Low Communication Threshold FHE from Standard (Module-)LWE

Hiroki Okada[1,2] and Tsuyoshi Takagi[2]

[1] KDDI Research, Inc., Saitama, Japan
[2] The University of Tokyo, Tokyo, Japan

Abstract. Threshold fully homomorphic encryption (ThFHE) is a multi-party extension of FHE; any subset of at least T out of N parties can decrypt the ciphertexts by combining their *decryption shares*. Recently, Passelègue and Stehlé (Asiacrypt 2024) presented a ThFHE scheme with polynomially short decryption shares from the "known-norm" variant of learning with errors (LWE) assumption, in which the norm of the secret key is leaked to the adversary. While known-norm LWE is reduced from standard LWE, its module extension, known-covariance module-LWE (MLWE), lacks a known reduction from standard MLWE. Hence, extending their ThFHE scheme to the MLWE-based construction remains an open question.

In this paper, we address this open problem: We construct a ThFHE scheme with polynomially small decryption shares from standard LWE/MLWE. Our core technique, which we call *noise padding*, eliminates the need of known-norm variants of LWE. We distribute shares of a *padding noise* and use them to adjust the distribution of decryption noise so that no information about the secret key is leaked. Furthermore, our ThFHE efficiently realizes arbitrary T-out-of-N threshold decryption via simple Shamir secret sharing instead of $\{0,1\}$-linear secret sharing. Hence, the sizes of the keys, ciphertexts and decryption shares in our scheme are compact: they are $O(1)$ w.r.t. the number of parties N.

1 Introduction

Fully homomorphic encryption (FHE) was first realized by Gentry [Gen09]. Constructions based on the learning with errors (LWE) problem [Reg09] and its variants ring-LWE (RLWE) [LPR10] or module-LWE (MLWE) [BGV12] are efficient, leading to active research in this field [CKKS17; CGGI20]. Threshold FHE (ThFHE) is an extension of FHE in which ciphertexts can be decrypted by collecting *partial decryption shares* from T out of N parties, where N is the number of parties and T ($\leq N$) is a threshold. In 2023, the US agency NIST

This preprint is the full version of the paper published in the proceedings of Asiacrypt 2025 [OT26]. The paper is available at: https://doi.org/10.1007/978-981-95-5122-4_6. See Sect. 1.3 for a summary of the changes.

initiated a project to establish guidelines and recommendations for implementing threshold cryptosystems, including ThFHE [BP23]. ThFHE is an important cryptographic tool that can be applied to construct round optimal MPC protocols [GGHR14; GLS15; BJMS20] and *universal thresholdizer* [BGG+18], which can be used to add threshold functionality to, e.g., CCA-secure PKE, signature schemes, pseudorandom functions (PRF) and functional encryptions.

ThFHE was first constructed by Asharov et al. [AJL+12], and Boneh et al. [BGG+18] proposed a ThFHE with arbitrary T-out-of-N decryption. A notable drawback of these schemes is that they require a super-polynomially large modulus, leading to super-polynomially long keys, ciphertexts and decryption shares. To address this drawback, Boudgoust and Scholl [BS23] proposed a ThFHE scheme with polynomially small modulus, providing game-based security. However, Passelègue and Stehlé [PS25] recently noted a flaw in the security proof of [BS23]. In response to this attack, Boudgoust and Scholl updated their scheme in [BS24], and their scheme now achieves only *selective* security, where the adversary is allowed to query the encryption (and homomorphic evaluation) oracle only before receiving challenge ciphertexts and partial decryption shares.

In addition to the cryptanalysis, Passelègue and Stehlé [PS25] proposed a simulation-secure ThFHE scheme with polynomially short decryption shares, realizing threshold decryption with a small communication size. Although this scheme still requires a super-polynomially large modulus for the input ciphertexts, the communication size of sending them can be reduced by *transciphering* [NLV11; BCK+23] at the expense of homomorphic evaluation of the decryption circuit of symmetric key encryption (SKE) such as AES or FHE-friendly SKEs [ARS+15; DEG+18; HKL+22]. Hence, we can achieve threshold FHE with small input/output communications.

One notable drawback of the ThFHE scheme of [PS25] is that it cannot be extended to the construction from MLWE (or RLWE). Thus, it cannot be applied to most instantiations of FHE, including [CKKS17; CGGI20]. This is because the ThFHE of [PS25] relies on "known-norm" LWE (knLWE) [MS25] where the norm of the secret vector is given to the adversary. (Precisely, [PS25] uses "yet another" LWE (yaLWE), which is a generalization of known-norm LWE.) Since there exists a reduction from (standard) LWE to knLWE [MS25], ThFHE of [PS25] can be based on LWE. To extend their scheme into an MLWE-based construction, a "known-covariance" variant of MLWE is needed. However, no reduction from standard assumption such as MLWE to the known-covariance MLWE is known. Hence, Passelègue and Stehlé posed the following open question:

Question 1.1. Can we construct (simulation-secure) threshold FHE with polynomially short decryption shares from MLWE (or RLWE)?

Another drawback in the ThFHE scheme of [PS25] and prior works ([BS23] and [BGG+18, Constr. 5.6]) is that they are based on the $\{0,1\}$-linear secret sharing scheme ($\{0,1\}$-LSSS). The size of secret key shares and decryption shares $\{0,1\}$-LSSS are large for T-out-of-N threshold decryption: Each key

Table 1. Comparison of ThFHE schemes with (arbitrary) T-out-of-N decryption. Q and q represents the modulus of ciphertexts and decryption shares, respectively. For the rows of "Dec share size" and "sk share size", we write the number of elements (over $\mathbb{Z}_q$ and $\mathbb{Z}_q^n$) included in decryption and secret shares, respectively. L_{Dec} represents the bound on the decryption query; we write $L_{\mathsf{Dec}} = \infty$ if it is unbounded. The row "Sim. Sec." marks schemes with simulation security. The row "Module-LWE" marks which schemes can be constructed from MLWE. The row "Without server" marks which schemes can be constructed without a semi-honest and uncorrupted server.

	[BGG+18]	[BS23]	[PS25]	Ours §5	Ours §6, §7
Enc mod. (Q)	$\Omega(N^{4.2})\kappa^{\omega(1)}$	$\Omega(N^{4.2})\kappa^{O(1)}$	$\Omega(N^{4.2})\kappa^{\omega(1)\dagger}$	$\kappa^{\omega(1)\dagger}$	$\kappa^{\omega(1)\dagger}$
Dec mod. (q)			$\Omega(N^{4.2})\kappa^{O(1)}$	$\kappa^{O(1)}$	$\kappa^{O(1)}$
Dec share size	$O(N^{4.2})$	$O(N^{4.2})$	$O(N^{4.2})$	$O(1)$	$O(1)$
sk share size					$O(L_{\mathsf{Dec}})$
Query (L_{Dec})	∞	Bounded	∞	∞	Bounded
Sim. Sec.	Yes	–‡	Yes	Yes	Yes
Module-LWE	Yes	Yes	–	–	Yes(§7)
Without server	Yes	Yes	–	–	–

† The communication size of sending super-polynomially long FHE ciphertexts can be reduced to polynomially short size by transciphering [NLV11; BCK+23].
‡ Only achieves *selective IND-CPA* [BS24], because of the attack of [PS25].

share and decryption share consist of $O(N^{4.2})$ [Val84; Gol20] (or $\binom{T}{N}$ [BD10]) elements over $\mathbb{Z}_q^n$ and $\mathbb{Z}_q$, respectively (for LWE-based ThFHE). Hence, the size of the noise that appears at the decryption also becomes $O(N^{4.2})$; the modulus needs to be $\Omega(N^{4.2})$ to satisfy correctness. A simple and efficient method for achieving threshold decryption is Shamir secret sharing [Sha79], whose share size is $O(1)$ w.r.t. N. Although [BGG+18, Constr. 5.11] presents a ThFHE scheme with Shamir secret sharing, it requires superexponentially large modulus, e.g., $\omega((N!)^3)$ because the Lagrange coefficients are not small and blow up the noise. Hence, as noted in [PS25], the following open question remains.

Question 1.2. Can we construct threshold FHE with polynomially short decryption shares from Shamir secret sharing (instead of $\{0,1\}$-LSSS)?

1.1 Our Contributions

In this paper, we address both Question 1.1 and Question 1.2. We provide a brief summary of our contributions (and subsequent future works) in Table 1.

Shamir-secret-sharing-friendly. We first solve Question 1.2 in Sect. 5: we propose a knLWE-based threshold FHE scheme with Shamir secret sharing by modifying the ThFHE scheme of [PS25] via a simple masking technique. Owing to the efficiency of Shamir secret sharing, our scheme becomes compact

($= O(1)$ w.r.t. N); the sizes of moduli (Q and q), decryption shares, and secret shares do not increase depending on N. Note that, although [BGG+18, Constr. 8.35] also presents a compact ThFHE, it relies on a heavy cryptographic tool called an *universal thresholdizer*, which requires (non-compact) ThFHE and NIZK. In contrast, we construct a compact ThFHE scheme directly from LWE.

From standard LWE/MLWE. Next, we solve Question 1.1 in Sect. 6 and Sect. 7: we modify our knLWE-based ThFHE (presented in Sect. 5) to present ThFHE schemes from standard LWE/MLWE in Sect. 6/Sect. 7. Our core technique is what we call *noise padding*, which adds a small noise ζ whose standard deviation is $\sqrt{B_{\mathsf{pub}} - \|\mathbf{s}\|}$ and adjusts the standard deviation of the noise obtained after decryption to be a public value composed of B_{pub}; we thereby prevent the leakage of $\|\mathbf{s}\|$. With this technique, we construct ThFHE without knLWE and prove its security directly from standard LWE. Furthermore, we extend our LWE-based ThFHE scheme to a ThFHE scheme from MLWE.

Solely discrete. As a minor contribution, our ThFHE schemes are constructed solely with discrete values, while [PS25] requires (flooding) noise sampled from *continuous* Gaussian distribution. In general, when we implement continuous values, we need to approximate them with, e.g., floating-point numbers. However, it needs to be formally proven that the rounding errors incurred by the approximation do not affect the security and correctness. In contrast, we use *discrete* Gaussian (flooding) noise instead of continuous Gaussian noise and formally analyse correctness and security.

1.2 Future Work

Our scheme from standard (M)LWE (presented in Sect. 6 and Sect. 7) needs a bound on the number of decryption queries, as with [BS23]. We need to generate fresh padding noises ζ as many as the decryption query bound L_{Dec}, and distribute their shares to the parties at the setup stage. As a result, the storage cost for the secret shares linearly increases w.r.t. L_{Dec}. However, we note that the value of L_{Dec} is independent of any other security-related parameters, unlike in [BS23]. Thus, this drawback only increases the storage cost for the secret shares. Hence, this limitation would not be problematic for some use cases in which the number of decryption queries is small or the parties can tolerate a storage cost of $O(L_{\mathsf{Dec}})$. We leave it an open problem to improve our scheme by removing the bound on L_{Dec}. Note that our ThFHE presented in Sect. 5 does not have this drawback since it does not use padding noise but rather relies on knLWE.

In addition, as in the prior work [PS25], our ThFHE schemes requires a party called Server who is untrusted (semi-honest) but assumed not to be corrupted by the adversary. This is because the ThFHE scheme of [PS25] and our schemes sanitize the input ciphertext via the $\mathsf{ServerDec}$ procedure (explained later in Sect. 2.2), using the private randomness of Server that remains unknown to all parties and the adversary. Let us now describe a potential idea to eliminate

the need of Server in ThFHE of [PS25]; we perform $\mathsf{ServerDec}$ in a distributed manner. At the first round, T parties perform $\mathsf{ServerDec}$ on the input ciphertext ct of plaintext μ using their private randomness, and broadcast the output ciphertext $\overline{\mathsf{ct}}_i$ to the parties. At the second round, the parties calculate $\overline{\mathsf{ct}} := \sum_i \overline{\mathsf{ct}}_i$, whose plaintext becomes $T \cdot \mu$; we now assume $\mu \in \{0, 1\}$ and the plaintext space is $\mathbb{Z}_t$ for some $t > N$ (or, if we assume T is odd, we can use $t = 2$ and obtain $T \cdot \mu = \mu \bmod 2$). The parties then generate the partial decryption shares of $\overline{\mathsf{ct}}$ and complete the decryption as in ThFHE of [PS25] to recover $T \cdot \mu$, then decode it to μ with division by T. In this distributed $\mathsf{ServerDec}$, the ciphertext is successfully sanitized since $\overline{\mathsf{ct}}$ includes randomness of honest parties. However, this idea requires one more round (for distributed $\mathsf{ServerDec}$) than prior ThFHE schemes without Server such as [AJL+12; BGG+18; BS23], which makes the security proof challenging. We leave the formal concrete construction and the security proof of this potential idea to future work.

1.3 Changelog

This paper has been updated multiple times since its initial appearance on the IACR Cryptology ePrint Archive. We briefly summarize the changes below.

- Ver.1 (2025-03-04): Initial version.
- Ver.2 (2025-07-03): Minor revision.
- Ver.3 (2025-09-24): The full version of [OT26], with full proofs and omitted definitions included in the appendices.
- Ver.4 (2026-03-26) (This version): While Ver.3 required the modulus q to be even for correctness and security, we remove this restriction in Ver.4 by performing a more refined analysis of the ciphertext noise. Specifically, we improve lemmas and theorems for noise analysis (Lems. 5.1, 6.1 and 7.1), correctness (Thms. 5.3, 6.2 and 7.4), and security (Thms. 5.5, 6.3 and 7.5) by eliminating the restriction on q at the cost of only a very slight[3] increase in the Gaussian parameter σ.

2 Technical Overview

We first recall the ThFHE schemes of [AJL+12; BGG+18], which are constructed (solely) with a super-polynomially large modulus in Sect. 2.1. Then, in Sect. 2.2, we describe the prior work [PS25], which achieves a polynomially small modulus for the decryption share, and we explain why their scheme needs known-norm variant of LWE. Then, Sect. 2.3, we provide a technical overview of our first ThFHE scheme, which can be efficiently constructed with Shamir secret sharing. Finally, in Sect. 2.4, we provide a technical overview of our second ThFHE scheme, which can be (directly) based on standard LWE/MLWE.

[3] E.g., while $\overline{\sigma_{\mathsf{dec}}} := \sqrt{n(\sigma_0 B_{\mathsf{lwe}})^2 + (\sigma_{\mathsf{flood}}/p)^2 + \sigma_1^2}$ is sufficient for [OT26, Thm. 5.3], Thm. 5.3 in this version requires $\overline{\sigma_{\mathsf{dec}}} := \sqrt{n(\sigma_0 B_{\mathsf{lwe}})^2 + (\sigma_{\mathsf{flood}}/p)^2} + \sigma_1$.

2.1 Noise Flooding [AJL+12; BGG+18]

We first recall the threshold decryption procedure of the ThFHE schemes of [AJL+12; BGG+18]. For simplicity, we assume the underlying FHE is based on LWE. Let the public key be $\mathsf{pk} := (\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e})$, where $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_Q^{m \times n})$, $\mathbf{s} \in \mathbb{Z}_Q^n$ is a secret key, $\mathbf{e} \in \mathbb{Z}_Q^m$ is a (short) secret error, and Q is a modulus. We assume that ciphertexts, after some homomorphic evaluation, are in the following form:

$$\mathsf{ct}_{\mathsf{eval}} := (\mathbf{a} \sim \mathcal{U}(\mathbb{Z}_Q^n), b := \mathbf{a}^\intercal \mathbf{s} + e_{\mathsf{eval}} + \lfloor \tfrac{q}{2} \rfloor \cdot \mu \bmod Q), \tag{1}$$

where e_{eval} is a (small) noise and $\mu \in \{0, 1\}$ is the plaintext. For simplicity, we now only consider N-out-of-N threshold decryption with additive secret sharing: each party $P_1, \ldots, P_N$ holds a secret key share $\mathbf{s}_1, \ldots, \mathbf{s}_N$ such that $\sum_{i=1}^N \mathbf{s}_i = \mathbf{s}$. Given ct, each party computes a partial decryption share as follows;

$$\mathsf{pd}_i \leftarrow \mathsf{PartDec}(\mathsf{ct}, \mathbf{s}_i) := \mathbf{a}^\intercal \mathbf{s}_i + e_i, \tag{2}$$

and broadcasts pd_i to all parties. Given $\{\mathsf{pd}_i\}_{i \in [N]}$, each party computes

$$\begin{aligned} \overline{\mu} := \mathsf{FinDec}(\mathsf{ct}, \{\mathsf{pd}_i\}_{i \in [N]}) &:= \lfloor (b - \textstyle\sum_{i \in [N]} \mathsf{pd}_i) / \lfloor \tfrac{Q}{2} \rfloor \rceil \\ &= \mu + \lfloor (e_{\mathsf{eval}} - \textstyle\sum_{i \in [N]} e_i) / \lfloor \tfrac{Q}{2} \rfloor \rceil \end{aligned}$$

and then correctly recovers μ if the parameters are selected so that the noise term $\overline{e} := e_{\mathsf{eval}} - \sum_{i=1}^N e_i$ satisfies $|\overline{e}| < \lfloor \frac{Q}{4} \rfloor$ (with overwhelming probability).

Notably, each party can also recover the noise term by calculating

$$e_{\mathsf{leak}} := b - \textstyle\sum_{i \in [N]} \mathsf{pd}_i - \lfloor \frac{Q}{2} \rfloor \cdot \overline{\mu} = e_{\mathsf{eval}} - \sum_{i \in [N]} e_i. \tag{3}$$

The noise term e_{eval} of the ciphertext after homomorphic evaluations may strongly depend on the input plaintexts. For example, in the CKKS FHE scheme [CHK+18], the noise term of the multiplication of two ciphertexts encrypting μ_0 and μ_1 with noise terms e_0 and e_1 is as follows:

$$e_{\mathsf{eval}} = \mu_0 e_1 + \mu_1 e_0 + e'.$$

More simply, even without homomorphic evaluation, the ciphertext noise contains secret information; e.g., the ciphertext noise of Regev PKE [Reg09] has the form of $e_{\mathsf{eval}} = \mathbf{r}^\intercal \mathbf{e}$, where $\mathbf{r}$ is a short vector and $\mathbf{e}$ is the secret error vector of the LWE public key $\mathsf{pk} := (\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e})$.

This leak of the noise e_{eval} in the ciphertext is not problematic in single-key FHE (or PKE) since the decryption is performed only by the party who owns the secret key $\mathbf{s}$ (and $\mathbf{e}$) and is allowed to recover plaintexts. However, it is problematic in threshold FHE since the decryption is executed by parties who are only permitted to have the share of the secret key $\mathbf{s}_i$ and the input and result plaintexts (excluding the input plaintexts of other parties). Hence, we need to hide undesirable information in e_{eval} for security.

