 2}$, the gadget product $\boxdot : R_q \times R_q^{d\times 2} \to R_q^2$ between a and $(\mathbf{u}_0, \mathbf{u}_1)$ is defined as*

$$a \boxdot (\mathbf{u}_0, \mathbf{u}_1) = (\langle h(a), \mathbf{u}_0 \rangle, \langle h(a), \mathbf{u}_1 \rangle) \pmod q.$$

In the BFV scheme, the gadget product is primarily used for the key-switching procedure. If $(\mathbf{u}_0, \mathbf{u}_1) \in R_q^{d\times 2}$ satisfies $\mathbf{u}_0 + \mathbf{u}_1 s = \mathbf{g}s' + \mathbf{e} \pmod q$ for some $s, s' \in R$ and small $\mathbf{e} \in R^d$, then the gadget product $(c_0, c_1) \leftarrow a \boxdot (\mathbf{u}_0, \mathbf{u}_1)$ satisfies $c_0 + c_1 s = \langle h(a), \mathbf{u}_0 + \mathbf{u}_1 s \rangle = as' + \langle h(a), \mathbf{e} \rangle \pmod q$. Here the error $e' = \langle h(a), \mathbf{e} \rangle$ is bounded by $\|e'\|_\infty \leq ndB_{\mathbf{g}}\|\mathbf{e}\|_\infty$.

B Missing Proofs

B.1 Proof of Lemma 4

Lemma 11 ([24, Adapted from Lemma 7]). *For $\mathbf{E} \in \mathbb{R}^{m\times n}$ and reals $\sigma, \tau > 0$, consider the distribution*

$$\{(\mathbf{r}, \mathbf{z}) \mid \mathbf{r} \leftarrow \mathcal{D}_{\mathbb{Z}^n,\sigma}, \mathbf{y} = \mathcal{D}_{\mathbb{R}^m,\tau}, \mathbf{z} = \mathbf{E}\mathbf{r} + \mathbf{y}\}.$$

Then, the conditional distribution of $\mathbf{r}$ given any $\mathbf{z}$ equals $\mathcal{D}_{\mathbb{Z}^n,\tau^{-2}\boldsymbol{\Sigma}\mathbf{E}^\top\mathbf{z},\sqrt{\boldsymbol{\Sigma}}}$ where $\boldsymbol{\Sigma}^{-1} := \sigma^{-2}\mathbf{I} + \tau^{-2}\mathbf{E}^\top\mathbf{E}$.

Proof. Suppose $\mathbf{z}$ is fixed. By simple computation, we have

$$\Pr_{\substack{\mathbf{r}\leftarrow\mathcal{D}_{\mathbb{Z}^n,\sigma}\\ \mathbf{y}\leftarrow\mathcal{D}_{\mathbb{R}^m,\tau}}}[\mathbf{r} = \mathbf{v} \mid \mathbf{E}\mathbf{r} + \mathbf{y} = \mathbf{z}] = \mathcal{D}_{\mathbb{Z}^n,\sigma}(\mathbf{v}) \cdot \mathcal{D}_{\mathbb{R}^m,\tau}(\mathbf{z} - \mathbf{E}\mathbf{v})$$

$$
\begin{aligned}
&\propto \exp\left[-\pi\left(\frac{1}{\sigma^2}\mathbf{v}^\top\mathbf{v} + \frac{1}{\tau^2}(\mathbf{z}-\mathbf{E}\mathbf{v})^\top(\mathbf{z}-\mathbf{E}\mathbf{v})\right)\right] \\
&\propto \exp\left[-\pi(\mathbf{v}-\tau^{-2}\boldsymbol{\Sigma}\mathbf{E}^\top\mathbf{z})^\top\boldsymbol{\Sigma}^{-1}(\mathbf{v}-\tau^{-2}\boldsymbol{\Sigma}\mathbf{E}^\top\mathbf{z})\right] \\
&\propto \rho_{\sqrt{\boldsymbol{\Sigma}}}(\mathbf{v}-\tau^{-2}\boldsymbol{\Sigma}\mathbf{E}^\top\mathbf{z}).
\end{aligned}
$$

Therefore, the conditional distribution of $\mathbf{r}$ follows $\mathcal{D}_{\mathbb{Z}^n,\tau^{-2}\boldsymbol{\Sigma}\mathbf{E}^\top\mathbf{z},\sqrt{\boldsymbol{\Sigma}}}$. □

Proof. Since

$$
\mathbf{c}_{\text{rand}} = e_2\cdot\mathsf{pk}+(e_0,e_1) = (-(p_1e_2+e_1)s+e_{\mathsf{pk}}e_2+se_1+e_0, p_1e_2+e_1) \pmod q,
$$

a sample $(\mathsf{sk},\mathsf{evk},\mathbf{c}_{\text{rand}})$ is equivalent to $(s,p_1,e_{\mathsf{pk}},p_1e_2+e_1,e_{\mathsf{pk}}e_2+se_1+e_0)$. Now, we claim that

$$
\left\{(p_1,\left[\mathbf{I}\ \mathbf{P}_1\right]\hat{\mathbf{e}},\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0') \;\middle|\; \begin{array}{r} p_1\xleftarrow{\$}R_q \\ \hat{\mathbf{e}}\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\sigma} \\ \mathbf{e}_0'\leftarrow\mathcal{D}_{\mathbb{R}^n,\tau}\end{array}\right\} \approx_c \left\{(p_1,u,\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0') \;\middle|\; \begin{array}{r} p_1,u\xleftarrow{\$}R_q \\ \hat{\mathbf{e}}\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\sigma} \\ \mathbf{e}_0'\leftarrow\mathcal{D}_{\mathbb{R}^n,\tau}\end{array}\right\}
$$

for $\hat{\mathbf{e}} = (\mathbf{e}_1,\mathbf{e}_2)$ and $\boldsymbol{\Gamma} = \left[\mathbf{S}\ \mathbf{E}_{\mathsf{pk}}\right]$, where $\mathbf{e}_0',\mathbf{e}_1,\mathbf{e}_2$ are coefficient vectors of e_0',e_1,e_2 and $\mathbf{P}_1,\mathbf{S},\mathbf{E}_{\mathsf{pk}}$ are negacyclic matrix corresponding to p_1,s,e_{pk} respectively. By Lemma 11,

$$
\left\{(p_1,\left[\mathbf{I}\ \mathbf{P}_1\right]\hat{\mathbf{e}},\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0') \;\middle|\; \begin{array}{r} p_1\xleftarrow{\$}R_q \\ \hat{\mathbf{e}}\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\sigma} \\ \mathbf{e}_0'\leftarrow\mathcal{D}_{\mathbb{R}^n,\tau}\end{array}\right\} \equiv \left\{(p_1,\left[\mathbf{I}\ \mathbf{P}_1\right]\hat{\mathbf{e}}',\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0') \;\middle|\; \begin{array}{r} p_1\xleftarrow{\$}R_q \\ \hat{\mathbf{e}}\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\sigma} \\ \mathbf{e}_0'\leftarrow\mathcal{D}_{\mathbb{R}^n,\tau} \\ \hat{\mathbf{e}}'\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\mathbf{c},\sqrt{\boldsymbol{\Sigma}'}}\end{array}\right\}
$$

where $\boldsymbol{\Sigma}'^{-1} := \sigma^{-2}\mathbf{I}+\tau^{-2}\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma}$ and $\mathbf{c} = \tau^{-2}\boldsymbol{\Sigma}'\boldsymbol{\Gamma}^\top(\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0')$. For further analysis, we decompose the distribution of $\hat{\mathbf{e}}'$ using Lemma 3, as

$$
\mathcal{D}_{\mathbb{Z}^{2n},\mathbf{c},\sqrt{\boldsymbol{\Sigma}'}} \overset{\hat{\varepsilon}}{\approx} \mathcal{D}_{\mathbb{Z}^{2n},\frac{\kappa}{\sqrt{2}}} + \mathcal{D}_{\mathbb{Z}^{2n},\mathbf{c},\sqrt{\boldsymbol{\Sigma}'-\frac{\kappa^2}{2}\mathbf{I}}}
$$

where $\hat{\varepsilon} = \log\frac{1+\varepsilon}{1-\varepsilon}$. Note that

$$
\boldsymbol{\Sigma}'^{-1} \preceq \|\boldsymbol{\Sigma}'^{-1}\|_2\mathbf{I} \preceq (\sigma^{-2}+\tau^{-2}\|\boldsymbol{\Gamma}\|_2^2)\mathbf{I} \preceq \kappa^{-2}\mathbf{I}
$$