Asharov et al. [AJL+12, Lemma 2.1] presented a simple method to "smudge out" any information in e_{eval}, known as "noise flooding", by adding super-polynomially large noise to the partial decryption share. We rewrite (2) as

$$\mathsf{pd}_i \leftarrow \mathsf{PartDec}(\mathsf{ct}, \mathbf{s}_i) := \mathbf{a}^\intercal \mathbf{s}_i + \mathfrak{E}_i, \text{ where } |\mathfrak{E}_i| = n^{\omega(1)} \cdot B_{\mathsf{eval}},$$

and we assume $|e_{\mathsf{eval}}| < B_{\mathsf{eval}}$. Then, as in (3), the recovered noise is

$$e_{\mathsf{leak}} := b - \textstyle\sum_{i\in[N]} \mathsf{pd}_i - \lfloor \frac{Q}{2} \rfloor \cdot \overline{\mu} = e_{\mathsf{eval}} - \sum_{i\in[N]} \mathfrak{E}_i \approx_{\mathsf{s}} - \sum_{i\in[N]} \mathfrak{E}_i,$$

which information-theoretically hides e_{eval} (see Fig. 1 for graphical explanation). A notable drawback is that we need a super-polynomially large modulus Q to satisfy correctness. Thus, the size of ciphertexts, keys and shares also become super-polynomially large.

2.2 Flood-and-round [PS25]

Passelègue and Stehlé [PS25] also use noise flooding to smudge out the noise e_{eval} to prevent the leakage of information. Thus, their ThFHE scheme also requires a super-polynomially large modulus Q for the ciphertexts. However, they switch the modulus Q to a polynomially small modulus q by rounding the ciphertexts, thereby obtaining small decryption shares. This "flood-and-round"[4] procedure, denoted by $\mathsf{ServerDec}$, takes a super-polynomially long ciphertext (after homomorphic evaluation) $\mathsf{ct}_{\mathsf{eval}}$ as input, and outputs a sanitized (by noise flooding) and rounded polynomially short ciphertext $\mathsf{ct}_{\mathsf{dec}}$:

$$\mathsf{ServerDec}(\mathsf{ct}_{\mathsf{eval}} \in \mathbb{Z}_Q^{n+1}) \to \mathsf{ct}_{\mathsf{dec}} \in \mathbb{Z}_q^{n+1}.$$

$\mathsf{ServerDec}$ is executed by a special party called Server, who is untrusted (semi-honest) but assumed not to be corrupted by the adversary. This is because the ciphertext sanitization of $\mathsf{ServerDec}$ requires a (private) randomness of Server. The threshold decryption procedure is performed only on the output of $\mathsf{ServerDec}$ $\mathsf{ct}_{\mathsf{dec}}$. This is why it is called Server"Dec": it is a part of the decryption process.

We describe the details of $\mathsf{ServerDec}$. On the input $\mathsf{ct}_{\mathsf{eval}} := (\mathbf{a}, b)$ (defined as in (1)), it generates $\mathsf{ct}_{\mathsf{flood}} := \mathsf{ct}_{\mathsf{eval}} + (0, \mathfrak{E})$, where $\mathfrak{E}$ is a super-polynomially large (Gaussian) flooding noise. Then, by the smudging lemma for Gaussian [PS25, Lemma 2.1] (Lem. 3.12), we obtain

$$\begin{aligned}\mathsf{ct}_{\mathsf{flood}} &:= (\mathbf{a}, \mathbf{a}^\intercal \mathbf{s} + e_{\mathsf{eval}} + \mathfrak{E} + \lfloor \tfrac{Q}{2} \rfloor \cdot \mu) \\ &\approx_{\mathsf{s}} (\mathbf{a}, \mathbf{a}^\intercal \mathbf{s} + \mathfrak{E} + \lfloor \tfrac{Q}{2} \rfloor \cdot \mu).\end{aligned}$$

[4] Passelègue and Stehlé [PS25] call their scheme "double-flood-and-round", as they note that one more rounding can be performed on the partial decryption shares. However, they also mention that they do not use this for security. Thus, we simply call their method "flood-and-round".

Next, the ciphertext with a large modulus Q, constructed as $Q = p \cdot q$ for $p = \Omega(2^\kappa)$ and $q = \mathrm{poly}(\kappa)$, is rounded to a ciphertext with small modulus q:

$$\mathsf{ct}_{\mathsf{dec}} := \left(\left\lfloor \tfrac{1}{p} \cdot \mathbf{a} \right\rceil_{\sigma_0}, \left\lfloor \tfrac{1}{p} \cdot (\mathbf{a}^\intercal \mathbf{s} + \mathfrak{E} + \lfloor \tfrac{Q}{2} \rfloor \cdot \mu) \right\rceil_{\sigma_1} \right) := (\overline{\mathbf{a}}, \overline{b}),$$

where $\lfloor \cdot \rceil_\sigma$ denotes a randomized Gaussian rounding (Def. 3.13) and σ_0, σ_1 are public parameters. Then, we have

$$\mathsf{ct}_{\mathsf{dec}} = (\overline{\mathbf{a}}, \overline{b} = \overline{\mathbf{a}}^\intercal \mathbf{s} + \mathfrak{e} + \lfloor \tfrac{Q}{2} \rfloor \cdot \mu \bmod q), \tag{4}$$

$$\text{where } \mathfrak{e} = \left\lfloor \tfrac{1}{p} \cdot (-\mathbf{r}^\intercal \mathbf{s} + \mathfrak{E}) \right\rceil_{\sigma_1} \approx_{\mathsf{s}} D_{\mathbb{Z}, \sqrt{\sigma_0^2 \|\mathbf{s}\|^2 + \sigma_1^2}} \tag{5}$$

and $\mathbf{r} := \mathbf{a} - p \cdot \overline{\mathbf{a}}$ is a rounding error. The partial decryption of [PS25] is $\mathsf{pd}_i := \mathbf{a}^\intercal \mathbf{s}_i + e_i$ as in (2). Thus, the parties can recover the noise term as follows:

$$e_{\mathsf{leak}} := \overline{b} - \textstyle\sum_{i \in [N]} \mathsf{pd}_i - \lfloor \tfrac{Q}{2} \rfloor \cdot \overline{\mu} = \mathfrak{e} - \textstyle\sum_{i \in [N]} e_i.$$

However, $\mathfrak{e}$ depends on $\|\mathbf{s}\|$ as shown in (5). Because of this, [PS25] relies on the *known-norm* variant of LWE (Def. 3.15). Although known-norm LWE is reduced from standard LWE, its ring variant, known-covariance ring-LWE, has no reduction from standard assumptions such as RLWE. Thus, [PS25] noted it is an open question to extend their scheme to the ring- or module- setting.

2.3 Our Solution 1: Masking Partial Decryption

Before solving Question 1.1, we first solve Question 1.2 in Sect. 5: we propose a knLWE-based threshold FHE scheme with Shamir secret sharing by modifying ThFHE of [PS25] via simple masking technique.

Difficulty of applying Shamir secret sharing to ThFHE. Our idea of masking the partial decryption is inspired by the threshold signature scheme of [PKM+24]. They use a masking technique, which is also used for constructing secure aggregation protocol, e.g., [BBG+20; BGL+23], to address the difficulty of handling Lagrange coefficients in lattice-based threshold signatures. Our construction demonstrates that masking techniques can also be applied to threshold (fully homomorphic) encryption.

We first explain the difficulty of handling Lagrange coefficients in threshold (homomorphic) encryption. The ThFHE with Shamir secret sharing of [BGG+18, Constr. 5.11] constructs the partial decryption in the form as in (2), and the Lagrange coefficient is used in FinDec;

$$\mathsf{FinDec}(\mathsf{ct}, \{\mathsf{pd}_i\}_{i \in \mathsf{act}}) := \textstyle\sum_{i \in \mathsf{act}} \lambda_{\mathsf{act},i} \cdot \mathsf{pd}_i = \mathbf{a}^\intercal \mathbf{s}_i + \sum_{i \in \mathsf{act}} \lambda_{\mathsf{act},i} \cdot e_i \mod q,$$

where $\mathsf{act} \subseteq [N]$ is the set of active parties in the decryption session s.t. $|\mathsf{act}| \geq T$. However, this requires $q = \Omega((N!)^2)$ to ensure that the large noise term

$\sum_{i\in\mathsf{act}} \lambda_{\mathsf{act},i} \cdot e_i$ relatively small to q [ABV+12]. One naive attempt to address this inefficiency is to include the Lagrange coefficient in the partial decryption:

$$\mathsf{pd}_i \leftarrow \mathsf{PartDec}(\mathsf{ct}, \mathbf{s}_i, \mathsf{act}) := \lambda_{\mathsf{act},i} \cdot \mathbf{a}^\intercal \mathbf{s}_i + e_i \mod q. \tag{6}$$

While correctness is satisfied, there is a trivial attack on this construction since the adversary can choose act to craft the coefficients $\lambda_{\mathsf{act},i}$, and e_i is relatively small compared to q. Let $\mathsf{act} \subseteq [N]$ s.t. $\hat{\lambda} := \lambda_{\mathsf{act},i}$ satisfies $q = \hat{\lambda} \cdot q'$ for some integer $q' < q$, and assume $|e_i| < \hat{\lambda}$ holds. Then, we have $b'_i := \lfloor \frac{\mathsf{pd}_i}{\hat{\lambda}} \rfloor \bmod q' = \mathbf{a}^\intercal \mathbf{s}_i \bmod q'$. By querying a sufficient number of such b'_i, the lower bits of the secret share $(\mathbf{s}_i \bmod q')$ can be recovered via simple linear algebra.

Our construction. We explain the construction of our Shamir-secret-sharing-friendly knLWE-based ThFHE. At the setup, each party P_i is given $\mathsf{seed}_i \xleftarrow{\$} \{0,1\}^\kappa$ and generates pseudorandom masks by $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$, where PRF is (assumed to be) a pseudorandom function and sid is a session ID of the threshold decryption sessions. Then, we construct the decryption shares as:

$$\mathsf{pd}_i \leftarrow \mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}}, \mathbf{s}_i, \mathsf{act}) := \lambda_{\mathsf{act},i} \cdot \mathbf{a}^\intercal \mathbf{s}_i + \mathfrak{m}_i \bmod q. \tag{7}$$

The mask value $\mathfrak{m}_i$ (of any uncorrupted party P_i) is essentially pseudorandom over $\mathbb{Z}_q$ (except that the sum of all masks $\sum_{i\in\mathsf{act}} \mathfrak{m}_i$ is included in $\mathsf{ct}'_{\mathsf{dec}}$ as described in (8)). Thus, the trivial attack to (6) is not valid to our construction since $\mathfrak{m}_i$ is not small and pd_i is also essentially pseudorandom.

Note that we have $\sum_{i\in\mathsf{act}} \mathsf{pd}_i = \mathbf{a}^\intercal \mathbf{s} + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q$ by construction of Shamir secret sharing. Hence, for correctness to be hold, the seed values $\{\mathsf{seed}_i\}_{i\in[N]}$ are also shared with the Server: The Server also calculates $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$ for $i \in \mathsf{act}$ and adds $\sum_{i\in\mathsf{act}} \mathfrak{m}_i$ to $\mathsf{ct}_{\mathsf{dec}}$ in (4), then outputs:

$$\mathsf{ct}'_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}' := \overline{\mathbf{a}}^\intercal \mathbf{s} + \mathfrak{e} + \lfloor \tfrac{q}{2} \rfloor \cdot \mu + \textstyle\sum_{i\in[N]} \mathfrak{m}_i \bmod q). \tag{8}$$

Note that Server is untrusted (semi-honest) but assumed not to collude with the adversary, as in [PS25]; Server never receives any partial decryption, and thus knowledge of $\{\mathsf{seed}_i\}_{i\in[N]}$ is meaningless to the untrusted Server.

Then, given $\{\mathsf{pd}_i\}_{i\in\mathsf{act}}$, parties perform FinDec to decrypt $\mathsf{ct}'_{\mathsf{dec}}$ as follows:

$$\overline{\mu} := \lfloor (\overline{b}' - \textstyle\sum_{i\in\mathsf{act}} \mathsf{pd}_i) / \lfloor \tfrac{q}{2} \rfloor \rceil = \mu + \lfloor \mathfrak{e} / \lfloor \tfrac{q}{2} \rfloor \rceil.$$

The correctness holds if we select the parameters s.t. $|\mathfrak{e}| < \lfloor \frac{q}{4} \rfloor$ holds (with overwhelming probability).

We now explain how to simulate the partial decryption in the security proof. The recovered noise term after decryption is

$$e_{\mathsf{leak}} := \overline{b}' - \textstyle\sum_{i\in\mathsf{act}} \mathsf{pd}_i - \lfloor \tfrac{q}{2} \rfloor \cdot \overline{\mu} = \mathfrak{e} \approx_{\mathsf{s}} D_{\mathbb{Z}, \sqrt{\sigma_0^2 \|\mathbf{s}\|^2 + \sigma_1^2}} \tag{9}$$

as described in (5). Let $|\mathsf{act}| = T$ and $\mathsf{corr} = \mathsf{act}\backslash\{h\}$ be the set of the corrupted parties. Then, by (5), we can simulate pd_h as follows:

$$\mathsf{pd}_h \approx_{\mathsf{s}} \vec{b}' - \lfloor \tfrac{q}{2} \rfloor \cdot \mu - \textstyle\sum_{i \in \mathsf{corr}} \mathsf{pd}_i + \mathfrak{e}, \text{ where } \mathfrak{e} \leftarrow D_{\mathbb{Z}, \sqrt{\sigma_0^2 \|\mathbf{s}\|^2 + \sigma_1^2}} \tag{10}$$

Note that we now rely on knLWE as in [PS25]; $\|\mathbf{s}\|$ is now public value and thus above $\mathfrak{e}$ is publicly sampleable. Given simulated pd_h instead of real one, the adversary only has information related to $\{\mathbf{s}_i\}_{i \in \mathsf{corr}}$. Due to the security of secret sharing, no information about $\mathbf{s}_h$ and $\mathbf{s}$ is obtained from $\{\mathbf{s}_i\}_{i \in \mathsf{corr}}$. We explain how to eliminate the need of knLWE in subsequent Sect. 2.4.

Side benefit: Compactness w.r.t. N. The recovered noise term after FinDec of the prior works [AJL+12; PS25] is in the form of $e_{\mathsf{eval}} + \sum_{i \in [N]} e_i$ as in (3) (for N-out-of-N threshold decryption). Thus, its upper bound is $O(N)$ w.r.t. N, and the moduli (Q and q) should also be $\Omega(N)$ to satisfy correctness. Furthermore, for general T-out-of-N threshold decryption, this factor becomes larger, e.g., $\Omega(N^{4.2})$ or $\Omega(\binom{T}{N})$, since the recovered noise term after FinDec is in the form of $e_{\mathsf{eval}} + \sum_{i \in S_i} e_i$, where S_i is the index set s.t. $|S_i| = O(N^{4.2})$.

In contrast, the recovered noise term after decryption of our scheme is only $\mathfrak{e}$ (or $\mathfrak{e} + \zeta$), whose size is independent on N ($=O(1)$ w.r.t. N). Thus, we achieve compactness. This is because we do not add a noise in the partial decryption share (7); instead, we add $\mathfrak{m}_i$ that is canceled out to be zero after FinDec. Therefore, modulus, secret shares and decryption shares are $O(1)$ w.r.t. N.

2.4 Our Solution 2: "Noise Padding"

We next solve Question 1.1 in Sect. 6: we modify our scheme described in Sect. 2.3 to construct a ThFHE scheme from *standard* LWE. We modify the partial decryption to prevent the leakage of $\|\mathbf{s}\|$, by a core technique which we call "noise padding".

Our construction. Our goal is to add a small noise ζ, which we call "padding noise", to adjust the distribution of the recovered noise term e_{leak} of (9). (see Fig. 2 for a graphical explanation).

At KeyGen, we sample the padding noise ζ depending on $\mathbf{s}$ as follows:

$$\zeta \leftarrow D_{\mathbb{Z}, \sigma_{\mathsf{padding}}}, \ \sigma_{\mathsf{padding}} := \sqrt{\sigma_0^2 (B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2)},$$

where B_{pub} is a public constant (s.t. $B_{\mathsf{pub}} > \|\mathbf{s}\|$ for any $\mathbf{s}$). Then, for $\mathfrak{e}$ in (5),

$$e_{\mathsf{leak}} := \mathfrak{e} - \zeta \approx_{\mathsf{s}} D_{\mathbb{Z}, \sigma_{\mathsf{pub}}}, \ \sigma_{\mathsf{pub}} := \sqrt{\sigma_0^2 B_{\mathsf{pub}}^2 + \sigma_1^2} \tag{11}$$

holds. Thus, if the noise term recovered after FinDec is e_{leak} as above, it does not leak any information about $\|\mathbf{s}\|$: We can construct a ThFHE scheme (directly) from standard LWE.

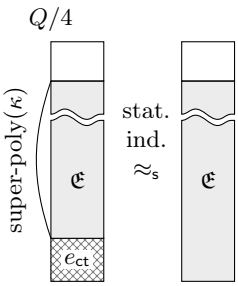


Fig. 1. Noise flooding [AJL+12]: Smudges out e_{eval} by adding super-polynomially large flooding noise $\mathfrak{E}$.

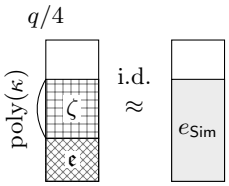


Fig. 2. Our "noise padding": Add ζ ($\mathrm{V}(\zeta) = \sigma^2(B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2)$) to $\mathfrak{e}$ ($\mathrm{V}(\mathfrak{e}) = \sigma^2\|\mathbf{s}\|^2$), obtaining the simulatable noise e_{Sim} ($\mathrm{V}(e_{\mathsf{Sim}}) = \sigma^2 B_{\mathsf{pub}}^2$).

We now explain how to securely add ζ in the noise term recovered after FinDec. Since ζ contains information about $\|\mathbf{s}\|$, it cannot be sent directly to the untrusted (semi-honest) Server or any parties. Hence, we perform secret sharing on the padding noise ζ, as with the secret key $\mathbf{s}$: $\mathsf{Shamir.Share}(\zeta) \to \{\zeta_1, \ldots, \zeta_N\}$. Then, we modify the partial decryption of (7) by adding the shares ζ_i;

$$\mathsf{pd}_i \leftarrow \mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}}, \mathbf{s}_i, \zeta_i, \mathsf{act}) := \lambda_{\mathsf{act},i} \cdot (\mathbf{a}^\intercal \mathbf{s}_i + \zeta_i) + \mathfrak{m}_i \bmod q. \tag{12}$$

We now successfully obtain that the noise term recovered after FinDec is

$$e_{\mathsf{leak}} := \overline{b} - \textstyle\sum_{i\in\mathsf{act}} \mathsf{pd}_i - \lfloor \frac{q}{2} \rfloor \cdot \overline{\mu} = \mathfrak{e} - \textstyle\sum_{i\in\mathsf{act}} \zeta_i = \mathfrak{e} - \zeta.$$

Hence, as in (10), we can simulate the partial decryption shares as follows:

$$\mathsf{pd}_h \approx_{\mathsf{s}} \overline{b}' - \lfloor \tfrac{q}{2} \rfloor \cdot \mu - \textstyle\sum_{i\in\mathsf{corr}} \mathsf{pd}_i + e_{\mathsf{Sim}}, \text{ where } e_{\mathsf{Sim}} \leftarrow D_{\mathbb{Z},\sigma_{\mathsf{pub}}},$$

where σ_{pub} is defined as in (11). Importantly, the distribution of $D_{\mathbb{Z},\sigma_{\mathsf{pub}}}$ contains no information about the secret key $\mathbf{s}$. Therefore, we can construct our scheme directly from *standard* LWE, without relying on known-norm LWE as in [PS25].

Limitation. Our construction needs a fresh padding noise ζ for each decryption session. We need to a priori fix the query bound $L_{\mathsf{Dec}} = \mathrm{poly}(\kappa)$ (as with [BS23]) and generate padding noise as many as L_{Dec}; $\boldsymbol{\zeta} := (\zeta^{(1)}, \ldots, \zeta^{(L_{\mathsf{Dec}})}) \leftarrow D_{\mathbb{Z},\sigma_{\mathsf{padding}}}^{L_{\mathsf{Dec}}}$. (Precisely, (12) is written as $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot (\mathbf{a}^\intercal \mathbf{s}_i + \zeta_i^{(\mathsf{sid})}) + \mathfrak{m}_i \bmod q$.) Then, we perform Shamir secret sharing on $\boldsymbol{\zeta}$ and parties need to store the shares $\{\boldsymbol{\zeta}_i \in \mathbb{Z}_q^{L_{\mathsf{Dec}}})\}_{i\in[N]} \leftarrow \mathsf{Shamir.Share}(\boldsymbol{\zeta})$: Thus, the storage cost of each party for the shares is $O(L_{\mathsf{Dec}})$. However, we note that the value L_{Dec} is independent of any other security-related parameters, unlike in [BS23]. Thus, this limitation would not be problematic for some use-cases in which decryption is not needed to perform many times. We leave it as an open problem to remove this limitation. Note that our previous scheme described in Sect. 2.3 (and presented in Sect. 5) does not have this limitation since the scheme does not use padding noise.

3 Preliminaries

In this section, we provide necessary definitions and lemmas. Some standard definition (of PRF) is deferred to Appendix A.

3.1 Notation

For $N \in \mathbb{N}$, we define $[N] := \{i \in \mathbb{N} \mid 1 \leq i \leq N\}$. The number of elements in a set S is denoted by $|S|$. Unless otherwise specified, we treat κ as a security parameter. We write $\mathrm{negl}(\kappa) = \kappa^{-\omega(1)}$ for the set of negligible functions and $\mathrm{poly}(\kappa) = \kappa^{O(1)}$ for the set of polynomial functions. We call $1 - \mathrm{negl}(\kappa)$ overwhelming functions.

We use bold lower-case for vectors and bold upper-case for matrices. The transpose of $\mathbf{x}$ is written as $\mathbf{x}^\intercal$. The l_2- and l_∞-norm are denoted by $\|\cdot\|$ and $\|\cdot\|_\infty$, respectively. The identity matrix is denoted by $\mathbf{I}_n \in \mathbb{Z}^{n\times n}$. We write $\mathbf{\Sigma} \succ 0$ if $\mathbf{\Sigma}$ is positive definite. A square root of $\mathbf{\Sigma} \succ 0$ is a nonsingular matrix $\mathbf{S}$ such that $\mathbf{S}\mathbf{S}^\intercal = \mathbf{\Sigma}$, which is written as $\mathbf{S} = \sqrt{\mathbf{\Sigma}}$. The largest and smallest singular values of $\mathbf{S}$ are denoted by $\sigma_{\max}(\mathbf{S})$ and $\sigma_{\min}(\mathbf{S})$, respectively. Similarly, the largest and smallest eigenvalues are denoted by $\lambda_{\max}(\mathbf{S})$ and $\lambda_{\min}(\mathbf{S})$, respectively. We denote by $\|\mathbf{S}\|$ the matrix norm of $\mathbf{S}$ induced by the l_2-norm. Let $\|\mathbf{S}\|_{\mathrm{len}} = \max_{i\in[n]} \|\mathbf{s}_i\|$, where $\mathbf{s}_i$ is the i-th column vector of $\mathbf{S}$.

We denote the uniform distribution over a set S by $\mathcal{U}(S)$. For any distributions χ_1 and χ_2, we denote by $\chi_1 + \chi_2$ the distribution $\{X_1 + X_2 \mid X_1 \leftarrow \chi_1 \text{ and } X_2 \leftarrow \chi_2 \text{ are mutually independent}\}$. We denote $X_1 \approx X_2$ if X_1 and X_2 are identically distributed. We write $\chi_1 \approx_{\mathsf{s}} \chi_2$ (resp. $\chi_1 \approx_{\mathsf{c}} \chi_2$) when χ_1 and χ_2 are statistically (resp. computationally) indistinguishable. For $B > 0$, we say a distribution χ over $\mathbb{R}$ is B-bounded if $\Pr_{X\leftarrow\chi}[|X| \geq B] = \mathrm{negl}(\kappa)$.

3.2 Polynomial Ring

We denote $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$ for a modulus $q \in \mathbb{N}$. We let $\mathcal{R} := \mathbb{Z}[X]/(X^n + 1)$ and $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^n + 1)$ for n a power of 2. For ease of notation, we define the *coefficient vector*, *coefficient matrix* and *coefficient Gram matrix* as follows:

Definition 3.1. *Let $a = \sum_{i=0}^{n-1} a_i X^i \in \mathcal{R}$, and define the coefficient vector of a as $\mathbf{a} := \mathsf{vec}(a) := (a_0, a_1, \ldots, a_{n-1})^\intercal \in \mathbb{Z}^n$. Let $\mathbf{P} := \left(\begin{array}{c|c} \mathbf{0} & -1 \\ \hline \mathbf{I}_{n-1} & \mathbf{0} \end{array}\right) \in \mathbb{Z}^{n\times n}$ be a (negacyclic) permutation matrix, and define the coefficient matrix of a as $\mathbf{A} := \mathsf{mat}(a) := \left(\mathbf{a}\ \mathbf{P}\mathbf{a} \cdots \mathbf{P}^{n-1}\mathbf{a}\right) \in \mathbb{Z}^{n\times n}$. The coefficient Gram matrix of a is defined as $\mathbf{\Sigma}_a := \mathsf{Gram}(a) := \mathbf{A}\mathbf{A}^\intercal \in \mathbb{Z}^{n\times n}$.*

For $a \in \mathcal{R}$, we define $\|a\| := \|\mathsf{vec}(a)\|$ and $\|a\|_\infty := \|\mathsf{vec}(a)\|_\infty$. For the distribution χ over $\mathbb{Z}^n$, we define $\mathcal{R}(\chi) := \{a \in \mathcal{R} \mid \mathsf{vec}(a) \sim \chi\}$. Note that the coefficient matrix is useful for describing a product:

Fact 3.2. *For $r, e \in \mathcal{R}$, we have $\mathsf{vec}(re) = \mathbf{R}\mathbf{e} = \mathbf{E}\mathbf{r}$, where $\mathbf{R} := \mathsf{mat}(r)$, $\mathbf{r} := \mathsf{vec}(r)$, $\mathbf{E} := \mathsf{mat}(e)$ and $\mathbf{e} := \mathsf{vec}(e)$.*

3.3 Lattices and Gaussians

A lattice $\mathcal{L}$ is the set of all integer linear combinations of linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, i.e., $\mathcal{L} = \{\sum_{i=1}^n z_i \mathbf{b}_i \mid \mathbf{z} \in \mathbb{Z}^n\}$. The rank of this lattice is n and its dimension is m. If $n = m$, then the lattice is called full rank. If we arrange the vectors $\mathbf{b}_i$ as the columns of a matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$, then we can write $\mathcal{L} := \mathcal{L}(\mathbf{B}) = \{\mathbf{Bz} \mid \mathbf{z} \in \mathbb{Z}^n\} = \mathbf{B} \cdot \mathbb{Z}^n$. For arbitrary $\mathbf{c} \in \mathbb{R}^m$, a coset of lattice $\mathcal{L}$ is defined as $A := \mathcal{L} + \mathbf{c} := \{\mathbf{v} + \mathbf{c} \mid \mathbf{v} \in \mathcal{L}\}$. The dual of a lattice $\mathcal{L}$ is $\widehat{\mathcal{L}} := \{\mathbf{x} \mid \forall \mathbf{y} \in \mathcal{L}, \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}\}$. For the n-rank lattice $\mathcal{L}$ and $i \in [n]$, the successive minimum $\tilde{\lambda}_i(\mathcal{L})$ is defined as the radius of the smallest ball that contains i linearly independent vectors in $\mathcal{L}$.

For a rank-n matrix $\mathbf{S} \in \mathbb{R}^{n \times m}$, the Gaussian function on $\mathbb{R}^n$ with the (scaled) covariance matrix $\mathbf{\Sigma} = \mathbf{SS}^\intercal \in \mathbb{R}^{n \times n}$ and a center $\mathbf{c} \in \mathbb{R}^n$ is defined as $\rho_{\mathbf{S},\mathbf{c}}(\mathbf{x}) := \exp(-\pi(\mathbf{x} - \mathbf{c})^\intercal (\mathbf{SS}^\intercal)^{-1}(\mathbf{x} - \mathbf{c}))$. Since $\rho_{\mathbf{S},\mathbf{c}}(\mathbf{x})$ is determined by $\mathbf{\Sigma}$ (and $\mathbf{c}$), we have $\rho_{\mathbf{S},\mathbf{c}} = \rho_{\sqrt{\mathbf{\Sigma}},\mathbf{c}}$. When $\mathbf{S} = s\mathbf{I}_n$, we write $\rho_{\mathbf{S},\mathbf{c}}$ as $\rho_{s,\mathbf{c}}$. For any set $A \subseteq \mathbb{R}^n$, we define $\rho_{\mathbf{S},\mathbf{c}}(A) := \sum_{\mathbf{x} \in A} \rho_{\mathbf{S},\mathbf{c}}(\mathbf{x})$. The subscript is $\mathbf{c} = \mathbf{0}$ when omitted.

We define the discrete Gaussian distribution over the lattice $\mathcal{L}$ as follows:

Definition 3.3 (Discrete Gaussian). *For a full column-rank matrix $\mathbf{S}$, the discrete Gaussian distribution over a lattice $\mathcal{L}$ with a center $\mathbf{c} \in \mathbb{R}^n$ is defined as $\forall \mathbf{x} \in \mathcal{L}, D_{\mathcal{L},\mathbf{S},\mathbf{c}}(\mathbf{x}) = \rho_{\mathbf{S},\mathbf{c}}(\mathbf{x})/\rho_{\mathbf{S},\mathbf{c}}(\mathcal{L})$. In particular, when $\mathbf{SS}^\intercal = s^2\mathbf{I}_n$ for some $s > 0$, we write $D_{\mathcal{L},\mathbf{S},\mathbf{c}}$ as $D_{\mathcal{L},s,\mathbf{c}}$ and call it the spherical discrete Gaussian distribution. The subscript is $\mathbf{c} = \mathbf{0}$ when omitted.*

The *smoothing parameter* of $\mathcal{L}$ is defined as $\eta_\epsilon(\mathcal{L}) = \min\{s \mid \rho_{1/s}(\widehat{\mathcal{L}}) \leq 1 + \epsilon\}$ for $\epsilon > 0$. Unless otherwise specified, we set $\epsilon = \mathrm{negl}(\kappa)$. An upper-bound of $\eta_\epsilon(\mathcal{L})$ is obtained with the successive minimum[5] $\tilde{\lambda}_n(\mathcal{L})$:

Lemma 3.4 ([MR07, Lemma 3.3]). *Define $\eta_\epsilon^+(\mathbb{Z}^n) := \sqrt{\ln(2n(1 + 1/\epsilon))/\pi}$. For any rank-$n$ lattice $\mathcal{L}$ and any $\epsilon > 0$, we have $\eta_\epsilon(\mathcal{L}) \leq \tilde{\lambda}_n(\mathcal{L}) \cdot \eta_\epsilon^+(\mathbb{Z}^n)$.*

In particular, $\eta_\epsilon(\mathbb{Z}^n) \leq \eta_\epsilon^+(\mathbb{Z}^n)$ holds; for any $\omega(\sqrt{\log n})$ function, there is $\epsilon = \mathrm{negl}(\kappa)$ s.t $\eta_\epsilon(\mathbb{Z}^n) \leq \omega(\sqrt{\log n})$.

The smoothing parameter is extended to matrices as follows:

Definition 3.5 ([Pei10, Definition 2.3]). *Let $\mathbf{\Sigma} \succ 0$ be any positive definite matrix. For any lattice $\mathcal{L}$, we say that $\sqrt{\mathbf{\Sigma}} \geq \eta_\epsilon(\mathcal{L})$ if $1 \geq \eta_\epsilon(\sqrt{\mathbf{\Sigma}}^{-1}\mathcal{L})$.*

We obtain a sufficient condition for showing $\sqrt{\mathbf{\Sigma}} \geq \eta_\epsilon(\mathcal{L})$ as follows[6]:

Fact 3.6. *For any full-rank lattice $\mathcal{L}(\mathbf{B})(= \mathbf{B}\mathbb{Z}^n)$ and $\mathbf{\Sigma} \succ 0$, we have $\sqrt{\mathbf{\Sigma}} \geq \eta_\epsilon(\mathcal{L})$ if $\sigma_{\min}(\sqrt{\mathbf{\Sigma}}) \geq \|\mathbf{B}\|_{\mathrm{len}} \eta_\epsilon^+(\mathbb{Z}^n) (\geq \eta_\epsilon(\mathcal{L}))$.*

Proof. By Lem. 3.4 and the hypothesis, we have $\eta_\epsilon(\sqrt{\mathbf{\Sigma}}^{-1}\mathcal{L}) \leq \tilde{\lambda}_n(\sqrt{\mathbf{\Sigma}}^{-1}\mathcal{L})\eta_\epsilon^+(\mathbb{Z}^n) \leq \|\sqrt{\mathbf{\Sigma}}^{-1}\mathbf{B}\|_{\mathrm{len}} \eta_\epsilon^+(\mathbb{Z}^n) \leq \|\sqrt{\mathbf{\Sigma}}^{-1}\| \|\mathbf{B}\|_{\mathrm{len}} \eta_\epsilon^+(\mathbb{Z}^n) \leq 1$. □

[5] Although [GPV08, Lemma 3.1] provides a sharper bound with the *Gram–Schmidt minimum*, we use Lem. 3.4 for ease of analysis.

[6] Although sharper bounds can be obtained by the successive minimum $\tilde{\lambda}_n$ or Gram–Schmidt minimum, we rely on $\|\cdot\|_{\mathrm{len}}$ for ease of analysis.

The linear sum of discrete Gaussians is close to a discrete Gaussian:

Lemma 3.7 ([MP13, Theorem 3.3]). *Let $\mathcal{L}$ be an n-dimensional lattice, $\epsilon = \mathrm{negl}(\kappa)$, $\mathbf{z} \neq \mathbf{0} \in \mathbb{Z}^m$, $s_i \geq \sqrt{2}\|\mathbf{z}\|_\infty \eta_\epsilon(\mathcal{L})$, $A_i := \mathcal{L} + \mathbf{c}_i$ be arbitrary cosets of $\mathcal{L}$, and $\mathbf{y}_i \leftarrow D_{A_i,s_i}$ for $i \in [m]$. Then, we have $\mathbf{y} := \sum_{i=1}^m z_i \mathbf{y}_i \approx_\mathsf{s} D_{Y,s}$, where $Y = \gcd(\mathbf{z})\mathcal{L} + \sum_{i=1}^m z_i \mathbf{c}_i$, and $s = \sqrt{\sum_{i=1}^m (z_i s_i)^2}$.*

In particular, if $\gcd(\mathbf{z}) = 1$ and $\sum_{i=1}^m z_i \mathbf{c}_i \in \mathcal{L}$, then $\mathbf{y} \approx_\mathsf{s} D_{\mathcal{L},s}$.

More generally, the sum of two ellipsoid discrete Gaussians is statistically close to an ellipsoid discrete Gaussian:

Lemma 3.8 ([GMPW20, Thm. 3]). *Let $\epsilon = \mathrm{negl}(\kappa)$. Let A_1 and A_2 be cosets of full-rank lattices $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively. Let $\mathbf{\Sigma}_1, \mathbf{\Sigma}_2 \succ 0$ be positive definite matrices and define $\mathbf{\Sigma}_3 := (\mathbf{\Sigma}_1^{-1} + \mathbf{\Sigma}_2^{-1})^{-1}$. Let $X := \{(\mathbf{x}_1, \mathbf{x}_2) \mid \mathbf{x}_1 \leftarrow D_{A_1,\sqrt{\mathbf{\Sigma}_1}}, \mathbf{x}_2 \leftarrow \mathbf{x}_1 + D_{A_2-\mathbf{x}_1,\sqrt{\mathbf{\Sigma}_2}}\}$. If $\sqrt{\mathbf{\Sigma}_2} \geq \eta_\epsilon(\mathcal{L}_2)$ and $\sqrt{\mathbf{\Sigma}_3} \geq \eta_\epsilon(\mathcal{L}_1)$ hold, The marginal distribution of $\mathbf{x}_2$ in X is statistically close to $D_{A_2,\sqrt{\mathbf{\Sigma}_1+\mathbf{\Sigma}_2}}$.*

In particular, when $A_1 \subseteq A_2$, we can simplify the above lemma:

Corollary 3.9. *Let $\epsilon = \mathrm{negl}(\kappa)$. Let $\mathbf{\Sigma}_1, \mathbf{\Sigma}_2 \succ 0$ be positive definite matrices and define $\mathbf{\Sigma}_3 := (\mathbf{\Sigma}_1^{-1} + \mathbf{\Sigma}_2^{-1})^{-1}$. Let A_1 and A_2 be cosets of full-rank lattices $\mathcal{L}_1$ and $\mathcal{L}_2$ such that $A_1 \subseteq A_2$. If $\sqrt{\mathbf{\Sigma}_2} \geq \eta_\epsilon(\mathcal{L}_2)$ and $\sqrt{\mathbf{\Sigma}_3} \geq \eta_\epsilon(\mathcal{L}_1)$ hold, we have $D_{A_1,\sqrt{\mathbf{\Sigma}_1}} + D_{A_2,\sqrt{\mathbf{\Sigma}_2}} \approx_\mathsf{s} D_{A_2,\sqrt{\mathbf{\Sigma}_1+\mathbf{\Sigma}_2}}$.*

The linear transformation of a discrete Gaussian is analysed as follows:

Lemma 3.10 ([GMPW20, Lemma 1]). *For any lattice coset $A = \mathcal{L}+\mathbf{c} \subseteq \mathbb{R}^n$ and matrices $\mathbf{S}, \mathbf{T}$ representing linear functions where $\mathbf{T}$ is injective on A, we have $\mathbf{T} \cdot D_{A,\mathbf{S}} = D_{\mathbf{T}\cdot A,\mathbf{TS}}$. In particular, for any $s, t > 0$, we have $t \cdot D_{\mathbb{Z},s} = D_{t\mathbb{Z},ts}$.*

The tail bound of discrete Gaussian can be obtained as follows:

Lemma 3.11. *For any $\epsilon \in (0,1)$, $s \geq \eta_\epsilon^+(\mathbb{Z})$ and $t \in \mathbb{N}$, we have $\Pr_{x \leftarrow D_{\mathbb{Z},s}}[|x| \geq t] \leq \frac{s}{(1-\epsilon)\pi t} e^{-\pi t^2/s^2}$. In particular, for any $t = s \cdot \Omega(\sqrt{\kappa})$, we have $\Pr_{x \leftarrow D_{\mathbb{Z},s}}[|x| \geq t] = \mathrm{negl}(\kappa)$, i.e., $D_{\mathbb{Z},s}$ is $s \cdot \Omega(\sqrt{\kappa})$-bounded.*

Passelègue and Stehlé [PS25] presented the smudging lemma for Gaussian. We use the following lemma adapted for multivariate Gaussian:

Lemma 3.12 (Adapted from [PS25, Lemma 2.1]). *For $n = \mathrm{poly}(\kappa)$, $\mathbf{c} \in \mathbb{R}^n$ and $\sigma = \Omega(\|\mathbf{c}\|_\infty 2^\kappa)$, we have $D_{\mathbb{Z}^n,\sigma,\mathbf{c}} \approx_\mathsf{s} D_{\mathbb{Z}^n,\sigma}$.*

Finally, we define the randomized Gaussian rounding operation:

Definition 3.13 ($\lfloor\cdot\rceil_\sigma$). *For any $n \in \mathbb{N}$, $\mathbf{c} \in \mathbb{R}^n$ and $\sigma > 0$, $\lfloor\mathbf{c}\rceil_\sigma$ denotes the randomized Gaussian rounding that returns $\mathbf{y} \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{c}}$ (or equivalently, samples $\mathbf{z} \leftarrow D_{\mathbb{Z}^n-\mathbf{c},\sigma}$ and and returns $\mathbf{z} + \mathbf{c}$).*

3.4 Learning with Errors (LWE)

The LWE distribution and the LWE problem are defined as follows:

Definition 3.14 (LWE). *Let n, m, q be integers and χ be an error distribution over $\mathbb{Z}_q$. The LWE distribution is defined as*

$$\mathsf{LWE}(n, m, q, \chi) := \left\{(\mathbf{A}, \mathbf{b} := \mathbf{As} + \mathbf{e}) \mid \mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{m \times n}), \mathbf{s} \leftarrow \chi^n, \mathbf{e} \leftarrow \chi^m\right\}.$$

The advantage of an algorithm $\mathcal{A}$ for solving the (decision) LWE problem is defined as

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{LWE}} = |\Pr[\mathcal{A}(\mathbf{A}, \mathbf{As} + \mathbf{e}) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)) = 1]|.$$

We say that LWE is hard if $\mathrm{Adv}_{\mathcal{A}}^{\mathsf{LWE}} = \mathrm{negl}(n)$ for any PPT algorithm $\mathcal{A}$.