so $\boldsymbol{\Sigma}' \succeq \kappa^2\mathbf{I}$. Hence, $\boldsymbol{\Sigma}'-\frac{\kappa^2}{2}\mathbf{I}$ is positive definite, and $\sqrt{\left(\frac{\kappa^2}{2}\right)^{-1}\mathbf{I}+(\boldsymbol{\Sigma}'-\frac{\kappa^2}{2}\mathbf{I})^{-1}}^{-1} \geq \frac{\kappa}{2} \geq \eta_\varepsilon(\mathbb{Z}^{2n})$, which implies that this decomposition is valid. We conclude that

$$
\begin{aligned}
&\left\{(p_1,\left[\mathbf{I}\ \mathbf{P}_1\right]\hat{\mathbf{e}}',\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0') \;\middle|\; \begin{array}{r} p_1\xleftarrow{\$}R_q \\ \hat{\mathbf{e}}\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\sigma},\mathbf{e}_0'\leftarrow\mathcal{D}_{\mathbb{R}^n,\tau} \\ \hat{\mathbf{e}}'\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\mathbf{c},\sqrt{\boldsymbol{\Sigma}'}}\end{array}\right\} \\
&\quad\overset{\hat{\varepsilon}}{\approx}\left\{(p_1,\left[\mathbf{I}\ \mathbf{P}_1\right](\hat{\mathbf{e}}_1'+\hat{\mathbf{e}}_2'),\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0') \;\middle|\; \begin{array}{r} p_1\xleftarrow{\$}R_q \\ \hat{\mathbf{e}}\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\sigma},\mathbf{e}_0'\leftarrow\mathcal{D}_{\mathbb{R}^n,\tau} \\ \hat{\mathbf{e}}_1'\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\frac{\kappa}{\sqrt{2}}},\hat{\mathbf{e}}_2'\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\mathbf{c},\sqrt{\boldsymbol{\Sigma}'-\frac{\kappa^2}{2}\mathbf{I}}}\end{array}\right\} \\
&\quad\approx_c\left\{(p_1,\mathbf{u}+\left[\mathbf{I}\ \mathbf{P}_1\right]\hat{\mathbf{e}}_2',\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}_0') \;\middle|\; \begin{array}{r} p_1\xleftarrow{\$}R_q,\mathbf{u}\xleftarrow{\$}\mathbb{Z}_q^n \\ \hat{\mathbf{e}}\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\sigma},\mathbf{e}_0\leftarrow\mathcal{D}_{\mathbb{R}^n,\tau} \\ \hat{\mathbf{e}}_2'\leftarrow\mathcal{D}_{\mathbb{Z}^{2n},\mathbf{c},\sqrt{\boldsymbol{\Sigma}'-\frac{\kappa^2}{2}\mathbf{I}}}\end{array}\right\}
\end{aligned}
$$

$$\equiv \left\{ (p_1, u, \boldsymbol{\Gamma}\hat{\mathbf{e}} + \mathbf{e}'_0) \;\middle|\; \begin{array}{c} p_1, u \xleftarrow{\$} R_q \\ \hat{\mathbf{e}} \leftarrow \mathcal{D}_{\mathbb{Z}^{2n},\sigma}, \mathbf{e}'_0 \leftarrow \mathcal{D}_{\mathbb{R}^n,\tau} \end{array} \right\}$$

where the last computational indistinguishability comes from the hardness of $\mathsf{RLWE}_{1,n,q,\kappa/\sqrt{2}}$. The lemma follows by rounding $\boldsymbol{\Gamma}\hat{\mathbf{e}}+\mathbf{e}'_0$ from both distributions.

Finally, notice that $\|\boldsymbol{\Gamma}\|_2^2 \le n^2 + \|e_{\mathsf{pk}}\|_2^2 \le n^2 + K^2 n^2 \sigma_{\mathrm{err}}^2$ by Lemma 6, and

$$\frac{1}{\sigma^2} + \frac{n^2 + K^2 n^2 \sigma_{\mathrm{err}}^2}{\tau^2} \le \frac{1}{4\eta_\varepsilon(\mathbb{Z}^{2n})^2} \le \frac{1}{\eta_\varepsilon(\mathbb{Z}^n)^2}.$$

Therefore, by Lemma 5,

$$\boldsymbol{\Gamma}\hat{\mathbf{e}} + \mathbf{e}_0 \sim \boldsymbol{\Gamma}\mathcal{D}_{\mathbb{Z}^{2n},\sigma} + \lfloor \mathcal{D}_{\mathbb{R}^n,\tau} \rceil \overset{\tilde{\varepsilon}}{\approx} \lfloor \mathcal{D}_{\mathbb{R}^n,\sqrt{\sigma^2 \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \tau^2 \mathbf{I}}} \rceil \equiv \lfloor \mathcal{D}_{\mathbb{R}^n,\sqrt{\boldsymbol{\Sigma}}} \rceil$$

where $\boldsymbol{\Sigma} = \sigma^2(\mathbf{S}\mathbf{S}^\top + \mathbf{E}_{\mathsf{pk}}\mathbf{E}_{\mathsf{pk}}^\top) + \tau^2\mathbf{I}$.

Moreover, since $\sigma_{\mathrm{err}} = 3.2\sqrt{2\pi} \ge \eta_\varepsilon(\mathbb{Z}^n)$, by Lemma 6, $\|e_{\mathsf{pk}}\|_2 \le \sqrt{n}\|e_{\mathsf{pk}}\|_\infty \le K\sqrt{n}\sigma_{\mathrm{err}}$ with overwhelming probability. Then, since every diagonal entry of $\boldsymbol{\Sigma} = \sigma^2(\mathbf{E}_{\mathsf{pk}}\mathbf{E}_{\mathsf{pk}}^\top + \mathbf{S}\mathbf{S}^\top) + \tau^2\mathbf{I}$ satisfies

$$\boldsymbol{\Sigma}_{i,i} = \sigma^2(\|e_{\mathsf{pk}}\|_2^2 + \|s\|_2^2) + \tau^2 \le \sigma^2(n + K^2 n\sigma_{\mathrm{err}}^2) + \tau^2,$$

the bound follows from Lemma 6. □

B.2 Proof of Theorem 2

Proof. We proceed by the hybrid argument, where $\mathbf{Hyb}_0$ is the distribution of $\mathsf{Eval}^{\mathsf{C}}$.

$\mathbf{Hyb}_1$. In this hybrid, we sample $\mathbf{c}_{\mathrm{rand}} = (c_{\mathrm{rand},0}, c_{\mathrm{rand},1})$ as $c_{\mathrm{rand},1} \xleftarrow{\$} R_q$, $c_{\mathrm{rand},0} \leftarrow -c_{\mathrm{rand},1}s + e_{\mathrm{rand}}$, where $e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}^{\sigma,\tau}$. $\mathbf{c}_{\mathrm{out}}, \mathbf{c}'_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_0$. By Lemma 4, $\mathbf{Hyb}_0$ and $\mathbf{Hyb}_1$ are computationally indistinguishable.

$\mathbf{Hyb}_2$. In this hybrid, we sample $\mathbf{c}'_{\mathrm{out}} = (c'_{\mathrm{out},0}, c'_{\mathrm{out},1})$ as $c'_{\mathrm{out},1} \xleftarrow{\$} R_q$, $c'_{\mathrm{out},0} \leftarrow -c'_{\mathrm{out},1}s + \Delta z + e_{\mathrm{rand}} + e_{\mathrm{out}}$ where $e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}^{\sigma,\tau}$, $e_{\mathrm{out}} \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{out}})$. $\mathbf{c}_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_1$, and $\mathbf{c}_{\mathrm{rand}}$ is removed. Notice that in $\mathbf{Hyb}_1$,

$$\begin{aligned} \mathbf{c}'_{\mathrm{out}} &= \mathbf{c}_{\mathrm{out}} + \mathbf{c}_{\mathrm{rand}} \\ &= (-(c_{\mathrm{out},1} + c_{\mathrm{rand},1})s + \Delta z + e_{\mathrm{rand}} + e_{\mathrm{out}}, c_{\mathrm{out},1} + c_{\mathrm{rand},1}) \pmod q \end{aligned}$$

where $c_{\mathrm{rand},1}$ follows a uniform distribution over R_q. Therefore, $c_{\mathrm{out},1} + c_{\mathrm{rand},1}$ also follows a uniform distribution over R_q, which is exactly the distribution of $c'_{\mathrm{out},1}$ in $\mathbf{Hyb}_2$. Therefore, $\mathbf{Hyb}_1$ and $\mathbf{Hyb}_2$ are identical.