Micciancio and Suhl [MS25] introduced a variant of LWE called "known-norm" LWE, where the norm of secret key $\mathbf{s}$ (and/or the norm of error vector $\mathbf{e}$) are known to the adversary. A reduction from (standard) LWE to known-norm LWE is also shown in [MS25]. We define known-norm LWE as follows:

Definition 3.15 (Known-norm LWE). *Let n, m, q be integers and χ be an error distribution over $\mathbb{Z}_q$, and let $(\mathbf{A}, \mathbf{b} := \mathbf{As} + \mathbf{e}) \leftarrow \mathsf{LWE}(n, m, q, \chi)$. The advantage of $\mathcal{A}$ for solving the decision known-norm LWE (knLWE) is*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{knLWE}} = |\Pr[\mathcal{A}(\mathbf{A}, \mathbf{As} + \mathbf{e}, \|\mathbf{s}\|) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u} \leftarrow \mathcal{U}(\mathbb{Z}_q^n), \|\mathbf{s}\|) = 1]|.$$

The module-LWE problem (MLWE) is defined as follows:

Definition 3.16 (Module-LWE). *Let $k, m, q \in \mathbb{N}$ and χ be a distribution over $\mathcal{R}_q$. The module-LWE distribution is defined as $\mathsf{MLWE}(k, m, q, \chi) := \{(\mathbf{A}, \mathbf{b} := \mathbf{As} + \mathbf{e}) \mid \mathbf{A} \xleftarrow{\$} \mathcal{R}_q^{m \times k}, \mathbf{s} \leftarrow \chi^k, \mathbf{e} \leftarrow \chi^m\}$. The advantage of an algorithm $\mathcal{A}$ for solving the (decision) module-LWE problem is defined as*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{MLWE}} = |\Pr[\mathcal{A}(\mathbf{A}, \mathbf{As} + \mathbf{e}) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u} \leftarrow \mathcal{U}(\mathcal{R}_q^k)) = 1]|.$$

3.5 Shamir Secret Sharing

We describe a construction of Shamir secret sharing [Sha79] as follows:

Construction 3.17. *(T, N)-Shamir secret sharing for a finite field $\mathbb{K}$ is a tuple of PPT algorithms $\mathsf{Shamir}_{\mathbb{K},T,N} := (\mathsf{Share}, \mathsf{Recon})$ defined as follows:*

- $\mathsf{Share}(s \in \mathbb{K})$*: Choose a degree $T-1$ polynomial $P \in \mathbb{K}[X]$ that satisfies $P(0) = s$ and outputs $(s_1, \ldots, s_N) = (P(1), \ldots, P(N)) \in \mathbb{K}^N$.*
- $\mathsf{Recon}(\{s_i\}_{i \in S \subseteq [N]})$*: Outputs $s = \sum_{i \in S} \lambda_{S,i} s_i$, where $\lambda_{S,i} = \prod_{j \in S \setminus \{i\}} \left(\frac{-j}{i-j}\right)$.*

$\mathsf{Shamir}_{\mathbb{K},T,N}$ satisfies correctness and privacy:

- *Correctness: For all $S \subseteq [N]$ s.t. $|S| \geq T$, $s \in \mathbb{K}$, and $(s_1, \ldots, s_N) \leftarrow \mathsf{Shamir}_{\mathbb{K},T,N}.\mathsf{Share}(s)$, we have $\mathsf{Shamir}_{\mathbb{K},T,N}.\mathsf{Combine}(\{\mathsf{s}_i\}_{i \in S}) = s$.*
- *Privacy: For all $S \subset [N]$ s.t. $|S| < T$, $s^{(0)}, s^{(1)} \in \mathbb{K}$, and $(s_1^{(b)}, \ldots, s_N^{(b)}) \leftarrow \mathsf{Shamir}_{\mathbb{K},T,N}.\mathsf{Share}(s^{(b)})$ for $b \in \{0,1\}$, we have $\{s_i^{(0)}\}_{i \in S} \approx \{s_i^{(1)}\}_{i \in S}$.*

The above Shamir secret sharing scheme naturally extends to secrets in vector form, by performing the algorithm in an elementwise manner.

Algorithm 1: LWE-based underlying FHE scheme $\mathsf{FHE} := (\mathsf{PP}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$

$\underline{\mathsf{PP}(1^\kappa) \to \mathsf{pp}}$:

1 **return** $\mathsf{pp} := (n, m, Q, \chi_{\mathsf{lwe}}, B_{\mathsf{eval}})$

$\underline{\mathsf{KeyGen}()}$:

2 $\mathsf{sk} := \mathbf{s} \leftarrow \chi_{\mathsf{lwe}}^n$, $\mathsf{pk} := (\mathbf{A}, \mathbf{b} := \mathbf{As} + \mathbf{e}) \sim \mathsf{LWE}(n, m, Q, \chi_{\mathsf{lwe}})$

3 Generate the evaluation key evk ▷ We omit the details of evk

4 **return** $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mu \in \mathcal{M} := \{0,1\})}$:

5 $\mathbf{r} \leftarrow \chi_{\mathsf{lwe}}^m, \mathbf{f} \leftarrow \chi_{\mathsf{lwe}}^n, f \leftarrow \chi_{\mathsf{lwe}}$

6 **return** $\mathsf{ct} := (\mathbf{a}, b) := (\mathbf{r}^\intercal \mathbf{A} + \mathbf{f}, \mathbf{r}^\intercal \mathbf{b} + f + \lfloor \frac{Q}{2} \rfloor \cdot \mu)$

Note: PP outputs pp s.t. the noise in the ciphertext $e_{\mathsf{ct}} := b - \mathbf{a}^\intercal \mathbf{s} - \lfloor \frac{Q}{2} \rfloor \cdot \mu$ satisfies $|e_{\mathsf{ct}}| < B_{\mathsf{eval}}$ with overwhelming probability

$\underline{\mathsf{Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)}$:

7 **return** $\overline{\mathsf{ct}} := (\overline{\mathbf{a}}, \overline{b})$ s.t. $e_{\mathsf{eval}} := \overline{b} - \overline{\mathbf{a}}^\intercal \mathbf{s} - \lfloor \frac{Q}{2} \rfloor \cdot C(\mu_1, \ldots, \mu_l)$ satisfies $|e_{\mathsf{eval}}| < B_{\mathsf{eval}}$ for any C, where $\mu_1, \ldots, \mu_l$ are the plaintexts of $\mathsf{ct}_1, \ldots, \mathsf{ct}_l$

$\underline{\mathsf{Dec}(\mathsf{sk}, \mathsf{ct} := (\mathbf{a}, b)) \to \overline{\mu} \in \mathcal{M}}$:

8 **return** $\overline{\mu} := \lfloor (b - \mathbf{a}^\intercal \mathbf{s}) / \lfloor \frac{Q}{2} \rfloor \rceil$

4 Definitions of FHE and Threshold FHE

In this section, we define the syntax and the framework of (threshold) FHE.

4.1 Syntax and Structure of the Underlying FHE

We first define the syntax of FHE as follows:

Definition 4.1 (FHE). *A fully homomorphic encryption (FHE) scheme* $\mathsf{FHE} = (\mathsf{PP}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *for a plaintext space* $\mathcal{M}$ *and a ciphertext space* $\mathcal{C}$ *is defined as follows.*

- $\mathsf{FHE.PP}(1^\kappa) \to \mathsf{pp}$*: On input a security parameter* κ*, outputs a set of public parameters* pp*. The following algorithms implicitly take* pp *as an argument.*
- $\mathsf{FHE.KeyGen}(\mathsf{pp}) \to (\mathsf{evk}, \mathsf{pk}, \mathsf{sk})$*: Outputs a public evaluation key* evk*, a public key* pk *and a secret key* sk*.*
- $\mathsf{FHE.Enc}(\mathsf{pk}, \mu \in \mathcal{M}) \to \mathsf{ct} \in \mathcal{C}$, $\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{ct} \in \mathcal{C}) \to \overline{\mu} \in \mathcal{M}$*: Have the usual syntax for public-key encryption/decryption.*
- $\mathsf{FHE.Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l \in \mathcal{C}) \to \mathsf{ct}_{\mathsf{eval}} \in \mathcal{C}$*: On input the evaluation key* evk*, a function* $C : \mathcal{M}^l \to \mathcal{M}$*, and a set of* l *ciphertexts* $\mathsf{ct}_1, \ldots, \mathsf{ct}_l$ *of* $\mu_1, \ldots, \mu_l$*, outputs the result ciphertext* $\mathsf{ct}_{\mathsf{eval}}$*.* FHE *is* correct*, if* $\mathsf{Dec}(\mathsf{sk}, \overline{\mathsf{ct}}) = f(m_1, \ldots, m_l)$ *holds with overwhelming probability.*

Then, we specify in Algo. 1 a high-level structure for the underlying FHE scheme based on LWE, which is identical to that of [PS25]. This scheme is based on the LWE-based PKE of [LP11].

Algorithm 2: Module-LWE-based underlying FHE scheme $\mathsf{mFHE} := (\mathsf{PP}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$

$\underline{\mathsf{PP}(1^\kappa) \to \mathsf{pp}}$:

return $\mathsf{pp} := (n, k, m, Q, \chi_{\mathsf{mlwe}}, B_{\mathsf{eval}})$ ▷ χ_{mlwe}: distribution over $\mathbb{Z}$

$\underline{\mathsf{KeyGen}()}$:

$\mathsf{sk} := \mathbf{s} \leftarrow \mathcal{R}(\chi^n_{\mathsf{mlwe}})^k$, $\mathsf{pk} := (\mathbf{A}, \mathbf{b} := \mathbf{As} + \mathbf{e}) \sim \mathsf{MLWE}(k, m, Q, \mathcal{R}(\chi^n_{\mathsf{mlwe}}))$

Generate the evaluation key evk ▷ We omit the details of evk

return $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mu \in \mathcal{M} := \mathcal{R}_2)}$:

$\mathbf{r} \leftarrow \mathcal{R}(\chi^n_{\mathsf{mlwe}})^m, \mathbf{f} \leftarrow \mathcal{R}(\chi^n_{\mathsf{mlwe}})^k, f \leftarrow \mathcal{R}(\chi^n_{\mathsf{mlwe}})$

return $\mathsf{ct} := (\mathbf{a}, b) := (\mathbf{r}^\intercal \mathbf{A} + \mathbf{f}, \mathbf{r}^\intercal \mathbf{b} + f + \lfloor \frac{Q}{2} \rfloor \cdot \mu) \in \mathcal{R}_Q^{k+1}$

Note: PP outputs pp s.t. the noise in the ciphertext $e_{\mathsf{ct}} := b - \mathbf{a}^\intercal \mathbf{s} - \lfloor \frac{Q}{2} \rfloor \cdot \mu$ satisfies $\|e_{\mathsf{ct}}\|_\infty < B_{\mathsf{eval}}$ with overwhelming probability

$\underline{\mathsf{Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)}$:

return $\overline{\mathsf{ct}} := (\overline{\mathbf{a}}, \overline{b})$ s.t. $e_{\mathsf{eval}} := \overline{b} - \overline{\mathbf{a}}^\intercal \mathbf{s} - \lfloor \frac{Q}{2} \rfloor \cdot C(\mu_1, \ldots, \mu_l)$ satisfies $\|e_{\mathsf{eval}}\|_\infty < B_{\mathsf{eval}}$ for any C, where $\mu_1, \ldots, \mu_l$ are the plaintexts of $\mathsf{ct}_1, \ldots, \mathsf{ct}_l$

$\underline{\mathsf{Dec}(\mathsf{sk}, \mathsf{ct} := (\mathbf{a}, b)) \to \overline{\mu} \in \mathcal{M}}$:

return $\overline{\mu} := \lfloor (b - \mathbf{a}^\intercal \mathbf{s}) / \lfloor \frac{Q}{2} \rfloor \rceil$

We construct our ThFHE scheme on the basis of this underlying FHE scheme (Algo. 1). This framework captures most known FHE schemes, except for GSW-like [GSW13] schemes. Note that we can also use ring- or module- variants of this FHE scheme, because we do not require the "known-norm" variant of LWE for security in contrast to ThFHE construction of [PS25]. We also describe a variant of the FHE scheme based on module-LWE in Algo. 2. We construct our MLWE-based ThFHE in Sect. 7 upon this underlying MLWE-based FHE scheme.

4.2 Syntax of our Threshold FHE

Passelègue and Stehlé [PS25] generalized the definition of threshold functional encryption with the additional ServerDec algorithm. We modify the definition with our additional entities; seed values for masking $\{\mathsf{seed}_i\}_{i \in [N]}$, session ID sid and active party set $\mathsf{act} \subseteq [N]$. Hence, we formally describe the syntax of our threshold FHE as follows:

Definition 4.2 (Threshold FHE). *Let $P_1, \ldots, P_N$ be parties and $T(\leq N)$ be a threshold. Let $\mathcal{M}$ denote the plaintext space, $\mathcal{C}$ ciphertext space and $\mathcal{C}_{\mathsf{dec}}$ the processed ciphertext space. A threshold FHE scheme is a tuple of PPT algorithms* $\mathsf{ThFHE} = (\mathsf{PP}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{ServerDec}, \mathsf{PartDec}, \mathsf{FinDec})$ *as follows:*

- $\mathsf{ThFHE.PP}(1^\kappa, 1^N) \to \mathsf{pp}$*: On input a security parameter κ and a number of parties N, outputs a set of public parameters* pp*. The following algorithms implicitly take* pp *as argument.*

– $\mathsf{ThFHE.KeyGen}(T, N) \to (\mathsf{evk}, \mathsf{pk}, \mathsf{sk}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in[N]})$*: On input a threshold* T *and a number of parties* N*, outputs a public evaluation key* evk*, a public key* pk *and a secret key* sk*, and secret shares* $\{\mathsf{sk}_i\}_{i\in[N]}$*, as well as the seed value* $\{\mathsf{seed}_i\}_{i\in[N]}$*.*
– $\mathsf{ThFHE.Enc}(\mathsf{pk}, \mu \in \mathcal{M}) \to \mathsf{ct} \in \mathcal{C}$ *and* $\mathsf{ThFHE.Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \dots \mathsf{ct}_l) \to \mathsf{ct}_{\mathsf{eval}} \in \mathcal{C}$*: Same as* FHE.Enc *and* FHE.Eval *in Def. 4.1.*
– $\mathsf{ThFHE.ServerDec}(\mathsf{pk}, \mathsf{ct}, \mathsf{act}, \{\mathsf{seed}_i\}_{i\in\mathsf{act}}) \to (\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid})$*: It has an internal counter* sid *of session ID, which is initialized* $\mathsf{sid} \leftarrow 0$ *at setup. On input the public key* pk*, a ciphertext* $\mathsf{ct} \in \mathcal{C}$ *(or an evaluation result* $\mathsf{ct}_{\mathsf{eval}} \in \mathcal{C}$*), a set of active parties* act*, and the set of seed values* $\{\mathsf{seed}_i\}_{i\in\mathsf{act}}$*, outputs a processed ciphertext* $\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}$ *and session ID* sid*.*
– $\mathsf{ThFHE.PartDec}(\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}, \mathsf{act}, \mathsf{sk}_i, \mathsf{seed}_i) \to \mathsf{pd}_i$*: On input a ciphertext* $\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}$ *(outputted by* ServerDec*), a secret share* sk_i*,* seed_i*,* sid *and* act*, outputs a partial decryption share* pd_i *related to the party* P_i*.*
– $\mathsf{ThFHE.FinDec}(\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \{\mathsf{pd}_i\}_{i\in\mathsf{act}}) \to \overline{\mu} \in \mathcal{M}$*: On input a ciphertext* $\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}$ *and a set of partial decryption shares* $\{\mathsf{pd}_i\}_{i\in\mathsf{act}}$*, outputs a decryption result* $\overline{\mu} \in \mathcal{M}$*.*

5 Our Threshold FHE from Known-Norm LWE

In this section, we solve Question 1.2 (described in Sect. 1): we present a threshold FHE scheme that can be (efficiently) constructed with Shamir secret sharing.

We first describe our construction and threshold decryption procedure in Sect. 5.1. Next, we analyse the distribution of noise which can be seen to the parties (and the adversary) after the threshold decryption in Sect. 5.2. Then, relying on the noise analysis, we prove the correctness and security of our scheme in Sect. 5.3 and Sect. 5.4, respectively.

5.1 Construction and the Threshold Decryption Procedure

We present our construction of knLWE-based ThFHE in Algo. 3. The command $\mathbf{assert}\{x\}$ outputs $\perp$ if x is false and does nothing otherwise. Our two-round threshold decryption procedure is described as follows.

Note that, in Algo. 3, pp virtually includes the bound on the decryption query L_{Dec}, and ServerDec and PartDec assert $\mathsf{sid} \leq L_{\mathsf{Dec}}$ only for generality of the construction: This L_{Dec} is needed for the LWE-based ThFHE that we construct in next Sect. 6. We can now regard L_{Dec} is infinitely large ($= \infty$) since Algo. 3 does not require a priori limit on the number of decryption queries.

Setting. We assume the deterministic function $\mathsf{PRF} : \{0,1\}^* \to \mathbb{Z}_q$ is pseudorandom function. As in [PS25], we also assume the existence of party called Server who is semi-honest and uncorrupted. Additionally, we assume a trusted key generation process as in [BGG+18, Constr. 5.11].

Input/output. Let a ciphertext ct (possibly outputted by Eval) be an input of the threshold decryption process, which is given to the Server. If we apply

Algorithm 3: Our threshold FHE from known-norm LWE: $\mathsf{ThFHE} := (\mathsf{PP}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{ServerDec}, \mathsf{PartDec}, \mathsf{FinDec})$.