$\mathbf{Hyb}_3$. In this hybrid, we sample $e_{\mathrm{out}} \leftarrow \mathsf{Sim}^{\mathsf{E}}(\mathsf{sk}, \mathsf{evk}, (e_1, \ldots, e_\ell))$ where $e_i = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_i)$ for $1 \le i \le \ell$. $e_{\mathrm{rand}}, \mathbf{c}'_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_2$, and $\mathbf{c}_{\mathrm{out}}$ is removed. By the error simulatability of $\mathsf{Eval}^{\mathsf{E}}$, $\mathbf{Hyb}_2$ and $\mathbf{Hyb}_3$ are computationally indistinguishable.

$\mathbf{Hyb}_4$. In this hybrid, we sample $\mathbf{c}_{\mathrm{rand}} = (c_{\mathrm{rand},0}, c_{\mathrm{rand},1})$ as $c_{\mathrm{rand},1} \xleftarrow{\$} R_q$, $c_{\mathrm{rand},0} \leftarrow -c_{\mathrm{rand},1}s + e_{\mathrm{rand}}$, and sample $\mathbf{c}'_{\mathrm{out}} \leftarrow (\Delta z + e_{\mathrm{out}}) + \mathbf{c}_{\mathrm{rand}}$. $e_{\mathrm{rand}}, e_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_3$. This hybrid is only a syntactic change, hence $\mathbf{Hyb}_3$ and $\mathbf{Hyb}_4$ are identical.

$\mathbf{Hyb}_5$. In this hybrid, we sample $\mathbf{c}_{\mathrm{rand}} \leftarrow \mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})$. $\mathbf{c}'_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_4$, and e_{rand} is removed. This hybrid is simply a reverse of $\mathbf{Hyb}_1$, hence $\mathbf{Hyb}_4$ and $\mathbf{Hyb}_5$ are computationally indistinguishable.

Finally, since $\mathbf{Hyb}_5$ is identical to the distribution of $\mathsf{Sim}^{\mathsf{C}}$, we conclude that $\mathsf{Eval}^{\mathsf{C}}$ is ciphertext-simulatable for C. □

B.3 Proof of Lemma 5

Lemma 12 (Gaussian Convolution). *Let $\mathbf{E} \in \mathbb{R}^{m\times n}$ a matrix and $\sigma, \tau > 0$ be reals. Let $\mathbf{c}_1 \in \mathbb{R}^n, \mathbf{c}_2 \in \mathbb{R}^m$ and $\mathbf{a}_1 + \Lambda_1 \subset \mathbb{R}^n, \mathbf{a}_2 + \Lambda_2 \subset \mathbb{R}^m$ be full-rank lattice cosets such that $\mathbf{E}\Lambda_1 \subseteq \Lambda_2$. If $\sigma^{-2} + \tau^{-2}\|\mathbf{E}\|_2^2 \le \eta_\varepsilon(\Lambda_1)^{-2}$ for some $0 < \varepsilon < 1$, then*

$$\Delta_{\mathrm{ML}}\left(\mathbf{E}\mathcal{D}_{\mathbf{a}_1+\Lambda_1,\mathbf{c}_1,\sigma} + \mathcal{D}_{\mathbf{a}_2+\Lambda_2,\mathbf{c}_2,\tau}, \mathcal{D}_{(\mathbf{E}\mathbf{a}_1+\mathbf{a}_2)+\Lambda_2,\mathbf{E}\mathbf{c}_1+\mathbf{c}_2,\sqrt{\sigma^2\mathbf{E}\mathbf{E}^\top+\tau^2\mathbf{I}}}\right) \le \log\frac{1+\varepsilon}{1-\varepsilon}.$$

Proof. Let $\hat{\mathbf{E}} = \begin{bmatrix}\mathbf{E}\ \mathbf{I}_m\end{bmatrix}$, $\hat{\mathbf{a}} = (\mathbf{a}_1, \mathbf{a}_2)$, $\hat{\Lambda} = \Lambda_1 \times \Lambda_2$, $\hat{\mathbf{c}} = (\mathbf{c}_0, \mathbf{c}_1)$, and $\hat{\mathbf{\Sigma}} = \begin{bmatrix}\sigma_1\mathbf{I}_n & \\ & \sigma_2\mathbf{I}_m\end{bmatrix}$. Then, we have

$$\begin{aligned}
\mathbf{E}\mathcal{D}_{\mathbf{a}_1+\Lambda_1,\mathbf{c}_1,\sqrt{\mathbf{\Sigma}_1}} + \mathcal{D}_{\mathbf{a}_2+\Lambda_2,\mathbf{c}_2,\sqrt{\mathbf{\Sigma}_2}} &\equiv \hat{\mathbf{E}}\mathcal{D}_{\hat{\mathbf{a}}+\hat{\Lambda},\hat{\mathbf{c}},\sqrt{\hat{\mathbf{\Sigma}}}} \\
&\overset{\hat{\varepsilon}}{\approx} \mathcal{D}_{\hat{\mathbf{E}}\hat{\mathbf{a}}+\hat{\mathbf{E}}\hat{\Lambda},\hat{\mathbf{E}}\hat{\mathbf{c}},\hat{\mathbf{E}}\sqrt{\hat{\mathbf{\Sigma}}}} \\
&\equiv \mathcal{D}_{(\mathbf{E}\mathbf{a}_1+\mathbf{a}_2)+\Lambda_2,\mathbf{E}\mathbf{c}_1+\mathbf{c}_2,\sqrt{\sigma_1^2\mathbf{E}\mathbf{E}^\top+\sigma_2^2\mathbf{I}}}
\end{aligned}$$

by Lemma 2 where $\hat{\varepsilon} = \log\frac{1+\varepsilon}{1-\varepsilon}$. This completes the proof under the condition that $\ker(\hat{\mathbf{E}})$ is a $\hat{\Lambda}$-subspace and $\eta_\varepsilon(\ker(\hat{\mathbf{E}}) \cap \hat{\Lambda}) \le \sqrt{\hat{\mathbf{\Sigma}}}$.

For the first condition, notice that $\ker(\hat{\mathbf{E}}) = \begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\mathbb{R}^n$. Therefore,

$$\mathrm{span}(\ker(\hat{\mathbf{E}}) \cap \hat{\Lambda}) = \mathrm{span}\left(\begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\Lambda_1\right) = \begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\mathrm{span}(\Lambda_1) = \begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\mathbb{R}^n$$

since Λ_1 is full-rank. This implies that $\ker(\hat{\mathbf{E}}) = \mathrm{span}(\ker(\hat{\mathbf{E}}) \cap \hat{\Lambda})$, hence $\ker(\hat{\mathbf{E}})$ is a $\hat{\Lambda}$-subspace as desired.

For the second condition, we need to show that $\eta_\varepsilon(\ker(\hat{\mathbf{E}}) \cap \hat{\Lambda}) = \eta_\varepsilon\left(\begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\Lambda_1\right) \le \sqrt{\hat{\mathbf{\Sigma}}}$, or equivalently $\eta_\varepsilon\left(\sqrt{\hat{\mathbf{\Sigma}}}^{-1}\begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\Lambda_1\right) \le 1$. Let $\mathbf{\Sigma} = (\sigma_1^{-2}\mathbf{I} + \sigma_2^{-2}\mathbf{E}^\top\mathbf{E})^{-1}$. For any $\mathbf{x} \in \Lambda_1$, we have

$$\left\|\sqrt{\hat{\mathbf{\Sigma}}}^{-1}\begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\mathbf{x}\right\|_2^2 = \mathbf{x}^\top\begin{bmatrix}\mathbf{I}_n & -\mathbf{E}^\top\end{bmatrix}\begin{bmatrix}\sigma_1^2\mathbf{I}_n & \\ & \sigma_2^2\mathbf{I}_m\end{bmatrix}\begin{bmatrix}\mathbf{I}_n \\ -\mathbf{E}\end{bmatrix}\mathbf{x}$$

$$= \mathbf{x}^\top(\sigma_1^{-2}\mathbf{I} + \sigma_2^{-2}\mathbf{E}^\top\mathbf{E})\mathbf{x}$$
$$= \|\sqrt{\mathbf{\Sigma}}^{-1}\mathbf{x}\|_2^2.$$

Hence, $\sqrt{\hat{\mathbf{\Sigma}}}^{-1}\begin{bmatrix}\mathbf{I}_n\\ -\mathbf{E}\end{bmatrix}\Lambda_1$ isometric to (more precisely, a rotation of) $\sqrt{\mathbf{\Sigma}}^{-1}\Lambda_1$. Since the smoothing parameter is invariant under rotation, we conclude that

$$\eta_\varepsilon\left(\sqrt{\hat{\mathbf{\Sigma}}}^{-1}\begin{bmatrix}\mathbf{I}_n\\ -\mathbf{E}\end{bmatrix}\Lambda_1\right) = \eta_\varepsilon\left(\sqrt{\mathbf{\Sigma}}^{-1}\Lambda_1\right) \le 1$$

where the last inequality comes from $\eta_\varepsilon(\Lambda_1) \le \sqrt{\sigma_1^{-2} + \sigma_2^{-2}\|\mathbf{E}\|_2^2}^{-1}\mathbf{I} \le \sqrt{\mathbf{\Sigma}}$. □

Proof. For any $p > 0$, we have

$$\mathbf{E}\mathcal{D}_{\mathbf{a}+\Lambda,\mathbf{c},\sigma} + \lfloor\mathcal{D}_{\frac{1}{p}\mathbb{Z}^m,\mathbf{c}',\tau}\rceil \equiv \lfloor\mathbf{E}\mathcal{D}_{\mathbf{a}+\Lambda,\mathbf{c},\sigma} + \mathcal{D}_{\frac{1}{p}\mathbb{Z}^m,\mathbf{c}',\tau}\rceil$$
$$\overset{\hat{\varepsilon}}{\approx} \left\lfloor\mathcal{D}_{\frac{1}{p}\mathbb{Z}^m,\mathbf{E}\mathbf{c}_1+\mathbf{c}_2,\sqrt{\sigma_1^2\mathbf{E}\mathbf{E}^\top+\sigma_2^2\mathbf{I}}}\right\rceil$$

by Lemma 12. The result follows by taking $p \to \infty$. □

B.4 Proof of Lemma 6

Proof. Recall that by the Chernoff bound of continuous Gaussian distribution, we have

$$\Pr_{x\leftarrow\mathcal{D}_{\mathbb{R},\sigma}}[|x| > k] \le 2e^{-\pi\frac{k^2}{\sigma^2}}$$

for any $k > 0$. This implies that when $\sigma \ge 1$,

$$\Pr_{x\leftarrow\mathcal{D}_{\mathbb{R},\sigma}}\left[|x| > k\sigma - \frac{1}{2}\right] \le 2e^{-\pi\frac{\left(k\sigma-\frac{1}{2}\right)^2}{\sigma^2}} \le 2e^{-\pi\left(k-\frac{1}{2}\right)^2}$$

since $\left(k\sigma - \frac{1}{2}\right)^2 \ge \left(k - \frac{1}{2}\right)^2\sigma^2$. Now, suppose $\mathbf{x} = (x_1, \ldots, x_n) \sim \mathcal{D}_{\mathbb{R}^n,\sqrt{\mathbf{\Sigma}}}$. Then, $x_i \sim \mathcal{D}_{\mathbb{R},\sqrt{\mathbf{\Sigma}_{i,i}}}$, and by the union bound, we get

$$\Pr_{\mathbf{x}\leftarrow\mathcal{D}_{\mathbb{R}^n,\sqrt{\mathbf{\Sigma}}}}\left[\|\mathbf{x}\|_\infty > k\max_i\sqrt{\mathbf{\Sigma}_{i,i}} - \frac{1}{2}\right] \le \Pr_{\mathbf{x}\leftarrow\mathcal{D}_{\mathbb{R}^n,\sqrt{\mathbf{\Sigma}}}}\left[\bigcup_i\left(|x_i| > k\sqrt{\mathbf{\Sigma}_{i,i}} - \frac{1}{2}\right)\right]$$
$$\le 2ne^{-\pi\left(k-\frac{1}{2}\right)^2}.$$

We conclude that

$$\Pr_{\mathbf{x}\leftarrow\left\lfloor\mathcal{D}_{\mathbb{R}^n,\sqrt{\mathbf{\Sigma}}}\right\rceil}[\|\mathbf{x}\|_\infty > k\max_i\sqrt{\mathbf{\Sigma}_{i,i}}] \le \Pr_{\mathbf{x}\leftarrow\mathcal{D}_{\mathbb{R}^n,\sqrt{\mathbf{\Sigma}}}}\left[\|\mathbf{x}\|_\infty > k\max_i\sqrt{\mathbf{\Sigma}_{i,i}} - \frac{1}{2}\right]$$
$$\le 2ne^{-\pi\left(k-\frac{1}{2}\right)^2}.$$

□

B.5 Proof of Lemma 8

Proof. Let $\mathbf{c}_{\mathrm{pmult}} = (c_{\mathrm{pmult},0}, c_{\mathrm{pmult},1})$. Since $\hat{\mu} \in \mu + t\mathbb{Z}^n$, we have

$$c_{\mathrm{pmult},0} + c_{\mathrm{pmult},1}s = \hat{\mu}(\Delta m + e) + \tilde{e} = \Delta\mu m + (\hat{\mu}e + \tilde{e}) \pmod q.$$

Moreover, since $\hat{\mu} \sim \mathcal{D}_{\mu+t\mathbb{Z}^n,\sigma}$, $\tilde{e} \sim \lfloor \mathcal{D}_{\mathbb{R}^n,\tau} \rceil$ and $\|\mathbf{E}\|_2^2 \le \|\mathbf{E}\|_\infty^2 \le n^2B^2$, the distribution of $e_{\mathrm{pmult}} = \hat{\mu}e + \tilde{e}$ follows

$$\mathbf{E}\mathcal{D}_{\mu+t\mathbb{Z}^n,\sigma} + \lfloor \mathcal{D}_{\mathbb{R}^n,\tau} \rceil \overset{\hat{\varepsilon}}{\approx} \left\lfloor \mathcal{D}_{\mathbb{R}^n,\sqrt{\sigma^2\mathbf{E}\mathbf{E}^\top + \tau^2\mathbf{I}}} \right\rceil$$

from Lemma 5. Finally, every diagonal entry of $\mathbf{\Sigma} = \sigma^2\mathbf{E}\mathbf{E}^\top + \tau^2\mathbf{I}$ satisfies

$$\mathbf{\Sigma}_{i,i} = \sigma^2\|e\|_2^2 + \tau^2 \le n\sigma^2B^2 + \tau^2,$$

so the bound is obtained from Lemma 6. □

B.6 Proof of Lemma 9

Proof. Note that the correctness is obvious from the definition. To analyze the error distribution, first notice that $e_{\mathrm{mult}} := \mathsf{Err}_{\mathsf{sk}}(\mathsf{Mult}^{\mathsf{E}}_{\sigma_1,\tau_1,\sigma_2,\tau_2}(\mathsf{pk},\mathsf{rlk},\hat{\mathbf{c}},\hat{\mathbf{c}}')) = \mathsf{Err}_{\mathsf{sk}}(\mathsf{Mult}(\mathsf{rlk},\hat{\mathbf{c}},\hat{\mathbf{c}}'))+\mathsf{Err}_{\mathsf{sk}}(\mathsf{PMult}(u,\mathbf{c}'))+\mathsf{Err}_{\mathsf{sk}}(\mathsf{PMult}(u',\mathbf{c}))$. Then, by Lemma 8, the distribution of e_{mult} is computationally indistinguishable from the distribution of $\mathsf{Err}_{\mathsf{sk}}(\mathsf{Mult}(\mathsf{rlk},\hat{\mathbf{c}},\hat{\mathbf{c}}')) + e_1 + e_2$, where $e_1 \leftarrow \mathcal{D}^{\sigma_1,\tau_1}_{\mathrm{ptmul}}(e)$, $e_2 \leftarrow \mathcal{D}^{\sigma_2,\tau_2}_{\mathrm{ptmul}}(e')$. Note that we can apply Lemma 8 even if $\hat{\mathbf{c}},\hat{\mathbf{c}}'$ has dependency with u,u' respectively, as Lemma 8 holds true for arbitrary plaintexts over R_t.