$\underline{\mathsf{PP}(1^\kappa, 1^N)}$:

return $\mathsf{pp} := (T, n, m, p, q, Q = p \cdot q, \chi_{\mathsf{lwe}}, \sigma_0, \sigma_1, \sigma_{\mathsf{flood}}, B_{\mathsf{eval}}, L_{\mathsf{Dec}})$

$\underline{\mathsf{KeyGen}(T, N)}$:

$(\mathsf{evk}, \mathsf{pk}_{\mathsf{FHE}}, \mathsf{sk} := \mathbf{s}) \leftarrow \mathsf{FHE.KeyGen}()$ ▷ $\mathbf{s} \leftarrow \chi_{\mathsf{lwe}}^n$

$\mathsf{pk} := (\mathsf{pk}_{\mathsf{FHE}}, \|\mathbf{s}\|)$ ▷ Include norm $\|\mathbf{s}\|$ in pk

$\{\mathsf{sk}_i := \mathbf{s}_i\}_{i\in[N]} \leftarrow \mathsf{Shamir}_{\mathbb{Z}_q^n, T, N}.\mathsf{Share}(\mathsf{sk})$

for $i \in [N]$ **do** $\mathsf{seed}_i \xleftarrow{\$} \{0,1\}^\kappa$

return $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in[N]})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mu \in \mathcal{M} := \{0,1\})}$:

return $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, \mu)$ (defined in Algo. 1)

$\underline{\mathsf{Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)}$:

return $\mathsf{ct}_{\mathsf{eval}} \leftarrow \mathsf{FHE.Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)$ (defined in Algo. 1)

$\underline{\mathsf{ServerDec}(\mathsf{pk}, \mathsf{ct} \in \mathcal{C}, \mathsf{act}, \{\mathsf{seed}_i\}_{i\in\mathsf{act}})}$:

$\mathsf{sid} \leftarrow \mathsf{sid} + 1$, **assert** $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$ ▷ Internal counter ($\mathsf{sid} \leftarrow 0$ at setup).

for $i \in \mathsf{act}$ **do** $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid}) \in \mathbb{Z}_q$

$\mathsf{ct}_{\mathsf{fresh}} \leftarrow \mathsf{Enc}(\mathsf{pk}, 0)$

$\mathsf{ct}_{\mathsf{in}} := (\mathsf{ct}_{\mathsf{in},0}, \mathsf{ct}_{\mathsf{in},1}) := \mathsf{ct} + \mathsf{ct}_{\mathsf{fresh}} \bmod Q$

$\mathsf{ct}_{\mathsf{dec},0} := \left\lfloor \frac{1}{p} \cdot \mathsf{ct}_{\mathsf{in},0} \right\rceil_{\sigma_0} \bmod q$, $\mathfrak{E} \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sigma_{\mathsf{flood}}}$ ▷ discrete Gaussian for $\mathfrak{E}$

$\mathsf{ct}_{\mathsf{dec},1} := \left\lfloor \frac{1}{p} \cdot (\mathsf{ct}_{\mathsf{in},1} + \mathfrak{E}) \right\rceil_{\sigma_1} + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q$

return $(\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid})$

$\underline{\mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}, \mathsf{act}, \mathsf{sk}_i, \mathsf{seed}_i)}$:

assert $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$, $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid}) \in \mathbb{Z}_q$ ▷ Private mask of P_i

return $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot \overline{\mathbf{a}}^\intercal \mathbf{s}_i + \mathfrak{m}_i \bmod q$

$\underline{\mathsf{FinDec}(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{act}, \{\mathsf{pd}_i\}_{i\in\mathsf{act}})}$:

assert $\{|\mathsf{act}| \ge T\}$

return $\overline{\mu} := \lfloor (\overline{b} - \sum_{i\in\mathsf{act}} \mathsf{pd}_i) / \lfloor \frac{q}{2} \rfloor \rceil$

the threshold decryption protocol to an MPC, each party P_i sends a ciphertext ct_i of its own input plaintext μ_i to the Server (as an additional round before the following first round). Then, the Server performs homomorphic evaluation of C to $\mathsf{ct}_1, \ldots, \mathsf{ct}_N$. The result ciphertext $\mathsf{ct}_{\mathsf{eval}}$ is used as an input of the threshold decryption protocol (i.e., an input of the ServerDec). As an output of threshold decryption, all parties except for the Server obtain the plaintext of $\mathsf{ct}_{\mathsf{eval}}$.

First round. On the input ciphertext ct of large modulus Q, the Server performs ServerDec to sanitize ct with noise flooding and round it to a ciphertext $\mathsf{ct}_{\mathsf{dec}}$ with a small modulus q as in [PS25]. In our scheme, the Server further adds

a sum of masks that corresponds to all parties in act, $\{\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})\}_{i\in\mathsf{act}}$ in $\mathsf{ServerDec}$. Then, Server broadcasts $\mathsf{ct}_{\mathsf{dec}}$ to all parties in act.

Second round. Each party P_i performs $\mathsf{PartDec}$ on $\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b})$ and generates the partial decryption share $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot \overline{\mathbf{a}}^{\intercal}\mathbf{s}_i + \mathfrak{m}_i$, where $\mathfrak{m}_i = \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$. P_i then broadcasts pd_i to all parties in act. Note that all parties perform $\mathsf{PartDec}$ only on the ciphertexts sent by the Server, and they never send partial decryption shares to the Server, under the non-collusion assumption between Server and parties as in [PS25]; the Server is assumed not to obtain any partial decryption shares. At the end of the second round, each party P_i performs FinDec to recover the plaintext, given all partial decryption shares $\{\mathsf{pd}_i\}_{i\in\mathsf{act}}$.

5.2 Noise Analysis

We next analyse the distribution of noise which can be seen to the parties (and the adversary) after the threshold decryption. The following lemma is used to prove both the correctness and security of our knLWE-based ThFHE scheme:

Lemma 5.1 (Noise analysis). *Let $\epsilon := \mathrm{negl}(\kappa)$, $p \in \mathbb{N}$, $B_{\mathsf{lwe}}, B_{\mathsf{eval}} \in \mathbb{R}$ and let χ_{lwe} be a B_{lwe}-bounded distribution s.t. $\mathbf{s} \leftarrow \chi_{\mathsf{lwe}}^n$ satisfies $\gcd(\mathbf{s}) = 1$ with overwhelming probability. Let $\sigma_{\mathsf{flood}} = \Omega(2^{\kappa} B_{\mathsf{eval}})$ and $\sigma_{\mathsf{flood}} \geq \frac{\sqrt{2}}{2} p\eta_{\epsilon}(\mathbb{Z})$, $\sigma_0 \geq \sqrt{2}B_{\mathsf{lwe}}\eta_{\epsilon}(\mathbb{Z})$, $\sigma_1 \geq \eta_{\epsilon}(\mathbb{Z})$. Let $\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1}) \in \mathcal{C}_{\mathsf{dec}}$ be the output of $\mathsf{ServerDec}$ in Algo. 3, and define $\mathfrak{e}$ as the ciphertext noise of $\mathsf{ct}_{\mathsf{dec}}$;*

$$\mathfrak{e} := \mathsf{ct}_{\mathsf{dec},1} - \mathsf{ct}_{\mathsf{dec},0}^{\intercal}\mathbf{s} - \lfloor \tfrac{q}{2} \rfloor \cdot \mu - \textstyle\sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q. \tag{13}$$

Then, we have

$$\mathfrak{e} \approx_{\mathsf{s}} \lfloor \tfrac{1}{p} e_0 + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}, \ \text{where } e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0\|\mathbf{s}\|)^2 + \sigma_{\mathsf{flood}}^2}}. \tag{14}$$

Proof. We first analyse $\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1})$ of Algo. 3. Let the rounding error of $\mathsf{ct}_{\mathsf{dec},0} := \left\lfloor \frac{1}{p} \cdot \mathsf{ct}_{\mathsf{in},0} \right\rceil_{\sigma_0}$ be denoted as

$$\mathbf{r} := \mathsf{ct}_{\mathsf{in},0} - p \cdot \mathsf{ct}_{\mathsf{dec},0} \sim D_{p\{\frac{1}{p}\mathsf{ct}_{\mathsf{in},0}\} + p\mathbb{Z}^n, p\sigma_0}, \tag{15}$$

where $\{x\} := x - \lfloor x \rceil$ is the fractional part of $x \in \mathbb{R}$. Let us denote

$$\mathsf{ct}_{\mathsf{in},1} := \mathsf{ct}_{\mathsf{in},0}^{\intercal}\mathbf{s} + \lfloor \tfrac{Q}{2} \rfloor \cdot \mu + e_{\mathsf{eval}} + e_{\mathsf{fresh}}, \tag{16}$$

where e_{eval} and e_{fresh} are the decryption noises of input ciphertext ct and $\mathsf{ct}_{\mathsf{fresh}}$. Then, by definition of $\mathsf{ct}_{\mathsf{dec},1}$, (15) and (16), we have

$$\begin{aligned}\mathsf{ct}_{\mathsf{dec},1} &:= \left\lfloor \tfrac{1}{p} \cdot (\mathsf{ct}_{\mathsf{in},1} + \mathfrak{E}) \right\rceil_{\sigma_1} + \textstyle\sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q \\ &= \mathsf{ct}_{\mathsf{dec},0}^{\intercal}\mathbf{s} + \left\lfloor \tfrac{1}{p} \cdot \left(\mathbf{r}^{\intercal}\mathbf{s} + \lfloor \tfrac{Q}{2} \rfloor \cdot \mu + e_{\mathsf{eval}} + e_{\mathsf{fresh}} + \mathfrak{E} \right) \right\rceil_{\sigma_1} + \textstyle\sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q.\end{aligned}$$

Thus, we obtain

$$\mathfrak{e} = \lfloor \tfrac{1}{p}(\mathbf{r}^\intercal \mathbf{s} - \{\tfrac{Q}{2}\} \cdot \mu + e_{\mathsf{eval}} + e_{\mathsf{fresh}} + \mathfrak{E}) + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}, \tag{17}$$

where we used fact that $\lfloor \frac{Q}{2} \rfloor = p\lfloor \frac{q}{2} \rfloor + p\{\frac{q}{2}\} - \{\frac{Q}{2}\}$.

We first simplify (17) by the smudging lemma. Recall that we have $|e_{\mathsf{eval}}|$, $|e_{\mathsf{fresh}}| < B_{\mathsf{eval}}$ (as described in Algo. 1), and $\mathfrak{E} \leftarrow D_{\frac{1}{2}\mathbb{Z},\sigma_{\mathsf{flood}}}$. Also note that $\{\frac{Q}{2}\} \cdot \mu \in \{0, \frac{1}{2}\} \subset \frac{1}{2}\mathbb{Z}$ (this is why we need to sample $\mathfrak{E}$ over $\frac{1}{2}\mathbb{Z}$). Thus, by the hypothesis of $\sigma_{\mathsf{flood}} = \Omega(2^\kappa B_{\mathsf{eval}})$ and the smudging lemma (Lem. 3.12), we obtain

$$-\{\tfrac{Q}{2}\} \cdot \mu + e_{\mathsf{eval}} + e_{\mathsf{fresh}} + \mathfrak{E} \approx_{\mathsf{s}} \mathfrak{E},$$

and thus, by (17), we obtain

$$\mathfrak{e} \approx_{\mathsf{s}} \lfloor \tfrac{1}{p}(\mathbf{r}^\intercal \mathbf{s} + \mathfrak{E}) + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}. \tag{18}$$

Next, we analyze the distribution of $\mathbf{r}^\intercal \mathbf{s} + \mathfrak{E}$. Recall (15), and let $\mathbf{c} := p\{\frac{1}{p}\mathsf{ct}_{\mathsf{in},0}\}$, then $\mathbf{c} \in p \cdot (\frac{1}{p}\mathbb{Z}^n) = \mathbb{Z}^n$ is an integer vector. Let us denote the i-th elements of $\mathbf{c}$ and $\mathbf{r}$ by c_i and r_i; then we have $r_i \sim D_{c_i + p\mathbb{Z}, p\sigma_0}$ for $i \in [n]$. By the hypothesis, we have $p \cdot \sigma_0 \geq \sqrt{2} B_{\mathsf{lwe}} \eta_\epsilon(p\mathbb{Z}) \geq \sqrt{2}\|\mathbf{s}\|_\infty \eta_\epsilon(p\mathbb{Z})$. Hence, by Lem. 3.7, we obtain $\mathbf{r}^\intercal \mathbf{s} \approx_{\mathsf{s}} D_{\mathbf{c}^\intercal \mathbf{s} + p\mathbb{Z}, p\sigma_0 \|\mathbf{s}\|}$. Note that $\mathbf{c}^\intercal \mathbf{s} + p\mathbb{Z} \subseteq \mathbb{Z} (\subset \frac{1}{2}\mathbb{Z})$ since $\mathbf{c}^\intercal \mathbf{s} \in \mathbb{Z}$. In addition, by the hypothesis $\sigma_{\mathsf{flood}} \geq \sqrt{2} p \eta_\epsilon(\frac{1}{2}\mathbb{Z})$ (and $\sigma_0 \geq \sqrt{2}\eta_\epsilon(\mathbb{Z})$), we have $((\sigma_0\|\mathbf{s}\|)^{-2} + (\sigma_{\mathsf{flood}}/p)^{-2})^{-1/2} \geq \frac{1}{\sqrt{2}} \min(\sigma_0\|\mathbf{s}\|, \sigma_{\mathsf{flood}}/p) \geq \eta_\epsilon(\frac{1}{2}\mathbb{Z})$. Thus, by Cor. 3.9, we have

$$e_0 := \mathbf{r}^\intercal \mathbf{s} + \mathfrak{E} \approx_{\mathsf{s}} D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0\|\mathbf{s}\|)^2 + \sigma_{\mathsf{flood}}^2}}.$$

Now, we obtain (14) by (18) and the above. □

Solely discrete analysis. As a minor contribution, the above analysis supports a discrete Gaussian for the flooding noise $\mathfrak{E} \leftarrow D_{\frac{1}{2}\mathbb{Z},\sigma_{\mathsf{flood}}}$, whereas [PS25] requires a continuous Gaussian. Thus, our scheme is constructed solely with discrete values, which is desirable for the implementation. Approximating real values with floating point values incurs rounding errors, which requires a careful analysis of security.

5.3 Correctness

We define the correctness of our knLWE-based ThFHE scheme (Algo. 3).

Definition 5.2 (Correctness). *We define* $\mathsf{Game}^{\mathsf{correct}}_{\mathsf{ThFHE}}$ *in Algo. 4.* ThFHE *(Algo. 3) is correct if, for any* $\kappa, N > 0, T \in [1, N]$, $l > 0$, $C : \mathcal{M}^l \to \mathcal{M}$, $(\mu_1, \ldots, \mu_l) \in M^l$, $\mathsf{act} \subseteq [N]$ *s.t.* $|\mathsf{act}| \geq T$, $L_{\mathsf{Dec}} > 0$ *and* $\mathsf{sid} \leq L_{\mathsf{Dec}}$, *we have:*

$$\Pr[\mathsf{Game}^{\mathsf{correct}}_{\mathsf{ThFHE}}(1^\kappa, T, N, \mathsf{sid}, \mathsf{act}, l, C, (\mu_1, \ldots, \mu_l) = 1] \geq 1 - \mathrm{negl}(\kappa).$$

We then show the correctness of our ThFHE scheme as follows:

Algorithm 4: Correctness game for our ThFHE

$\underline{\mathsf{Game}^{\mathsf{correct}}_{\mathsf{ThFHE}}(1^\kappa, T, N, \mathsf{sid}, \mathsf{act}, l, C, (\mu_1, \ldots, \mu_l) \in \mathcal{M}^l)}$:
1 **assert** $\{$ $\mathsf{act} \subseteq [N]$ and $|\mathsf{act}| \geq T$ and $\mathsf{sid} \leq L_{\mathsf{Dec}}\}$
2 $\mathsf{pp} \leftarrow \mathsf{PP}(1^\kappa, 1^N)$, $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in[N]}) \leftarrow \mathsf{KeyGen}(T, N)$
3 **for** $j \in [l]$ **do** $\mathsf{ct}_j \leftarrow \mathsf{Enc}(\mathsf{pk}, \mu_j)$
4 $\mathsf{ct} \leftarrow \mathsf{Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)$
5 $(\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}) \leftarrow \mathsf{ServerDec}(\mathsf{pk}, \mathsf{ct}, \mathsf{act}, \{\mathsf{seed}_i\}_{i\in\mathsf{act}})$
6 **for** $i \in \mathsf{act}$ **do** $\mathsf{pd}_i \leftarrow \mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}, \mathsf{act}, \mathsf{sk}_i, \mathsf{seed}_i)$
7 $\overline{\mu} \leftarrow \mathsf{FinDec}(\mathsf{ct}_{\mathsf{dec}}, \{\mathsf{pd}_i\}_{i\in\mathsf{act}}, \mathsf{act})$
8 **if** $\overline{\mu} = C(\mu_1, \ldots, \mu_l)$ **then return** 1 **else return** 0

Theorem 5.3. *Assume that parameters $\epsilon, p, \chi_{\mathsf{lwe}}, B_{\mathsf{lwe}}, B_{\mathsf{pub}}, \sigma_{\mathsf{flood}}, \sigma_0, \sigma_1$ are selected as in Lem. 5.1. Furthermore, let $\overline{\sigma_{\mathsf{dec}}} := \sqrt{n(\sigma_0 B_{\mathsf{lwe}})^2 + (\sigma_{\mathsf{flood}}/p)^2} + \sigma_1$ and $q = \overline{\sigma_{\mathsf{dec}}} \cdot \Omega(\sqrt{\kappa})$. Then, ThFHE (Algo. 3) is correct if the underlying FHE (Algo. 1) is correct (Def. 4.1).*

Proof. Let $\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1}) \in \mathcal{C}_{\mathsf{dec}}$ be the output of ServerDec in Algo. 3. Then, by Lem. 5.1, we obtain

$$\mathsf{ct}_{\mathsf{dec},1} \approx_{\mathsf{s}} \mathsf{ct}_{\mathsf{dec},0}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \tfrac{q}{2} \rfloor \cdot \mu + \textstyle\sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q, \tag{19}$$

where $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$ and $e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0\|\mathbf{s}\|)^2 + \sigma_{\mathsf{flood}}^2}}$. Let $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot \mathsf{ct}_{\mathsf{dec},0}^{\intercal}\mathbf{s}_i + \mathfrak{m}_i \bmod q$ for $i \in \mathsf{act}$. Then, by the correctness of Shamir secret sharing (Constr. 3.17), we have

$$\textstyle\sum_{i\in\mathsf{act}} \mathsf{pd}_i := \mathsf{ct}_{\mathsf{dec},0}^{\intercal}\mathbf{s} + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q. \tag{20}$$

Finally, by (19) and (20), we have

$$\mathsf{FinDec}(\overline{\mathsf{ct}}, \{\mathsf{pd}_i\}_{i\in\mathsf{act}}, \mathsf{act}) = \lfloor (\mathsf{ct}_{\mathsf{dec},1} - \textstyle\sum_{i\in\mathsf{act}} \mathsf{pd}_i)/\lfloor \frac{q}{2} \rfloor \rceil \approx_{\mathsf{s}} \mu + \lfloor \mathfrak{e}/\lfloor \frac{q}{2} \rfloor \rceil.$$

We now only need to analyse the bound of $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$. We rewrite $\mathfrak{e}$ as follows;

$$\mathfrak{e} := \tfrac{1}{p} e_0 + \{\tfrac{q}{2}\} \cdot \mu + e_1,$$

where we defined $e_1 \leftarrow \mathcal{D}_{\mathbb{Z} - \frac{1}{p}e_0 - \{\frac{q}{2}\}\cdot\mu, \sigma_1}$. By Lem. 3.10, we have $\frac{1}{p}e_0 \sim D_{\frac{1}{2p}\mathbb{Z}, \sqrt{(\sigma_0\|\mathbf{s}\|)^2 + (\sigma_{\mathsf{flood}}/p)^2}}$. Since χ_{lwe} is B_{lwe}-bounded, we have $\|\mathbf{s}\| < \sqrt{n} B_{\mathsf{lwe}}$ (with overwhelming probability). Hence, let $\sigma' := \sqrt{n(\sigma_0 B_{\mathsf{lwe}})^2 + (\sigma_{\mathsf{flood}}/p)^2}$, then $\frac{1}{p}e_0$ is $\sigma' \cdot \Omega(\sqrt{\kappa})$-bounded by Lem. 3.11. Similarly, e_1 is $\sigma_1 \Omega(\sqrt{\kappa})$-bounded. Also note that $|\{\frac{q}{2}\} \cdot \mu| \leq \frac{1}{2} (= O(1))$. Thus, we have $|\mathfrak{e}|$ is $(\sigma_1 + \sigma') \cdot \Omega(\sqrt{\kappa})$-bounded and $(\sigma_1 + \sigma') = \overline{\sigma_{\mathsf{dec}}}$. Hence, by selecting $q = \overline{\sigma_{\mathsf{dec}}} \cdot \Omega(\sqrt{\kappa})$, we have $\lfloor \mathfrak{e}/\lfloor \frac{q}{2} \rfloor \rceil = 0$ with overwhelming probability; i.e., correctness holds. □

Algorithm 5: The real experiment $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$ and the ideal experiment $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$. Differences between $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$ and $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ are highlighted.