Now, we analyze the distribution of $e_0 := \mathsf{Err}_{\mathsf{sk}}(\mathsf{Mult}(\mathsf{rlk},\hat{\mathbf{c}},\hat{\mathbf{c}}'))$. First, the distribution of $\hat{\mathbf{c}}$ follows

$$\begin{aligned}
&\{(\mathbf{c} + \mathbf{r} + (\Delta u, 0) \mid \mathbf{r} \leftarrow \mathsf{Rand}(\mathsf{pk}), u \xleftarrow{\$} R_t\} \\
&\quad\approx_c \{(-\hat{a}s + \Delta m + e + e_{\mathrm{rand}}, \hat{a}) + (\Delta u, 0) \mid \hat{a} \xleftarrow{\$} R_q, e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}(s,\mathsf{pk}), u \xleftarrow{\$} R_t\} \\
&\quad\equiv \{(\hat{a}s + \Delta u + e + e_{\mathrm{rand}}) \mid \hat{a} \xleftarrow{\$} R_q, e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}(s,\mathsf{pk}), u \xleftarrow{\$} R_t\} \\
&\quad\equiv \mathcal{D}_{\mathrm{multr}}(s,\mathsf{pk},e)
\end{aligned}$$

by Lemma 4 and the fact that $t|q$. Similarly, the distribution of $\hat{\mathbf{c}}'$ is computationally indistinguishable from $\mathcal{D}_{\mathrm{multr}}(s,\mathsf{pk},e')$. Therefore, the distribution of e_0 is computationally indistinguishable from

$$\{\mathsf{Err}_{\mathsf{sk}}(\mathsf{Mult}(\mathsf{rlk},\hat{\mathbf{c}},\hat{\mathbf{c}}')) \mid \hat{\mathbf{c}} \leftarrow \mathcal{D}_{\mathrm{multr}}(s,\mathsf{pk},e), \hat{\mathbf{c}}' \leftarrow \mathcal{D}_{\mathrm{multr}}(s,\mathsf{pk},e')\}$$

as desired. Finally, the error bound is obtained from the definition and Lemma 8. □

B.7 Proof of Corollary 1

Lemma 13 (Tensor Product). *Let $\mathbf{c}, \mathbf{c}' \in R_q^2$ be BFV encryptions of $m, m' \in R_t$ respectively, with $e = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c})$, $e' = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}')$ such that $\|e\|_\infty \le B$, $\|e'\|_\infty \le B'$. If $\sigma, \tau > 0$ satisfy $\sigma^{-2} + (n^4+n^2)(B^2+B'^2)\tau^{-2} \le (q\eta_\varepsilon(\mathbb{Z}^{4n}))^{-2}$ for some negligible $\varepsilon > 0$, then $(\hat{d}_0, \hat{d}_1, \hat{d}_2) \leftarrow \mathsf{Tensor}^{\mathsf{E}}_{\sigma,\tau}(\mathbf{c}, \mathbf{c}')$ satisfies $\hat{d}_0 + \hat{d}_1 s + \hat{d}_2 s^2 = \Delta^2 mm' + e_{\mathrm{tensor}} \pmod{\Delta q}$ for some $e_{\mathrm{tensor}} \in R$, and the distribution of e_{tensor} is within max-log distance $\hat{\varepsilon} = \log \frac{1+\varepsilon}{1-\varepsilon}$ from*

$$\mathcal{D}^{\sigma,\tau}_{\mathrm{tensor}}(s,e,e') := ee' + \Delta\left\lfloor \mathcal{D}_{\mathbb{R}^n, -\frac{2}{\Delta}ee', \sqrt{\mathbf{\Sigma}}} \right\rceil$$

where $\mathbf{\Sigma} = \frac{1}{\Delta^2}(\sigma^2 \hat{\mathbf{E}}\hat{\mathbf{E}}^\top + \tau^2 \mathbf{I})$ and $\hat{\mathbf{E}} = \left[\mathbf{E}\ \mathbf{ES}\ \mathbf{E}'\ \mathbf{E}'\mathbf{S}\right]$. Moreover, $\|e_{\mathrm{tensor}}\|_\infty$ is bounded by $B^{\sigma,\tau}_{\mathrm{tensor}}(B,B') := K\sqrt{\sigma^2(n^3+n)(B^2+B'^2) + \tau^2} + nBB'$ with overwhelming probability.

Proof. By simple computation, we have

$$\begin{aligned}\hat{d}_0 + \hat{d}_1 s + \hat{d}_2 s^2 &= \Delta^2 mm' + e(\hat{c}'_0 + \hat{c}'_1 s) + e'(\hat{c}_0 + \hat{c}_1 s) + \tilde{e} - ee' \\ &= \Delta^2 mm' + (e\hat{c}'_0 + es\hat{c}'_1 + e'\hat{c}_0 + e's\hat{c}_1 + \tilde{e} - ee') \pmod{\Delta q}.\end{aligned}$$

Since $\|\hat{\mathbf{E}}\|_2^2 \le \|\mathbf{E}\|_2^2 + \|\mathbf{ES}\|_2^2 + \|\mathbf{E}'\|_2^2 + \|\mathbf{E}'\mathbf{S}\|_2^2 \le (n^4+n^2)(B^4+B^2)$, the distribution of $e_{\mathrm{tensor}} = e\hat{c}'_0 + es\hat{c}'_1 + e'\hat{c}_0 + e's\hat{c}_1 + \tilde{e} - ee'$ follows

$$\begin{aligned}&\mathbf{E}\mathcal{D}_{c'_0+q\mathbb{Z}^n,\sigma} + \mathbf{ES}\mathcal{D}_{c'_1+q\mathbb{Z}^n,\sigma} + \mathbf{E}'\mathcal{D}_{c_0+q\mathbb{Z}^n,\sigma} + \mathbf{E}'\mathbf{S}\mathcal{D}_{c_1+q\mathbb{Z}^n,\sigma} + \lfloor \mathcal{D}_{\mathbb{R}^n,\tau} \rceil_\Delta - ee' \\ &\quad\equiv \hat{\mathbf{E}}\mathcal{D}_{\hat{\mathbf{c}}+q\mathbb{Z}^{4n},\sigma} + \Delta\left\lfloor \mathcal{D}_{\mathbb{R}^n,\frac{1}{\Delta}\tau} \right\rceil - ee' \\ &\quad\equiv [\hat{\mathbf{E}}\hat{\mathbf{c}}]_\Delta + \Delta\left(\hat{\mathbf{E}}\mathcal{D}_{\frac{1}{\Delta}(\hat{\mathbf{c}}-\hat{\mathbf{E}}^{-1}[\hat{\mathbf{E}}\hat{\mathbf{c}}]_\Delta)+t\mathbb{Z}^n, -\frac{1}{\Delta}\hat{\mathbf{E}}^{-1}[\hat{\mathbf{E}}\hat{\mathbf{c}}]_\Delta, \frac{1}{\Delta}\sigma} + \left\lfloor \mathcal{D}_{\mathbb{R}^n,\frac{1}{\Delta}\tau} \right\rceil\right) - ee' \\ &\quad\overset{\hat{\varepsilon}}{\approx} 2ee' + \Delta\left\lfloor \mathcal{D}_{\mathbb{R}^n, -\frac{2}{\Delta}ee', \sqrt{\mathbf{\Sigma}}} \right\rceil - ee' \\ &\quad\equiv ee' + \Delta\left\lfloor \mathcal{D}_{\mathbb{R}^n, -\frac{2}{\Delta}ee', \sqrt{\mathbf{\Sigma}}} \right\rceil\end{aligned}$$

from Lemma 5, where $\hat{\mathbf{c}} = (c'_0, c'_1, c_0, c_1)$ and $\hat{\mathbf{E}}^{-1}$ is the right inverse of $\hat{\mathbf{E}}$. We note that our use of Lemma 5 is valid, since $\hat{\mathbf{E}} \cdot \frac{1}{\Delta}(\hat{\mathbf{c}} - \hat{\mathbf{E}}^{-1}[\hat{\mathbf{E}}\hat{\mathbf{c}}]_\Delta) = \frac{1}{\Delta}(\hat{\mathbf{E}}\hat{\mathbf{c}} - [\hat{\mathbf{E}}\hat{\mathbf{c}}]_\Delta) \in \mathbb{Z}^n$. Finally, every diagonal entry of $\mathbf{\Sigma}$ satisfies

$$\mathbf{\Sigma}_{i,i} = \frac{1}{\Delta^2}(\sigma^2(\|e\|_2^2 + \|es\|_2^2 + \|e'\|_2^2 + \|e's\|_2^2) + \tau^2) \le \frac{1}{\Delta^2}(\sigma^2(n^3+n)(B^2+B'^2) + \tau^2),$$

so the bound is obtained from Lemma 6. □

Lemma 14 (Modulus Switching). *Suppose that $(\hat{d}_0, \hat{d}_1, \hat{d}_2) \in R^3_{\Delta q}$ satisfies $\hat{d}_0 + \hat{d}_1 s + \hat{d}_2 s^2 = \Delta^2 m + e \pmod{\Delta q}$ for some $m \in R_t$ and $e \in R$ with $\|e\|_\infty \le B$. If $\sigma, \tau > 0$ satisfy $\sigma^{-2} + (n^4+n^2+1)\tau^{-2} \le \eta_\varepsilon(\mathbb{Z}^{3n})^{-2}$ for some negligible $\varepsilon > 0$, then $(d_0, d_1, d_2) \leftarrow \mathsf{ModSwitch}^{\mathsf{E}}_{\sigma,\tau}((\hat{d}_0, \hat{d}_1, \hat{d}_2))$ satisfies $d_0 + d_1 s + d_2 s^2 =$*

$\Delta m + e_{\mathrm{switch}} \pmod q$ *for some* $e_{\mathrm{switch}} \in R$ *whose distribution is within max-log distance* $\hat{\varepsilon} = \log \frac{1+\varepsilon}{1-\varepsilon}$ *from*

$$\mathcal{D}^{\sigma,\tau}_{\mathrm{scale}}(s,e) := \left\lfloor \frac{1}{\Delta} e + \mathcal{D}_{\mathbb{R}^n, \sqrt{\sigma^2(\mathbf{S}^2\mathbf{S}^{2\top} + \mathbf{S}\mathbf{S}^\top + \mathbf{I}) + \tau^2\mathbf{I}}} \right\rceil.$$

Moreover, $\|e_{\mathrm{switch}}\|_\infty$ *is bounded by* $B^{\sigma,\tau}_{\mathrm{switch}}(B) := K\sqrt{\sigma^2(n^3+n+1)+\tau^2} + \frac{1}{\Delta}B$ *with overwhelming probability.*

Proof. We can write $d'_i = \frac{1}{\Delta}\hat{d}_i + k_i$ where $k_i \sim \mathcal{D}_{-\frac{1}{\Delta}\hat{d}_i + \mathbb{Z}^n, \sigma}$ for $i = 0, 1, 2$. Then, we have

$$\begin{aligned} d_0 + d_1 s + d_2 s^2 &= \tilde{e} + d'_0 + d'_1 s + d'_2 s^2 \\ &= \frac{1}{\Delta}(\hat{d}_0 + \hat{d}_1 s + \hat{d}_2 s^2) + (k_0 + k_1 s + k_2 s^2) + \tilde{e} \\ &= \Delta m + \frac{1}{\Delta} e + (k_0 + k_1 s + k_2 s^2) + \tilde{e} \pmod q. \end{aligned}$$

From Lemma 5, the distribution of $e_{\mathrm{scale}} = \Delta^{-1} e + (k_0 + k_1 s + k_2 s^2) + \tilde{e}$ follows

$$\begin{aligned} e_{\mathrm{scale}} &\sim \frac{1}{\Delta} e + \mathcal{D}_{-\frac{1}{\Delta}\hat{d}_0 + \mathbb{Z}^n, \tau} + \mathbf{S}\mathcal{D}_{-\frac{1}{\Delta}\hat{d}_1 + \mathbb{Z}^n, \sigma} + \mathbf{S}^2\mathcal{D}_{-\frac{1}{\Delta}\hat{d}_2 + \mathbb{Z}^n, \sigma} + \lfloor \mathcal{D}_{\mathbb{R}^n, \tau} \rceil \\ &\equiv \frac{1}{\Delta} e + \left[\mathbf{I}\ \mathbf{S}\ \mathbf{S}^2\right] \mathcal{D}_{-\frac{1}{\Delta}(\hat{d}_0, \hat{d}_1, \hat{d}_2) + \mathbb{Z}^{3n}, \sigma} + \lfloor \mathcal{D}_{\mathbb{R}^n, \tau} \rceil \\ &\equiv \left\lfloor \frac{1}{\Delta} e + \left[\mathbf{I}\ \mathbf{S}\ \mathbf{S}^2\right] \mathcal{D}_{-\frac{1}{\Delta}(\hat{d}_0, \hat{d}_1, \hat{d}_2) + \mathbb{Z}^{3n}, \sigma} + \mathcal{D}_{\mathbb{R}^n, \tau} \right\rceil \\ &\overset{\hat{\varepsilon}}{\approx} \left\lfloor \frac{1}{\Delta} e + \mathcal{D}_{\mathbb{R}^n, \sqrt{\sigma^2(\mathbf{S}^2\mathbf{S}^{2\top} + \mathbf{S}\mathbf{S}^\top + \mathbf{I}) + \tau^2\mathbf{I}}} \right\rceil \end{aligned}$$

since $\|\left[\mathbf{I}\ \mathbf{S}\ \mathbf{S}^2\right]\|_2^2 \le \|\mathbf{I}\|_2^2 \|\mathbf{S}\|_2^2 + \|\mathbf{S}^2\|_2^2 \le 1 + n^2 + n^4$ and e_{switch} are integral. Finally, every diagonal entry of $\mathbf{\Sigma} = \sigma^2(\mathbf{S}^2\mathbf{S}^{2\top} + \mathbf{S}\mathbf{S}^\top + \mathbf{I}) + \tau^2\mathbf{I}$ satisfies

$$\mathbf{\Sigma}_{i,i} = \sigma^2(\|s^2\|_2^2 + \|s\|_2^2 + 1) + \tau^2 \le \sigma^2(n^3 + n + 1) + \tau^2,$$

so the bound directly follows from Lemma 6. □

Lemma 15 (Relinearization). *Suppose that* $(d_0, d_1, d_2) \in R_q^3$ *satisfies* $d_0 + d_1 s + d_2 s^2 = \Delta m + e \pmod q$ *for some* $m \in R_t$ *and* $e \in R$ *with* $\|e\|_\infty \le B$. *If* $\sigma_3, \tau_3 > 0$ *satisfy* $\sigma_3^{-2} + dn^2K^2\sigma_{\mathrm{err}}^2\tau_3^{-2} \le \eta_\varepsilon(\Lambda_q^\perp(\mathbf{g} \otimes \mathbf{I}))^{-2}$ *for some negligible* $\varepsilon > 0$, *then* $\mathbf{c}_{\mathrm{mult}} \leftarrow \mathsf{Relin}^{\mathsf{E}}(\mathsf{rlk}, (d_0, d_1, d_2))$ *is a BFV encryption of* m *where the distribution of error* $e_{\mathrm{relin}} := \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{mult}})$ *is computationally indistinguishable from*

$$\mathcal{D}^{\sigma_3,\tau_3}_{\mathrm{relin}}(s, \mathsf{rlk}, e) := e + \left\lfloor \mathcal{D}_{\mathbb{R}^n, \sqrt{\sigma_3^2 \mathbf{E}_{\mathsf{rlk}} \mathbf{E}_{\mathsf{rlk}}^\top + \tau_3^2 \mathbf{I}}} \right\rceil$$

where $\mathbf{E}_{\mathsf{rlk}} := \left[\mathbf{E}_{\mathsf{rlk},1} \cdots \mathbf{E}_{\mathsf{rlk},d}\right]$. *Moreover,* $\|e_{\mathrm{relin}}\|_\infty$ *bounded by* $B_{\mathrm{relin}}(B) := K\sqrt{dn\sigma_3^2K^2\sigma_{\mathrm{err}}^2 + \tau_3^2}$ *with an overwhelming probability.*

Proof. Since $h_{\mathrm{rand}}(d_2) \in h(d_2) + \Lambda_q^{\perp}(\mathbf{g} \otimes \mathbf{I})$, $\langle h_{\mathrm{rand}}(d_2), \mathbf{g}\rangle = \langle h(d_2), \mathbf{g}\rangle = d_2 \pmod q$. Therefore, from $(c_{\mathrm{prod},0}, c_{\mathrm{prod},1} \leftarrow \langle h_{\mathrm{rand}}(d_2), \mathsf{rlk}\rangle$, we have $c_{\mathrm{prod},0} + c_{\mathrm{prod},1}s = d_2 s^2 + \langle h_{\mathrm{rand}}(d_2), \mathbf{e}_{\mathsf{rlk}}\rangle \pmod q$. Hence, $\mathbf{c}_{\mathrm{mult}} := (c_{\mathrm{mult},0}, c_{\mathrm{mult},1})$ satisfies

$$\begin{aligned} c_{\mathrm{mult},0} + c_{\mathrm{mult},1}s &= d_0 + d_1 s + \overline{e} + c_{\mathrm{prod},0} + c_{\mathrm{prod},1}s \\ &= d_0 + d_1 s + d_2 s^2 + \langle h_{\mathrm{rand}}(d_2), \mathbf{e}_{\mathsf{rlk}}\rangle + \overline{e} \\ &= \Delta m + e + \langle h_{\mathrm{rand}}(d_2), \mathbf{e}_{\mathsf{rlk}}\rangle + \overline{e} \pmod q. \end{aligned}$$