$\underline{\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}(1^\kappa)}$:

1 $\mathsf{pp} \leftarrow \mathsf{PP}(1^\kappa, 1^N)$, $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in[N]}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, T, N)$

2 $(\mathsf{corr} \subseteq [N] \text{ s.t. } |\mathsf{corr}| < T, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{pp}, \mathsf{pk})$

3 $\mathsf{hon} := [N] \backslash \mathsf{corr}$, $\mathsf{ctr}_{\mathsf{Enc}} \leftarrow 0, \mathsf{ctr}_{\mathsf{Dec}} \leftarrow 0, \mathsf{sid} \leftarrow 0, \mathsf{List} \leftarrow \emptyset$

4 **return** $\{0,1\} \leftarrow \mathcal{A}_2^{\mathsf{OEnc},\mathsf{OEval},\mathsf{ODec}}(\{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in\mathsf{corr}}, \mathsf{st})$

$\underline{\mathsf{OEnc}(\mu)}$:

5 $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mu)$, $\mathsf{ctr}_{\mathsf{Enc}} \leftarrow \mathsf{ctr}_{\mathsf{Enc}} + 1$, $\mathsf{List}[\mathsf{ctr}] \leftarrow (\mu, \mathsf{ct})$

$\underline{\mathsf{OEval}(C, (i_1, \ldots, i_l))}$:

6 **assert** $\{(i_1, \ldots, i_l) \subseteq [\mathsf{ctr}_{\mathsf{Enc}}]\}$, **for** $j \in [l]$ **do** $(\mu_j, \mathsf{ct}_j) \leftarrow \mathsf{List}[i_j]$

7 $\mu \leftarrow C(\mu_1, \ldots, \mu_l)$, $\mathsf{ct} \leftarrow \mathsf{Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)$

8 $\mathsf{ctr}_{\mathsf{Enc}} \leftarrow \mathsf{ctr}_{\mathsf{Enc}} + 1$, $\mathsf{List}[\mathsf{ctr}_{\mathsf{Enc}}] \leftarrow (\mu, \mathsf{ct})$

$\underline{\mathsf{ODec}(\mathsf{act})}$:

9 **assert** $\{|\mathsf{act}| \geq T$ and $\mathsf{ctr}_{\mathsf{Dec}} < \mathsf{ctr}_{\mathsf{Enc}}$ and $\mathsf{ctr}_{\mathsf{Dec}} < L_{\mathsf{Dec}}\}$

10 $\mathsf{ctr}_{\mathsf{Dec}} \leftarrow \mathsf{ctr}_{\mathsf{Dec}} + 1$, $(\mu, \mathsf{ct}) \leftarrow \mathsf{List}[\mathsf{ctr}_{\mathsf{Dec}}]$, $\mathsf{corr}_{\mathsf{act}} := \mathsf{corr} \cap \mathsf{act}$, $\mathsf{hon}_{\mathsf{act}} := \mathsf{hon} \cap \mathsf{act}$

11 $(\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}) \leftarrow \mathsf{ServerDec}(\mathsf{pk}, \mathsf{ct}, \mathsf{act}, \{\mathsf{seed}_i\}_{i\in\mathsf{act}})$

12 $\{\mathsf{pd}_i\}_{i\in\mathsf{act}} \leftarrow \{\mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}, \mathsf{act}, \mathsf{sk}_i, \mathsf{seed}_i)\}_{i\in\mathsf{act}}$

13 **return** $(\mathsf{ct}_{\mathsf{dec}}, \{\mathsf{pd}_i\}_{i\in\mathsf{act}})$

$\underline{\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}(1^\kappa)}$:

Same as $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}(1^\kappa)$ except for that $\{\mathsf{sk}_i\}_{i\in[N]}$ is replaced with:

14 $\{\mathsf{sk}_i\}_{i\in[N]} \leftarrow \mathsf{Shamir.Share}_{\mathbb{Z}_q,T,N}((\mathbf{0},\mathbf{0}))$ ▷ Shares with no information

▷ OEnc and OEval are identical to those in $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$.

$\underline{\mathsf{ODec}(\mathsf{act})}$:

▷ Perform lines 9–10 of $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$.ODec(act) in Algo. 5

15 **return** $(\mathsf{ct}_{\mathsf{dec}}, \{\mathsf{pd}_i\}_{i\in\mathsf{act}}) \leftarrow \mathsf{Sim}(\mathsf{ct}, \mu, \mathsf{pk}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in\mathsf{corr}}, \mathsf{act})$

5.4 Security

We define the security of our ThFHE scheme in Def. 5.4. Informally, the definition implies that the view of the real world (:= $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$), where the adversary receives an honestly generated set of partial decryption shares

$$\{\mathsf{pd}_i\}_{i\in\mathsf{act}} \leftarrow \{\mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}} \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}, \mathsf{act}, \mathsf{sk}_i, \mathsf{seed}_i)\}_{i\in\mathsf{act}}$$

is (computationally) indistinguishable from the view of the ideal world (:= $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$), where the adversary receives simulated partial decryption shares

$$\{\mathsf{pd}_i\}_{i\in\mathsf{act}} \leftarrow \mathsf{Sim}(\overline{\mathsf{ct}}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in\mathsf{corr}})$$

generated by some PPT simulator Sim by using only incomplete secret share sets $\{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in\mathsf{corr}}$ ($|\mathsf{corr}| < t$) that are given to the adversary. Hence, owing to the privacy of secret sharing (Constr. 3.17), the secret share $\{\mathsf{sk}_i\}_{i\in\mathsf{corr}}$ provide

no information regarding sk to the adversary; they are simulated with the shares of meaningless zero vectors $\{\mathsf{sk}_i\}_{i\in[N]} \leftarrow \mathsf{Shamir.Share}(\mathbf{0})$.

Definition 5.4 (Security). ThFHE *(Algo. 3) is simulation-secure if, for any PPT algorithms $\mathcal{D}$ and $\mathcal{A}$, there exists a PPT simulator* Sim *s.t.* $\mathrm{Adv}^{\mathsf{ThFHE}}_{\mathcal{D},\mathcal{A}}(\kappa) := |\Pr[\mathcal{D}(\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}(1^\kappa)) = 1] - \Pr[\mathcal{D}(\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}(1^\kappa)) = 1]| = \mathrm{negl}(\kappa)$*, where* $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}(1^\kappa)$ *and* $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}(1^\kappa)$ *are defined in Algo. 5.*

Then, we prove the above security in Thm. 5.5.

Theorem 5.5. ThFHE *(Algo. 3) is simulation-secure under the assumption on the pseudorandomness of* PRF *and the LWE assumption: Formally, for any PPT algorithms $\mathcal{D}$ and $\mathcal{A}$, there exists PPT algorithms* Sim*, $\mathcal{B}_{\mathsf{LWE}}$ and $\mathcal{B}_{\mathsf{PRF}}$ s.t.*

$$\mathrm{Adv}^{\mathsf{ThFHE}}_{\mathcal{D},\mathcal{A}}(\kappa) < \mathrm{Adv}^{\mathsf{LWE}(m,n,Q,\chi_{\mathsf{lwe}})}_{\mathcal{B}_{\mathsf{LWE}}} + \mathrm{Adv}^{\mathsf{PRF}}_{\mathcal{B}_{\mathsf{PRF}}}(\kappa) + \mathrm{negl}(\kappa),$$

if $\epsilon, p, q, Q, \chi_{\mathsf{lwe}}, B_{\mathsf{lwe}}, B_{\mathsf{pub}}, \sigma_{\mathsf{flood}}, \sigma_0, \sigma_1$ are selected as in Thm. 5.3.

Proof. We provide the full formal proof in Appendix B. Here, we provide an overview of the proof. We define hybrid experiments $\mathsf{Hyb}_1, \ldots, \mathsf{Hyb}_6$ to show that $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}} \approx_{\mathsf{c}} \mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ holds for some PPT simulator Sim.

In Hyb_1, we replace the output $\mathsf{ct}_{\mathsf{dec},1}$ of ServerDec with $\mathsf{ct}_{\mathsf{dec},1} = \mathsf{ct}^{\intercal}_{\mathsf{dec},0}\mathbf{s} + \mathfrak{e} + \lfloor \frac{q}{2} \rfloor \mu + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q$, where $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\}\mu \rceil_{\sigma_1}$ and $e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z},\sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma^2_{\mathsf{flood}}}}$ (as in (14)). We have $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}} \approx_{\mathsf{s}} \mathsf{Hyb}_1$ by Lem. 5.1.

In Hyb_2, $\mathsf{ct}_{\mathsf{dec},0} := \lfloor \frac{1}{p}\mathsf{ct}_{\mathsf{in},0} \rceil_{\sigma_0}$ is replaced with $\mathsf{ct}_{\mathsf{dec},0} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$. We have $\mathsf{Hyb}_1 \approx_{\mathsf{c}} \mathsf{Hyb}_2$ by LWE assumption. Now, the output of the ServerDec is

$$\mathsf{ct}_{\mathsf{dec}} := (\bar{\mathbf{a}} \leftarrow \mathcal{U}(\mathbb{Z}_q^n), \bar{b} := \bar{\mathbf{a}}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \tfrac{q}{2} \rfloor \cdot \mu + \textstyle\sum_{i\in\mathsf{act}} \mathfrak{m}_i). \tag{21}$$

In Hyb_3, for $i \in \mathsf{hon}_{\mathsf{act}}$, we replace $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$ with $\mathfrak{m}_i \leftarrow \mathcal{U}(\mathbb{Z}_q)$. We have $\mathsf{Hyb}_2 \approx_{\mathsf{c}} \mathsf{Hyb}_3$ by the assumption on the pseudorandomness of PRF.

In Hyb_4, we fix some $h \in \mathsf{hon}_{\mathsf{act}}$ and replace the decryption share of P_h with

$$\mathsf{pd}_h := \bar{b} - \lfloor \tfrac{q}{2} \rfloor \cdot \mu - \textstyle\sum_{i\in\mathsf{act}\setminus\{h\}} \mathsf{pd}_i - \mathfrak{e}. \tag{22}$$

Since we have $\sum_{i\in\mathsf{act}} \mathsf{pd}_i = \bar{b} - \lfloor \frac{q}{2} \rfloor \cdot \mu - \mathfrak{e}$ in both Hyb_3 and Hyb_4 (by (19) and (20)), we have $\mathsf{Hyb}_3 \approx \mathsf{Hyb}_4$ (=identically distributed).

In Hyb_5, $\mathfrak{m}_h$ is no longer sampled, and $\bar{b}$ is directly sampled uniformly random. In the previous Hyb_4, $\mathfrak{m}_h \leftarrow \mathcal{U}(\mathbb{Z}_q)$ is only used to calculate $\bar{b}$ in (21); thus, $\bar{b} \sim \mathcal{U}(\mathbb{Z}_q)$ and $\mathsf{Hyb}_4 \approx \mathsf{Hyb}_5$ hold. Similarly, in Hyb_6, for $i \in \mathsf{hon}_{\mathsf{act}}\setminus\{h\}$, $\mathfrak{m}_i$ is no longer sampled and pd_i is directly sampled uniformly random. We have $\mathsf{Hyb}_5 \approx \mathsf{Hyb}_6$ since $\mathfrak{m}_i \sim \mathcal{U}(\mathbb{Z}_q)$ is only used to calculate pd_i in Hyb_5.

Let Sim be the (PPT) procedure of Hyb_6 for outputting ($\mathsf{ct}_{\mathsf{dec}} := (\bar{\mathbf{a}}, \bar{b})$, $\{\mathsf{pd}_i\}_{i\in\mathsf{act}}$). Sim only uses the corrupted secret shares $\{\mathsf{sk}_i, \mathsf{seed}_i\}_{i\in\mathsf{corr}}$ (and other public values), and $e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z},\sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma^2_{\mathsf{flood}}}}$ is simulatable since pk contains $\|\mathbf{s}\|$ by construction (Algo. 3). Hence, by the privacy of Shamir secret sharing, $\{\mathsf{sk}_i\}_{i\in\mathsf{corr}} \leftarrow \mathsf{Shamir.Share}(\mathsf{sk})$ and $\{\mathsf{sk}_i\}_{i\in\mathsf{corr}} \leftarrow \mathsf{Shamir.Share}(\mathbf{0})$ are identically distributed. Thus, we obtain $\mathsf{Hyb}_6 \approx \mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$. □

The need for semantic security of the underlying FHE. Note that we do not rely on the known-norm LWE assumption for this simulation security, but rely on the standard LWE assumption. Interestingly, we do not require the semantic security of the underlying FHE for our security proof (Thm. 5.5); we use the standard LWE assumption (with ephemeral secret vector $\mathbf{r}$ for encryption) only for sanitizing ciphertexts in the $\mathsf{ServerDec}$ procedure. Intuitively, this is because the partial decryption shares are masked with pseudorandom outputs of the PRF. Our security definition (Def. 5.4) keeps the encryption oracle Enc unchanged in the ideal experiment, and we only show that partial decryption shares do not reveal any information to the adversary. The semantic security of underlying FHE, which can be shown from known-norm LWE, becomes necessary for applications of our threshold decryption procedure. For example, in MPC, we can use the semantic security of the underlying FHE to prove that no information about the plaintext input of the ciphertext of each party P_i is provided to Server (and other parties).

Security against the untrusted (semi-honest) Server. Def. 5.4 defines security against an adversary who corrupts parties. Although the Server is assumed not to collude with any parties, it is untrusted (semi-honest). We can prove the security against the Server more easily than in Thm. 5.5. Although the Server receives seed_i for PRF of all parties act, outputs of PRF ($=\mathsf{m}_i$) are used only for calculating partial decryption shares. Since Server does not receive any partial decryption shares under the non-collusion assumtion, the knowledge of seed_i is meaningless to Server. Therefore, Server sees only input FHE ciphertexts ct (and other public entities), the semantic security of which follows from that of the underlying FHE scheme.

6 Our Threshold FHE from (Standard) LWE

In this section, we present a threshold FHE scheme, which can be (directly) constructed from (standard) LWE without using the known-norm variant of LWE as in [PS25] (and Sect. 5). This threshold FHE scheme is extended to the one from Module-LWE in Sect. 7, which solves Question 1.1.

We first describe our construction in Sect. 6.1. Next, we analyse the distribution of noise which can be seen to the parties (and the adversary) after the threshold decryption in Sect. 6.2. Then, we prove the correctness and security of our scheme in Sect. 6.3 and Sect. 6.4, respectively.

6.1 Construction

We construct our (standard-)LWE-based ThFHE in Algo. 6 based on our knLWE-based ThFHE presented in the previous section (Algo. 3). The only difference between Algo. 6 and Algo. 3 is the additional entity called "padding noise" ζ.

In Algo. 6, we first sample padding noise $\zeta^{(\mathsf{sid})}$ in line 3 for $\mathsf{sid} \in L_{\mathsf{Dec}}$, where L_{Dec} is the bound on the threshold decryption query: Since we need a fresh

Algorithm 6: Our threshold FHE from (standard) LWE:
$\mathsf{ThFHE} := (\mathsf{PP}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{ServerDec}, \mathsf{PartDec}, \mathsf{FinDec})$.

$\underline{\mathsf{PP}(1^\kappa, 1^N)}$:
return $\mathsf{pp} := (T, n, m, p, q, Q = p \cdot q, \chi_{\mathsf{lwe}}, \sigma_0, \sigma_1, \sigma_{\mathsf{flood}}, B_{\mathsf{pub}}, B_{\mathsf{eval}}, L_{\mathsf{Dec}})$

$\underline{\mathsf{KeyGen}(T, N)}$:
$(\mathsf{evk}, \mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.KeyGen}()$ ▷ $\mathsf{sk} := \mathbf{s} \leftarrow \chi_{\mathsf{lwe}}^n$
$\mathsf{err} := \boldsymbol{\zeta} := (\zeta^{(1)}, \ldots, \zeta^{(L_{\mathsf{Dec}})}) \leftarrow D^{L_{\mathsf{Dec}}}_{\mathbb{Z}, \sigma_0 \sqrt{B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2}}$ ▷ "Padding noise"
$\{\mathsf{sk}_i := (\mathbf{s}_i \in \mathbb{Z}_q^n, \boldsymbol{\zeta}_i \in \mathbb{Z}_q^{L_{\mathsf{Dec}}})\}_{i \in [N]} \leftarrow \mathsf{Shamir}_{\mathbb{Z}_q, T, N}.\mathsf{Share}((\mathsf{sk}, \mathsf{err}))$
for $i \in [N]$ **do** $\mathsf{seed}_i \xleftarrow{\$} \{0,1\}^\kappa$
return $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i \in [N]})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mu \in \mathcal{M} := \{0,1\})}$:
return $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, \mu)$ (defined in Algo. 1)

▷ Eval, ServerDec and FinDec are identical to those of Algo. 3

$\underline{\mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}, \mathsf{act}, \mathsf{sk}_i := (\mathbf{s}_i, \boldsymbol{\zeta}_i), \mathsf{seed}_i)}$:
assert $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$, $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid}) \in \mathbb{Z}_q$
return $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot (\overline{\mathbf{a}}^\intercal \mathbf{s}_i + \zeta_i^{(\mathsf{sid})}) + \mathfrak{m}_i \bmod q$ ▷ Add padding noise share $\zeta_i^{(\mathsf{sid})}$

padding noise $\zeta^{(\mathsf{sid})}$ for each threshold decryption session, we need to sample padding noise as many as the number of queries on the threshold decryption at the setup. Shamir shares $(\zeta_1^{(\mathsf{sid})}, \ldots, \zeta_N^{(\mathsf{sid})})$ of each padding noise $\zeta^{(\mathsf{sid})}$ are distributed for each party as with the shares of secret keys. The threshold decryption shares of Algo. 6 includes $\zeta_i^{(\mathsf{sid})}$ as $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot (\overline{\mathbf{a}}^\intercal \mathbf{s}_i + \zeta_i^{(\mathsf{sid})}) + \mathfrak{m}_i$. The threshold decryption procedure is identical to that described in Sect. 5.1.