Moreover, the distribution of $\langle h_{\mathrm{rand}}(d_2), \mathbf{e}_{\mathsf{rlk}}\rangle + \overline{e}$ follows

$$\begin{aligned} \sum_{i=1}^{d} \mathbf{E}_{\mathsf{rlk},i} \mathcal{D}_{h(d_2)_i + \Lambda_q^{\perp}(\mathbf{g}), \sigma_3} + \lfloor \mathcal{D}_{\mathbb{R}^n, \tau_3} \rceil &\equiv \mathbf{E}_{\mathsf{rlk}} \mathcal{D}_{h(d_2) + \Lambda_q^{\perp}(\mathbf{g}\otimes\mathbf{I}), \sigma_3} + \lfloor \mathcal{D}_{\mathbb{R}^n, \tau_3} \rceil \\ &\overset{\hat{\varepsilon}}{\approx} \left\lfloor \mathcal{D}_{\mathbb{R}^n, \sqrt{\sigma_3^2 \mathbf{E}_{\mathsf{rlk}} \mathbf{E}_{\mathsf{rlk}}^{\top} + \tau_3^2 \mathbf{I}}} \right\rceil \end{aligned}$$

by Lemma 5, as $\|\mathbf{E}_{\mathsf{rlk}}\|_2^2 \le \sum_{i=1}^{d} \|\mathbf{E}_{\mathsf{rlk},i}\|_2^2 \le dn^2 K^2 \sigma_{\mathrm{err}}^2$. Therefore, e_{mult} follows $e + \left\lfloor \mathcal{D}_{\mathbb{R}^n, \sqrt{\sigma_3^2 \mathbf{E}_{\mathsf{rlk}} \mathbf{E}_{\mathsf{rlk}}^{\top} + \tau_3^2 \mathbf{I}}} \right\rceil$, as desired.

Finally, every diagonal entry of $\mathbf{\Sigma} = \sigma_3^2 \mathbf{E}_{\mathsf{rlk}} \mathbf{E}_{\mathsf{rlk}}^{\top} + \tau_3^2 \mathbf{I}$ satisfies

$$\mathbf{\Sigma}_{i,i} = \sum_{j=1}^{d} \sigma_3^2 \|\mathbf{e}_{\mathsf{rlk},j}\|_2^2 + \tau_3^2 \le dn\sigma_3^2 K^2 \sigma_{\mathrm{err}}^2 + \tau_3^2,$$

the bound follows from Lemma 6. □

B.8 Proof of Lemma 10

Proof. Let $e_{\mathrm{rand}} = r_0 + r_1 s \pmod q$. Then,

$$d_0 + d_1 \psi_j(s) = \psi_j(c_0 + c_1 s) + \psi_j(r_0 + r_1 s) = \Delta \psi_j(m) + (\psi_j(e + e_{\mathrm{rand}})) \pmod q.$$

From $(c_{\mathrm{prod},0}, c_{\mathrm{prod},1}) \leftarrow d_1 \boxdot \mathsf{atk}_j$, we have $c_{\mathrm{prod},0} + c_{\mathrm{prod},1}s = d_1\psi_j(s) + \langle h(d_1), \mathbf{e}_{\mathsf{atk}_j}\rangle \pmod q$. Hence, $\mathbf{c}_{\mathrm{aut}} = (c_{\mathrm{aut},0}, c_{\mathrm{aut},1})$ satisfies

$$\begin{aligned} c_{\mathrm{aut},0} + c_{\mathrm{aut},1}s &= d_0 + c_{\mathrm{prod},0} + c_{\mathrm{prod},1}s = d_0 + d_1\psi_j(s) + \langle h(d_1), \mathbf{e}_{\mathsf{atk}_j}\rangle \\ &= \Delta m + (\psi_j(e + e_{\mathrm{rand}}) + \langle h(d_1), \mathbf{e}_{\mathsf{atk}_j}\rangle) \pmod q. \end{aligned}$$

To analyze the distribution of $e_{\mathrm{aut}} = \psi_j(e + e_{\mathrm{rand}}) + \langle h(d_1), \mathbf{e}_{\mathsf{Aut}_j}\rangle$, we first claim that d_1 is computationally indistinguishable from a uniformly random variable over R_q. From Lemma 4, we have

$$\begin{aligned} &\{(\mathsf{sk}, \mathsf{evk}, (d_0, d_1))\} \equiv \{(\mathsf{sk}, \mathsf{evk}, (\psi_j(c_0 + r_0, \psi_j(c_1 + r_1)) \mid (r_0, r_1) \leftarrow \mathsf{Rand}(\mathsf{pk})\} \\ &\approx_c \{(\mathsf{sk}, \mathsf{evk}, (-u\psi_j(s) + \Delta\psi_j(m) + \psi_j(e + e_{\mathrm{rand}}), u)) \mid u \xleftarrow{\$} R_q, e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}\}. \end{aligned}$$

Therefore, the distribution of e_{aut} follows

$$\left\{ (\psi_j(e + e_{\mathrm{rand}}) + \langle h(d_1), \mathbf{e}_{\mathsf{atk}_j} \rangle) \;\middle|\; \begin{array}{c} (r_0, r_1) \leftarrow \mathsf{Rand}(\mathsf{pk}) \\ e_{\mathrm{rand}} = \mathsf{Err}_{\mathsf{sk}}((r_0, r_1) \\ d_1 = \psi_j(c_1 + r_1) \end{array} \right\}$$
$$\approx_c \{(\psi_j(e + e_{\mathrm{rand}}) + \langle h(u), \mathbf{e}_{\mathsf{atk}_j} \rangle) \mid u \xleftarrow{\$} R_q, e \leftarrow \mathcal{D}_{\mathrm{rand}}\}.$$

Finally, the bound is directly obtained from the definition of $\mathcal{D}_{\mathrm{aut}}$. □

B.9 Proof of Theorem 5

Proof. Let $\mathsf{Sim}^{\mathsf{E}}$ be an error simulator for $\mathsf{Eval}^{\mathsf{E}}$ and a circuit $C : \mathcal{M}^\ell \times \mathcal{M}^k \to \mathcal{M}$. We construct a strongly ciphertext-simulatable evaluation algorithm $\mathsf{Eval}^{\mathsf{SC}}$ for C, and the corresponding strong ciphertext simulator $\mathsf{Sim}^{\mathsf{SC}}$ from $\mathsf{Eval}^{\mathsf{E}}$ and $\mathsf{Sim}^{\mathsf{E}}$ as follows:

$\mathsf{Eval}^{\mathsf{SC}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y})$
$\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{PMult}^{\mathsf{E}}(\mathsf{Eval}^{\mathsf{E}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y}), 1)$
$\mathbf{c}_{\mathrm{rand}} \leftarrow \mathsf{Rand}(\mathsf{pk})$
$\mathbf{c}'_{\mathrm{out}} \leftarrow \mathbf{c}_{\mathrm{out}} + \mathbf{c}_{\mathrm{rand}} \pmod q$
return $\mathbf{c}'_{\mathrm{out}}$

$\mathsf{Sim}^{\mathsf{SC}}(\mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)$
$\mathbf{c}_{\mathrm{out}} \leftarrow \mathsf{PMult}^{\mathsf{E}}(\mathsf{Eval}^{\mathsf{E}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{0}), 0)$
$\mathbf{c}_{\mathrm{rand}} \leftarrow \mathsf{Rand}(\mathsf{pk})$
$\mathbf{c}'_{\mathrm{out}} \leftarrow \mathbf{c}_{\mathrm{out}} + (\Delta z, 0) + \mathbf{c}_{\mathrm{rand}} \pmod q$
return $\mathbf{c}'_{\mathrm{out}}$

We claim that for any $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathcal{M}^\ell$ and $\mathbf{y} = (y_1, \ldots, y_k) \in \mathcal{M}^k$, $\mathsf{Eval}^{\mathsf{SC}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y})$ and $\mathsf{Sim}^{\mathsf{SC}}(\mathsf{evk}, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), z)$ produce computationally indistinguishable distributions where $\mathbf{c}_i \leftarrow \mathsf{Enc}_{\mathsf{sk}}(x_i)$ for $1 \le i \le \ell$ and $z = C(\mathbf{x}, \mathbf{y})$. We proceed by the hybrid argument, where $\mathbf{Hyb}_0$ is the distribution of $\mathsf{Eval}^{\mathsf{SC}}$.

$\mathbf{Hyb}_1$. In this hybrid, we sample $\mathbf{c}_{\mathrm{rand}} = (c_{\mathrm{rand},0}, c_{\mathrm{rand},1})$ as $c_{\mathrm{rand},1} \xleftarrow{\$} R_q, c_{\mathrm{rand},0} \leftarrow -c_{\mathrm{rand},1}s + e_{\mathrm{rand}}$, where $e_{\mathrm{rand}} \leftarrow \mathcal{D}^{\sigma,\tau}_{\mathrm{rand}}$. $\mathbf{c}_{\mathrm{out}}, \mathbf{c}'_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_0$. By Lemma 4, $\mathbf{Hyb}_0$ and $\mathbf{Hyb}_1$ are computationally indistinguishable.

$\mathbf{Hyb}_2$. In this hybrid, we sample $\mathbf{c}'_{\mathrm{out}} = (c'_{\mathrm{out},0}, c'_{\mathrm{out},1})$ as $c'_{\mathrm{out},1} \xleftarrow{\$} R_q, c'_{\mathrm{out},0} \leftarrow -c'_{\mathrm{out},1}s + \Delta z + e_{\mathrm{rand}} + e_{\mathrm{out}}$ where $e_{\mathrm{rand}} \leftarrow \mathcal{D}_{\mathrm{rand}}, e_{\mathrm{out}} \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\mathrm{out}})$. $\mathbf{c}_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_1$, and $\mathbf{c}_{\mathrm{rand}}$ is removed. Notice that $\mathsf{PMult}(C(\mathbf{x}, \mathbf{y}), 1) = C(\mathbf{x}, \mathbf{y}) = z$, so we can write $\mathbf{c}_{\mathrm{out}} = (-c_{\mathrm{out},1}s + \Delta z + e_{\mathrm{out}}, c_{\mathrm{out},1})$. Hence, in $\mathbf{Hyb}_1$,

$$\begin{aligned} \mathbf{c}'_{\mathrm{out}} &= \mathbf{c}_{\mathrm{out}} + \mathbf{c}_{\mathrm{rand}} \\ &= (-(c_{\mathrm{out},1} + c_{\mathrm{rand},1})s + \Delta z + e_{\mathrm{rand}} + \hat{e}_{\mathrm{out}}, c_{\mathrm{out},1} + c_{\mathrm{rand},1}) \pmod q \end{aligned}$$

where $c_{\mathrm{rand},1}$ follows a uniform distribution over R_q. Therefore, $c_{\mathrm{out},1} + c_{\mathrm{rand},1}$ also follows a uniform distribution over R_q, which is exactly the distribution of $c'_{\mathrm{out},1}$ in $\mathbf{Hyb}_2$. Therefore, $\mathbf{Hyb}_1$ and $\mathbf{Hyb}_2$ are identical.

$\mathbf{Hyb}_3$. In this hybrid, we sample $e_{\mathrm{out}} \leftarrow \mathcal{D}^{\sigma,\tau}_{\mathrm{pmult}}(\hat{e}_{\mathrm{out}})$, where $\hat{\mathbf{c}}_{\mathrm{out}} \leftarrow \mathsf{Eval}^{\mathsf{E}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{y}), \hat{e}_{\mathrm{out}} \leftarrow \mathsf{Err}_{\mathsf{sk}}(\hat{\mathbf{c}}_{\mathrm{out}})$. $e_{\mathrm{rand}}, \mathbf{c}'_{\mathrm{out}}$ is sampled identically as $\mathbf{Hyb}_2$, and

$\mathbf{c}_{\text{out}}$ is removed. By Lemma 8, $\mathbf{Hyb}_2$ and $\mathbf{Hyb}_3$ are computationally indistinguishable.

$\mathbf{Hyb}_4$. In this hybrid, we sample $\hat{e}_{\text{out}} \leftarrow \mathsf{Sim}^{\mathsf{E}}(\mathsf{sk}, \mathsf{evk}, (e_1, \ldots, e_\ell))$ where $e_i = \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_i)$ for $1 \leq i \leq \ell$ and $\mathsf{Sim}^{\mathsf{E}}$ is the error simulator for $\mathsf{Eval}^{\mathsf{E}}$ and C. $e_{\text{out}}, e_{\text{rand}}, \mathbf{c}'_{\text{out}}$ is sampled identically as $\mathbf{Hyb}_2$, and $\hat{\mathbf{c}}_{\text{out}}$ is removed. By the error simulatability of $\mathsf{Eval}^{\mathsf{E}}$, $\mathbf{Hyb}_4$ and $\mathbf{Hyb}_4$ are computationally indistinguishable.

$\mathbf{Hyb}_5$. In this hybrid, we sample $\hat{\mathbf{c}}_{\text{out}} \leftarrow \mathsf{Eval}^{\mathsf{E}}(\mathsf{evk}, C, (\mathbf{c}_1, \ldots, \mathbf{c}_\ell), \mathbf{0})$ and $e_{\text{out}} \leftarrow \mathsf{Err}_{\mathsf{sk}}(\hat{\mathbf{c}}_{\text{out}})$. $e_{\text{out}}, e_{\text{rand}}, \mathbf{c}'_{\text{out}}$ is sampled identically as $\mathbf{Hyb}_4$. This hybrid is simply a reverse of $\mathbf{Hyb}_4$, so by the error simulatability of $\mathsf{Eval}^{\mathsf{E}}$, $\mathbf{Hyb}_4$ and $\mathbf{Hyb}_5$ are computationally indistinguishable.

$\mathbf{Hyb}_6$. In this hybrid, we sample $\mathbf{c}_{\text{out}} \leftarrow \mathsf{PMult}^{\mathsf{E}}_{\sigma,\tau}(\hat{\mathbf{c}}_{\text{out}})$ and $e_{\text{out}} \leftarrow \mathsf{Err}_{\mathsf{sk}}(\mathbf{c}_{\text{out}})$. $e_{\text{rand}}, \mathbf{c}'_{\text{out}}$ is sampled identically as $\mathbf{Hyb}_5$, and $\hat{e}_{\text{out}}$ is removed. This hybrid is simply a reverse of $\mathbf{Hyb}_3$, so by Lemma 8, $\mathbf{Hyb}_5$ and $\mathbf{Hyb}_6$ are computationally indistinguishable.

$\mathbf{Hyb}_7$. In this hybrid, we sample $\mathbf{c}_{\text{rand}} = (c_{\text{rand},0}, c_{\text{rand},1})$ as $c_{\text{rand},1} \xleftarrow{\$} R_q, c_{\text{rand},0} \leftarrow -c_{\text{rand},1}s + e_{\text{rand}}$, and sample $\mathbf{c}'_{\text{out}} \leftarrow \mathbf{c}_{\text{out}} + (\Delta z, 0) + \mathbf{c}_{\text{rand}}$. $\mathbf{c}_{\text{out}}$ is sampled identically as $\mathbf{Hyb}_4$, and e_{out} is removed. Notice that $\mathsf{PMult}(C(\mathbf{x}, \mathbf{y}), 0) = 0 \cdot C(\mathbf{x}, \mathbf{y}) = 0$, so we can write $\mathbf{c}_{\text{out}} + (\Delta z, 0) = (-c_{\text{out},1}s + \Delta z + e_{\text{out}}, c_{\text{out},1})$. Therefore, this hybrid is simply a reverse of $\mathbf{Hyb}_2$, so $\mathbf{Hyb}_4$ and $\mathbf{Hyb}_5$ are identical.

$\mathbf{Hyb}_8$. In this hybrid, we sample $\mathbf{c}_{\text{rand}} \leftarrow \mathsf{Rand}_{\sigma,\tau}(\mathsf{pk})$. $\mathbf{c}_{\text{out}}, \mathbf{c}_{\text{rand}}$ is sampled identically as $\mathbf{Hyb}_5$, and e_{rand} is removed. This hybrid is simply a reverse of $\mathbf{Hyb}_1$, hence $\mathbf{Hyb}_7$ and $\mathbf{Hyb}_8$ are identical.

Finally, since $\mathbf{Hyb}_8$ is identical to the distribution of $\mathsf{Sim}^{\mathsf{SC}}$, we conclude that $\mathsf{Eval}^{\mathsf{SC}}$ is strongly ciphertext-simulatable for C. □