6.2 Noise Analysis

We next analyse the distribution of noise which can be seen to the parties (and the adversary). The only difference from Lem. 5.1 is that we now have an additional term of padding noise ζ:

Lemma 6.1 (Noise analysis). *Assume that parameters $\epsilon, p, \chi_{\mathsf{lwe}}, B_{\mathsf{lwe}}, \sigma_{\mathsf{flood}}, \sigma_0, \sigma_1$ are selected as in Lem. 5.1, and define $\mathfrak{e}$ as in (13). Furthermore, let $B_{\mathsf{pub}} \ge \sqrt{nB_{\mathsf{lwe}}^2 + 1}$ and $\zeta \sim D_{\mathbb{Z}, \sigma_0 \sqrt{B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2}}$, then we have*

$$\mathfrak{e} + \zeta \approx_{\mathsf{s}} \lfloor \tfrac{1}{p} e_0' + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}, \text{ where } e_0' \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0 B_{\mathsf{pub}})^2 + \sigma_{\mathsf{flood}}^2}}. \tag{23}$$

Proof. By Lem. 5.1, we have

$$\mathfrak{e} \approx_{\mathsf{s}} \lfloor \tfrac{1}{p} e_0 + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}, \text{ where } e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0 \|\mathbf{s}\|)^2 + \sigma_{\mathsf{flood}}^2}}$$

as in (14). Thus, we have $\mathfrak{e} + \zeta \approx_{\mathsf{s}} \lfloor \frac{1}{p}(e_0 + p \cdot \zeta) + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$. By Lem. 3.10, we have $p \cdot \zeta \sim D_{p\mathbb{Z}, p\sigma_0 \sqrt{B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2}}$. By the hypothesis $\sigma_{\mathsf{flood}} \ge \sqrt{2} p \eta_\epsilon(\frac{1}{2}\mathbb{Z})$ and

$\sigma_0 \geq \sqrt{2}\eta_\epsilon(\mathbb{Z})$, we have

$$\begin{aligned}&(((p\sigma_0\|\mathbf{s}\|)^2 + \sigma_{\mathsf{flood}}^2)^{-1} + (p^2\sigma_0^2(B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2))^{-1})^{-1/2}\\ &\geq \tfrac{p}{\sqrt{2}}\min(\sigma_0\|\mathbf{s}\|, \sigma_0\sqrt{B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2}, \sigma_{\mathsf{flood}}/p) \geq \tfrac{p}{\sqrt{2}}\min(\sigma_0, \sigma_{\mathsf{flood}}/p) \geq \eta_\epsilon(p\mathbb{Z}),\end{aligned}$$

where we use the fact that $\sqrt{B_{\mathsf{pub}}^2 - \|\mathbf{s}\|^2} \geq \sqrt{B_{\mathsf{pub}}^2 - nB_{\mathsf{lwe}}^2} \geq 1$. Thus, by Cor. 3.9, we have $e_0' := e_0 + p \cdot \zeta \approx_{\mathsf{s}} D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0 B_{\mathsf{pub}})^2 + \sigma_{\mathsf{flood}}^2}}$. □

6.3 Correctness

The correctness of our LWE-based ThFHE scheme is identically defined as in Def. 5.2. We prove the correctness as follows:

Theorem 6.2 (Correctness). *Assume that parameters ϵ, p, χ_{lwe}, B_{lwe}, B_{pub}, σ_{flood}, σ_0, and σ_1 are selected as in Lem. 6.1. Furthermore, let $\sigma_{\mathsf{dec}} := \sqrt{(\sigma_0 B_{\mathsf{pub}})^2 + (\sigma_{\mathsf{flood}}/p)^2} + \sigma_1$ and $q = \sigma_{\mathsf{dec}} \cdot \Omega(\sqrt{\kappa})$. Then, ThFHE (Algo. 6) is correct (Def. 5.2) if the underlying FHE (Algo. 1) is correct (Def. 4.1).*

Proof. The proof is identical to the proof of Thm. 5.3 until (19) is reached. After that, we now have $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot (\mathsf{ct}_{\mathsf{dec},0}^{\mathsf{T}}\mathbf{s}_i + \zeta_i^{(\mathsf{sid})}) + \mathfrak{m}_i \bmod q$ for $i \in \mathsf{act}$. By the correctness of Shamir secret sharing, we have

$$\textstyle\sum_{i\in\mathsf{act}} \mathsf{pd}_i := \mathsf{ct}_{\mathsf{dec},0}^{\mathsf{T}}\mathbf{s} + \zeta^{(\mathsf{sid})} + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q. \tag{24}$$

By (19) and (24), we have

$$\mathsf{FinDec}(\overline{\mathsf{ct}}, \{\mathsf{pd}_i\}_{i\in\mathsf{act}}, \mathsf{act}) = \lfloor(\mathsf{ct}_{\mathsf{dec},1} - \textstyle\sum_{i\in\mathsf{act}} \mathsf{pd}_i)/\lfloor\frac{q}{2}\rfloor\rceil \approx_{\mathsf{s}} \mu + \lfloor(\mathfrak{e} + \zeta^{(\mathsf{sid})})/\lfloor\frac{q}{2}\rfloor\rceil.$$

Hence, we only need to analyse the bound of $\mathfrak{e} + \zeta^{(\mathsf{sid})}$. By Lem. 6.1, we have

$$\mathfrak{e} + \zeta^{(\mathsf{sid})} \approx_{\mathsf{s}} \lfloor\tfrac{1}{p}e_0' + \{\tfrac{q}{2}\} \cdot \mu\rceil_{\sigma_1}, \text{ where } e_0' \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0 B_{\mathsf{pub}})^2 + \sigma_{\mathsf{flood}}^2}}.$$

We rewrite the above as follows;

$$\mathfrak{e} + \zeta^{(\mathsf{sid})} \approx_{\mathsf{s}} \tfrac{1}{p}e_0' + \{\tfrac{q}{2}\} \cdot \mu + e_1,$$

where we defined $e_1 \leftarrow \mathcal{D}_{\mathbb{Z} - \frac{1}{p}e_0' - \{\frac{q}{2}\}\cdot\mu, \sigma_1}$. By Lem. 3.10, we have $\frac{1}{p}e_0' \sim D_{\frac{1}{2p}\mathbb{Z}, \sqrt{(\sigma_0 B_{\mathsf{pub}})^2 + (\sigma_{\mathsf{flood}}/p)^2}}$. Hence, let $\sigma' := \sqrt{(\sigma_0 B_{\mathsf{pub}})^2 + (\sigma_{\mathsf{flood}}/p)^2}$, then $\frac{1}{p}e_0'$ is $\sigma' \cdot \Omega(\sqrt{\kappa})$-bounded by Lem. 3.11. Similarly, e_1 is $\sigma_1\Omega(\sqrt{\kappa})$-bounded. Also note that $|\{\frac{q}{2}\} \cdot \mu| \leq \frac{1}{2} (= O(1))$. Thus, we have $|\mathfrak{e} + \zeta^{(\mathsf{sid})}|$ is $(\sigma_1 + \sigma') \cdot \Omega(\sqrt{\kappa})$-bounded and $(\sigma_1 + \sigma') = \overline{\sigma_{\mathsf{dec}}}$. Hence, by selecting $q = \overline{\sigma_{\mathsf{dec}}} \cdot \Omega(\sqrt{\kappa})$, we have $\lfloor\mathfrak{e} + \zeta^{(\mathsf{sid})}/\lfloor\frac{q}{2}\rfloor\rceil = 0$ with overwhelming probability; i.e., correctness holds. □

6.4 Security

The security of our LWE-based ThFHE (Algo. 6) is also defined as in Def. 5.4, since the syntax is identical to that of knLWE-based ThFHE (Algo. 3). We now prove the security:

Theorem 6.3 (Security). ThFHE *(Algo. 6) is simulation-secure (Def. 5.4) under the assumption on the pseudorandomness of* PRF *and the LWE assumption: Formally, for any PPT algorithms* $\mathcal{D}$ *and* $\mathcal{A}$*, there exists PPT algorithms* Sim*,* $\mathcal{B}_{\mathsf{LWE}}$ *and* $\mathcal{B}_{\mathsf{PRF}}$ *s.t.*

$$\mathrm{Adv}^{\mathsf{ThFHE}}_{\mathcal{D},\mathcal{A}}(\kappa) < \mathrm{Adv}^{\mathsf{LWE}(n,m,Q,\chi_{\mathsf{lwe}})}_{\mathcal{B}_{\mathsf{LWE}}} + \mathrm{Adv}^{\mathsf{PRF}}_{\mathcal{B}_{\mathsf{PRF}}}(\kappa) + \mathrm{negl}(\kappa),$$

if $\epsilon, p, q, Q, \chi_{\mathsf{lwe}}, B_{\mathsf{lwe}}, B_{\mathsf{pub}}, \sigma_{\mathsf{flood}}, \sigma_0, \sigma_1$ *are selected as in Thm. 6.2.*

Proof. The proof is almost identical to Thm. 5.5, except for the simulation of the noise revealed after decryption. Let Hyb'_6 be a variant of Hyb_6 in the proof of Thm. 5.5, where pd_h in (22) is replaced with

$$\mathsf{pd}_h := \overline{b} - \lfloor \tfrac{q}{2} \rfloor \cdot \mu - \textstyle\sum_{i \in \mathsf{act} \setminus \{h\}} \mathsf{pd}_i - \mathfrak{e} + \zeta^{(\mathsf{sid})},$$

where $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$, $e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0\|\mathbf{s}\|)^2 + \sigma^2_{\mathsf{flood}}}}$ as in (14) and $\zeta^{(\mathsf{sid})} \leftarrow D_{\mathbb{Z}, \sigma_0 \sqrt{B^2_{\mathsf{pub}} - \|\mathbf{s}\|^2}}$. In a manner that is similar to the proof of Thm. 5.5, we obtain $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}} \approx_{\mathsf{c}} \mathsf{Hyb}'_6$. Furthermore, by Lem. 6.1, we obtain $-\mathfrak{e} + \zeta^{(\mathsf{sid})} \approx_{\mathsf{s}} \lfloor \frac{1}{p} e'_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$, where $e'_0 \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0 B_{\mathsf{pub}})^2 + \sigma^2_{\mathsf{flood}}}}$ as in (23). Thus, the challenger can simulate $-\mathfrak{e} + \zeta^{(\mathsf{sid})}$ (without knowledge of $\|\mathbf{s}\|$, as in Thm. 5.5), since it consists solely of public values. Let the PPT simulator Sim of $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ (in Algo. 5) be the procedures of Hyb'_6, then $\mathsf{Hyb}'_6 \approx \mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ holds by the privacy of Shamir secret sharing (Constr. 3.17). □

7 Our Threshold FHE from Module-LWE

Our threshold FHE scheme (Algo. 6) in the previous section is constructed from standard LWE, without relying on known-norm LWE as in [PS25] (and Sect. 5). Thus, Algo. 6 can be naturally extended to the construction based on module-LWE. However, the distribution of the noise is nontrivially different in the MLWE setting. The padding noise is changed from the univariate Gaussian $D_{\mathbb{Z}, \sigma_0 \sqrt{B^2_{\mathsf{pub}} - \|\mathbf{s}\|^2}}$ in the LWE setting (Algo. 6) to the multivariate Gaussian $D_{\mathbb{Z}^n, \sigma_0 \sqrt{B^2_{\mathsf{pub}} \mathbf{I} - \mathbf{\Sigma}_{\mathbf{s}}}}$ in the MLWE setting (Algo. 7), where $\mathbf{\Sigma}_{\mathbf{s}}$ is a covariance matrix that corresponds to the secret vector $\mathbf{s} \in \mathcal{R}^k$.

We first describe our construction in Sect. 7.1. Next, we provide the noise analysis in the MLWE setting in Sect. 7.2. Then, we prove the correctness and security of mThFHE in Sect. 7.3 and Sect. 7.4, respectively.

Algorithm 7: Our threshold FHE from module-LWE: $\mathsf{mThFHE} := (\mathsf{PP}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{ServerDec}, \mathsf{PartDec}, \mathsf{FinDec})$.

$\underline{\mathsf{PP}(1^\kappa, 1^N)}$:

return $\mathsf{pp} := (T, n, k, m, p, q, Q = p \cdot q, \chi_{\mathsf{mlwe}}, \sigma_0, \sigma_1, \sigma_{\mathsf{flood}}, B_{\mathsf{pub}}, B_{\mathsf{eval}}, L_{\mathsf{Dec}})$

$\underline{\mathsf{KeyGen}(T, N)}$:

Let $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{mFHE.KeyGen}()$, and denote $\mathsf{sk} = \mathbf{s} = (s_1, \ldots, s_k)^\intercal$.

$\mathsf{err} := \boldsymbol{\zeta} := (\zeta^{(1)}, \ldots, \zeta^{(L_{\mathsf{Dec}})}) \leftarrow (\mathcal{R}(D_{\mathbb{Z}^n, \sigma_0 \sqrt{B_{\mathsf{pub}}^2 \mathbf{I} - \boldsymbol{\Sigma}_{\mathbf{s}}}}))^{L_{\mathsf{Dec}}}$, where $\boldsymbol{\Sigma}_{\mathbf{s}} := \sum_{i \in [k]} \mathsf{Gram}(s_i)$.

$\{\mathsf{sk}_i := (\mathbf{s}_i \in \mathcal{R}_q^k, \boldsymbol{\zeta}_i \in \mathcal{R}_q^{L_{\mathsf{Dec}}})\}_{i \in [N]} \leftarrow \mathsf{Shamir}_{\mathcal{R}_q, T, N}.\mathsf{Share}((\mathsf{sk}, \mathsf{err}))$

for $i \in [N]$ **do** $\mathsf{seed}_i \xleftarrow{\$} \{0,1\}^\kappa$

return $(\mathsf{evk}, \mathsf{pk}, \mathsf{sk}, \{\mathsf{sk}_i, \mathsf{seed}_i\}_{i \in [N]})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mu \in \mathcal{M} := \mathcal{R}_2)}$:

return $\mathsf{ct} \leftarrow \mathsf{mFHE.Enc}(\mathsf{pk}, \mu)$ (defined in Algo. 2)

$\underline{\mathsf{Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)}$:

return $\mathsf{ct}_{\mathsf{eval}} \leftarrow \mathsf{mFHE.Eval}(\mathsf{evk}, C, \mathsf{ct}_1, \ldots, \mathsf{ct}_l)$ (defined in Algo. 2)

$\underline{\mathsf{ServerDec}(\mathsf{pk}, \mathsf{ct} \in \mathcal{C}, \mathsf{act}, \{\mathsf{seed}_i\}_{i \in \mathsf{act}})}$:

$\mathsf{sid} \leftarrow \mathsf{sid} + 1$, **assert** $\{\mathsf{sid} \leq L_{\mathsf{Dec}}\}$ ▷ Internal counter ($\mathsf{sid} \leftarrow 0$ at setup).

for $i \in \mathsf{act}$ **do** $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid}) \in \mathcal{R}_q$

$\mathsf{ct}_{\mathsf{fresh}} \leftarrow \mathsf{Enc}(\mathsf{pk}, 0)$

$\mathsf{ct}_{\mathsf{in}} := (\mathsf{ct}_{\mathsf{in},0}, \mathsf{ct}_{\mathsf{in},1}) := \mathsf{ct} + \mathsf{ct}_{\mathsf{fresh}} \in \mathcal{R}_Q^{k+1}$

$\mathsf{ct}_{\mathsf{dec},0} := \left\lfloor \frac{1}{p} \cdot \mathsf{ct}_{\mathsf{in},0} \right\rceil_{\sigma_0} \in \mathcal{R}_q^k$, $\mathfrak{E} \leftarrow \mathcal{R}(D_{\frac{1}{2}\mathbb{Z}^n, \sigma_{\mathsf{flood}}})$

$\mathsf{ct}_{\mathsf{dec},1} := \left\lfloor \frac{1}{p} \cdot (\mathsf{ct}_{\mathsf{in},1} + \mathfrak{E}) \right\rceil_{\sigma_1} + \sum_{i \in \mathsf{act}} \mathfrak{m}_i \in \mathcal{R}_q$

return $(\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid})$

$\underline{\mathsf{PartDec}(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid}, \mathsf{act}, \mathsf{sk}_i := (\mathbf{s}_i, \boldsymbol{\zeta}_i), \mathsf{seed}_i)}$:

assert $\{\mathsf{sid} \leq L_{\mathsf{Dec}}\}$, $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid}) \in \mathcal{R}_q$

return $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot (\overline{\mathbf{a}}^\intercal \mathbf{s}_i + \zeta_i^{(\mathsf{sid})}) + \mathfrak{m}_i \in \mathcal{R}_q$

$\underline{\mathsf{FinDec}(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{act}, \{\mathsf{pd}_i\}_{i \in \mathsf{act}})}$:

assert $\{|\mathsf{act}| \geq T\}$

return $\overline{\mu} := \lfloor (\overline{b} - \sum_{i \in \mathsf{act}} \mathsf{pd}_i) / \lfloor \frac{q}{2} \rfloor \rceil$

7.1 Construction

The construction is almost identical to our LWE-based scheme. We describe our ThFHE from MLWE in Algo. 7. The threshold decryption procedure is identical to that described in Sect. 5.1.

7.2 Noise Analysis

We provide the counterpart of Lem. 6.1 in the MLWE setting, which is required for the proof of correctness and security of our MLWE-based ThFHE.

Lemma 7.1 (Noise analysis). *Let χ_{mlwe} be a B_{mlwe}-bounded distribution over $\mathbb{Z}$ s.t. $\mathbf{s} := (s_1, \ldots, s_k) \leftarrow \mathcal{R}(\chi_{\mathsf{mlwe}}^n)^k$ satisfies $\min_{i\in[k]} \sigma_{\min}(\mathbf{S}_i) \geq \frac{1}{c}$ for some $c > 0$, where $\mathbf{S}_i := \mathsf{mat}(s_i)$ (Def. 3.1). We also define $\mathbf{\Sigma}_{\mathbf{s}} := \sum_{i\in[k]} \mathbf{\Sigma}_{s_i}$, where $\mathbf{\Sigma}_{s_i} := \mathsf{Gram}(s_i)$. Let $\epsilon := \mathrm{negl}(\kappa)$, $p \in \mathbb{N}$, $\sigma_{\mathsf{flood}} = \Omega(2^\kappa B_{\mathsf{eval}})$, $\sigma_{\mathsf{flood}} \geq p\sqrt{n/2}B_{\mathsf{mlwe}}\eta_\epsilon^+(\mathbb{Z}^n)$, $\sigma_0 \geq c\sqrt{n/2}B_{\mathsf{mlwe}}\eta_\epsilon^+(\mathbb{Z}^n)$, $\sigma_1 \geq \eta_\epsilon(\mathbb{Z}^n)$. Let $\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1}) \in \mathcal{C}_{\mathsf{dec}}$ be the output of* ServerDec *in Algo. 7, and define $\mathfrak{e}$ as the ciphertext noise of $\mathsf{ct}_{\mathsf{dec}}$;*

$$\mathfrak{e} := \mathsf{ct}_{\mathsf{dec},1} - \mathsf{ct}_{\mathsf{dec},0}^\intercal \mathbf{s} - \lfloor \tfrac{q}{2} \rfloor \cdot \mu - \textstyle\sum_{i\in\mathsf{act}} \mathfrak{m}_i \in \mathcal{R}_q.$$

Then, we have

$$\mathfrak{e} \approx_{\mathsf{s}} \lfloor \tfrac{1}{p} e_0 + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}, \text{ where } e_0 \leftarrow \mathcal{R}(D_{\frac{1}{2}\mathbb{Z}^n, \sqrt{p^2\sigma_0^2\mathbf{\Sigma}_{\mathbf{s}} + \sigma_{\mathsf{flood}}^2\mathbf{I}}}). \tag{25}$$

Furthermore, for $B_{\mathsf{pub}} > \sqrt{kn^2B_{\mathsf{mlwe}}^2 + 1}$ and $\zeta \leftarrow \mathcal{R}(D_{\mathbb{Z}^n, \sigma_0\sqrt{B_{\mathsf{pub}}^2\mathbf{I} - \mathbf{\Sigma}_{\mathbf{s}}}})$, we have

$$\mathfrak{e} + \zeta \approx_{\mathsf{s}} \lfloor \tfrac{1}{p} e_0' + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}, \text{ where } e_0' \leftarrow \mathcal{R}(D_{\frac{1}{2}\mathbb{Z}^n, \sqrt{p^2\sigma_0^2B_{\mathsf{pub}}^2 + \sigma_{\mathsf{flood}}^2}}). \tag{26}$$

Proof. We first analyse $\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1})$. Let the rounding error of $\mathsf{ct}_{\mathsf{dec},0} := \lfloor \frac{1}{p} \cdot \mathsf{ct}_{\mathsf{in},0} \rceil_{\sigma_0}$ be denoted as

$$\mathbf{r} := \mathsf{ct}_{\mathsf{in},0} - p \cdot \mathsf{ct}_{\mathsf{dec},0} \in \mathcal{R}^k. \tag{27}$$

In a manner similar to Lem. 5.1, we obtain

$$\mathfrak{e} = \lfloor \tfrac{1}{p}(\mathbf{r}^\intercal\mathbf{s} - \{\tfrac{Q}{2}\} \cdot \mu + e_{\mathsf{eval}} + e_{\mathsf{fresh}} + \mathfrak{E}) + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}, \tag{28}$$

where e_{eval} and e_{fresh} are the ciphertext noises of input ciphertext ct and $\mathsf{ct}_{\mathsf{fresh}}$. We first simplify (28). By applying the smudging lemma (Lem. 3.12) coefficient-wise and under the hypothesis that $\sigma_{\mathsf{flood}} = \Omega(2^\kappa B_{\mathsf{eval}})$, we obtain

$$-\{\tfrac{Q}{2}\} \cdot \mu + e_{\mathsf{eval}} + e_{\mathsf{fresh}} + \mathfrak{E} \approx_{\mathsf{s}} \mathfrak{E}.$$

Thus, by (28) and the above, we obtain

$$\mathfrak{e} \approx_{\mathsf{s}} \lfloor \tfrac{1}{p}(\mathbf{r}^\intercal\mathbf{s} + \mathfrak{E}) + \{\tfrac{q}{2}\} \cdot \mu \rceil_{\sigma_1}. \tag{29}$$

Next, we analyze the distribution of $\mathbf{r}^\intercal\mathbf{s} + \mathfrak{E}$. For any $\mathbf{x} \in \mathcal{R}^k$, we denote the i-th elements of $\mathbf{x}$ by $x_i \in \mathcal{R}$, and j-th coefficient of x_i by $x_{i,j} \in \mathbb{Z}$. To simplify notation, we let $\mathbf{u} := \mathsf{ct}_{\mathsf{in},0} \in \mathcal{R}_q^k$. Since $\lfloor \cdot \rceil_{\sigma_0}$ is performed element- and coefficient-wise, we have $r_{i,j} = u_{i,j} - p \cdot \lfloor \frac{1}{p} u_{i,j} \rceil_{\sigma_0}$ (recall (27) and $\mathsf{ct}_{\mathsf{dec},0} := \lfloor \frac{1}{p} \cdot \mathsf{ct}_{\mathsf{in},0} \rceil_{\sigma_0}$). Let $c_{i,j} := p\{\frac{1}{p}u_{i,j}\}$, and note that $c_{i,j} \in p \cdot (\frac{1}{p}\mathbb{Z}) = \mathbb{Z}$ is an integer. Then, we have $r_{i,j} \sim D_{c_{i,j}+p\mathbb{Z}, p\sigma_0}$ and $r_i = \mathcal{R}(D_{\mathbf{c}_i + p\mathbb{Z}^n, p\sigma_0})$, where $\mathbf{c}_i := (c_{i,1}, \ldots, c_{i,n}) \in \mathbb{Z}^n$. Thus, we have $r_i s_i \sim \mathcal{R}(\mathbf{S}_i \cdot D_{\mathbf{c}_i + p\mathbb{Z}^n, p\sigma_0})$ by Fact 3.2. Furthermore, for any $i \in [k]$, we have

$$r_i s_i \sim \mathcal{R}(D_{A_i, p\sigma_0\mathbf{S}_i}), \text{ where } A_i := \mathbf{S}_i \cdot \mathbf{c}_i + p\mathbf{S}_i \cdot \mathbb{Z}^n,$$

by Lem. 3.10 and the fact that $\mathsf{mat}(a)$ is nonsingular for any $a \neq 0 \in \mathcal{R}$. Note that the coset A_i satisfies $A_i \subseteq \mathbb{Z}^n$ since the elements of $\mathbf{S}_i$ and $\mathbf{c}_i$ are integers.

We now analyse the distribution of $\mathbf{r}^\intercal \mathbf{s} + \mathfrak{E} = \sum_{i=1}^n r_i s_i + \mathfrak{E} \in \mathcal{R}$, where $\mathfrak{E} \leftarrow \mathcal{R}(D_{\frac{1}{2}\mathbb{Z}^n, \sigma_{\mathsf{flood}}})$. Let $\widehat{\sigma} := \min(\sigma_{\mathsf{flood}}/p, \sigma_0/c)$, then we have $\widehat{\sigma} \geq \sqrt{2n} B_{\mathsf{mlwe}} \eta_\epsilon^+(\frac{1}{2}\mathbb{Z}^n)$ by the hypothesis. Hence, we have

$$\begin{aligned}\min(\sigma_{\mathsf{flood}}, \min_{i\in[k]} \sigma_{\min}(p\sigma_0 \mathbf{S}_i)) = p\widehat{\sigma} &\geq \sqrt{2n} p B_{\mathsf{mlwe}} \eta_\epsilon^+(\tfrac{1}{2}\mathbb{Z}^n) \\ &\geq \sqrt{2} \max_{i\in[k]} \|p\mathbf{S}_i\|_{\mathrm{len}} \eta_\epsilon^+(\tfrac{1}{2}\mathbb{Z}^n),\end{aligned}$$

since $\sqrt{n} B_{\mathsf{mlwe}} \geq \max_{i\in[k]} \|\mathbf{S}_i\|_{\mathrm{len}} = \max_{i\in[k]} \|\mathsf{vec}(s_i)\|$ holds. Note that $\|\mathsf{vec}(a)\| = \|\mathsf{mat}(a)\|_{\mathrm{len}}$ holds for any $a \in \mathcal{R}$. Therefore, by the subsequent Lem. 7.3, we obtain

$$e_0 := \mathbf{r}^\intercal \mathbf{s} + \mathfrak{E} \approx_{\mathsf{s}} \mathcal{R}(D_{\frac{1}{2}\mathbb{Z}^n, \sqrt{\mathbf{\Sigma}}}), \text{ where } \mathbf{\Sigma} := p^2\sigma_0^2 \mathbf{\Sigma}_{\mathbf{s}} + \sigma_{\mathsf{flood}}^2 \mathbf{I}.$$

Thus, by (29) and the above, we obtain (25).

Next, we prove (26). To simplify the notation, let $\mathbf{\Sigma}_\zeta := B_{\mathsf{pub}}^2 \mathbf{I} - \mathbf{\Sigma}_{\mathbf{s}}$, then we have $\zeta \sim \mathcal{R}(D_{\mathbb{Z}^n, \sigma_0\sqrt{\mathbf{\Sigma}_\zeta}})$. By Lem. 3.10, we also have $p \cdot \zeta \sim \mathcal{R}(D_{p\mathbb{Z}^n, p\sigma_0\sqrt{\mathbf{\Sigma}_\zeta}})$. By Cor. 3.9, $e_0' := e_0 + p \cdot \zeta \approx_{\mathsf{s}} \mathcal{R}(D_{\frac{1}{2}\mathbb{Z}^n, \sqrt{\mathbf{\Sigma} + p^2\sigma_0^2 \mathbf{\Sigma}_\zeta}}) = \mathcal{R}(D_{\frac{1}{2}\mathbb{Z}^n, \sqrt{p^2\sigma_0^2 B_{\mathsf{pub}}^2 + \sigma_{\mathsf{flood}}^2}})$ holds if we have $\sqrt{\mathbf{\Sigma}} \geq \eta_\epsilon(\frac{1}{2}\mathbb{Z}^n)$ and $(\mathbf{\Sigma}^{-1} + (p^2\sigma_0^2 \mathbf{\Sigma}_\zeta)^{-1})^{-1/2} \geq \eta_\epsilon(p\mathbb{Z}^n)$. By Fact 7.2 and the hypothesis, we have

$$\sigma_{\min}(\sqrt{\mathbf{\Sigma}}) = \sigma_{\min}(\sqrt{p^2\sigma_0^2 \mathbf{\Sigma}_{\mathbf{s}} + \sigma_{\mathsf{flood}}^2 \mathbf{I}}) \geq \sqrt{2} p\widehat{\sigma} \geq \sqrt{n} B_{\mathsf{mlwe}} \eta_\epsilon^+(p\mathbb{Z}^n) \tag{30}$$

Thus, $\sqrt{\mathbf{\Sigma}} \geq \eta_\epsilon(p\mathbb{Z}^n) > \eta_\epsilon(\frac{1}{2}\mathbb{Z}^n)$ holds by Fact 3.6. Again by Fact 7.2, we have

$$\sigma_{\min}((\mathbf{\Sigma}^{-1} + (p^2\sigma_0^2 \mathbf{\Sigma}_\zeta)^{-1})^{-1/2}) \geq \tfrac{1}{\sqrt{2}} \min(\sigma_{\min}(\sqrt{\mathbf{\Sigma}}), p\sigma_0 \sigma_{\min}(\sqrt{\mathbf{\Sigma}_\zeta})).$$

Hence, by (30) and the above, we now need only $p\sigma_0\sigma_{\min}(\sqrt{\mathbf{\Sigma}_\zeta}) \geq \sqrt{2}\eta_\epsilon(p\mathbb{Z}^n)$, i.e., $\sigma_0\sigma_{\min}(\sqrt{\mathbf{\Sigma}_\zeta}) \geq \sqrt{2}\eta_\epsilon(\mathbb{Z}^n)$. Since $\sigma_0 \geq \sqrt{2}\eta_\epsilon(\mathbb{Z}^n)$ by the hypothesis, we need only $\sigma_{\min}(\sqrt{\mathbf{\Sigma}_\zeta}) \geq 1$. To prove this, it is sufficient to show $\mathbf{\Sigma}_\zeta - \mathbf{I} = (B_{\mathsf{pub}}^2 - 1)\mathbf{I} - \sum_{i\in[k]} \mathbf{S}_i \mathbf{S}_i^\intercal \succ 0$. Note that, the absolute value of every elements of $\mathbf{S}_i \mathbf{S}_i^\intercal$ is $\leq n\|\mathsf{vec}(s_i)\|^2 \leq n^2 B_{\mathsf{mlwe}}^2$. Hence, $\mathbf{\Sigma}_\zeta - \mathbf{I}$ is strictly diagonally dominant since $(B_{\mathsf{pub}}^2 - 1) - kn^2 B_{\mathsf{mlwe}}^2 > 0$ by the hypothesis. Furthermore, this implies that $\mathbf{\Sigma}_\zeta - \mathbf{I} \succ 0$ (c.f., [HJ85, Thm. 6.1.10]). Thus, we have $\sigma_{\min}(\sqrt{\mathbf{\Sigma}_\zeta}) = \sqrt{\lambda_{\min}((\mathbf{\Sigma}_\zeta - \mathbf{I}) + \mathbf{I})} \geq \sqrt{\lambda_{\min}(\mathbf{\Sigma}_\zeta - \mathbf{I}) + 1} \geq 1$. □

We complete the proof by describing the deferred Fact 7.2 and Lem. 7.3:

Fact 7.2. *For any $\mathbf{\Sigma}_1, \mathbf{\Sigma}_2 \succ 0$, we have*

$$\begin{aligned}\sigma_{\max}(\sqrt{\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2}) &\leq \sqrt{\sigma_{\max}^2(\sqrt{\mathbf{\Sigma}_1}) + \sigma_{\max}^2(\sqrt{\mathbf{\Sigma}_2})} \\ \sigma_{\min}(\sqrt{\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2}) &\geq \sqrt{2} \min\{\sigma_{\min}(\sqrt{\mathbf{\Sigma}_1}), \sigma_{\min}(\sqrt{\mathbf{\Sigma}_2})\} \\ \sigma_{\min}(\sqrt{(\mathbf{\Sigma}_1^{-1} + \mathbf{\Sigma}_2^{-1})^{-1}}) &\geq \tfrac{1}{\sqrt{2}} \min\{\sigma_{\min}(\sqrt{\mathbf{\Sigma}_1}), \sigma_{\min}(\sqrt{\mathbf{\Sigma}_2})\}.\end{aligned}$$

Lemma 7.3 (Generalized from [OT25, Lemma 5.4]). *Let $\epsilon = \mathrm{negl}(\kappa)$. Let $\mathbf{\Sigma}_0, \dots, \mathbf{\Sigma}_k \in \mathbb{R}^{n\times n}$ be positive definite matrices. Let $A_0, \dots, A_k$ be cosets of full-rank lattices $\mathcal{L}_0(\mathbf{B}_0), \dots, \mathcal{L}_k(\mathbf{B}_k)$ with (nonsingular) basis $\mathbf{B}_0, \dots, \mathbf{B}_k$ s.t. $A_1, \dots, A_k \subseteq A_0$. Let $\widehat{\sigma} := \min_{i\in\{0,1,\dots,k\}} \sigma_{\min}(\sqrt{\mathbf{\Sigma}_i})$ and $\widehat{B} := \max_{i\in[k]} \|\mathbf{B}_i\|_{\mathrm{len}}$. If $\widehat{\sigma} \geq \sqrt{2}\widehat{B}\|\mathbf{B}_0\|_{\mathrm{len}}\eta^+_\epsilon(\mathbb{Z}^n)$, we have*

$$\sum_{i=1}^{k} D_{A_i,\sqrt{\mathbf{\Sigma}_i}} + D_{A_0,\sqrt{\mathbf{\Sigma}_0}} \approx_{\mathsf{s}} D_{A_0,\sqrt{\sum_{i=0}^{k}\mathbf{\Sigma}_i}}. \tag{31}$$

Proof. The proof is almost identical to that of [OT25, Lemma 5.4]. For completeness, we provide the proof in Appendix C. □

7.3 Correctness

Since the syntax is identical, the correctness of our MLWE-based ThFHE (Algo. 7) is also defined as in Def. 5.2. We prove the correctness:

Theorem 7.4 (Correctness). *Assume that parameters $\epsilon, p, \chi_{\mathsf{mlwe}}, B_{\mathsf{mlwe}}, B_{\mathsf{pub}}, \sigma_{\mathsf{flood}}, \sigma_0, \sigma_1$ are selected as in Lem. 7.1. Furthermore, assume $q = \sigma_{\mathsf{dec}} \cdot \Omega(\sqrt{\kappa})$, where $\sigma_{\mathsf{dec}} := \sqrt{(\sigma_0 B_{\mathsf{pub}})^2 + (\sigma_{\mathsf{flood}}/p)^2} + \sigma_1$. Then, mThFHE (Algo. 7) is correct if the underlying mFHE (Algo. 2) is correct.*

Proof. The proof is essentially identical to Thm. 6.2, except for that we rely on Lem. 7.1 instead of Lem. 6.1. □

7.4 Security

The security of our mThFHE (Algo. 7) is almost identically defined as Def. 5.4, by replacing variables over $\mathbb{Z}_q^n$ with the corresponding variables over R_q^k. The security can be proven by MLWE assumption as follows:

Theorem 7.5 (Security). mThFHE *(Algo. 7) is simulation-secure under the assumption on the pseudorandomness of* PRF *and* MLWE *assumption: For any PPT algorithms $\mathcal{D}, \mathcal{A}$, there exists PPT algorithms* Sim, $\mathcal{B}_{\mathsf{LWE}}$ *and* $\mathcal{B}_{\mathsf{PRF}}$ *s.t.*

$$\mathrm{Adv}^{\mathsf{ThFHE}}_{\mathcal{D},\mathcal{A}}(\kappa) < \mathrm{Adv}^{\mathsf{MLWE}(m,k,Q,\mathcal{R}(\chi^n_{\mathsf{mlwe}}))}_{\mathcal{B}_{\mathsf{LWE}}} + \mathrm{Adv}^{\mathsf{PRF}}_{\mathcal{B}_{\mathsf{PRF}}}(\kappa) + \mathrm{negl}(\kappa),$$

if $\epsilon, p, q, Q, B_{\mathsf{mlwe}}, B_{\mathsf{pub}}, \chi_{\mathsf{mlwe}}, \sigma_{\mathsf{flood}}, \sigma_0, \sigma_1$ are selected as in Thm. 7.4.

Proof. The proof is almost identical to that of Thm. 5.5, except for that we rely on Lem. 7.1 instead of Lem. 6.1. □

Acknowledgement. This work was supported by JST K Program Grant Number JPMJKP24U2, Japan.

References

[ABV+12] S. Agrawal, X. Boyen, V. Vaikuntanathan, P. Voulgaris, H. Wee. "Functional Encryption for Threshold Functions (or Fuzzy IBE) from Lattices". *PKC 2012*. 2012, pp. 280–297. https://doi.org/10.1007/978-3-642-30057-8_17.

[ARS+15] M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, M. Zohner. "Ciphers for MPC and FHE". *EUROCRYPT 2015*. 2015, pp. 430–454. https://doi.org/10.1007/978-3-662-46800-5_17.

[AJL+12] G. Asharov, A. Jain, A. López-Alt, E. Tromer, V. Vaikuntanathan, D. Wichs. "Multiparty Computation with Low Communication, Computation and Interaction via Threshold FHE". *EUROCRYPT 2012*. 2012, pp. 483–501. https://doi.org/10.1007/978-3-642-29011-4_29.

[BJMS20] S. Badrinarayanan, A. Jain, N. Manohar, A. Sahai. "Secure MPC: Laziness Leads to GOD". *ASIACRYPT 2020*. 2020, pp. 120–150. https://doi.org/10.1007/978-3-030-64840-4_5.

[BCK+23] Y. Bae, J. H. Cheon, J. Kim, J. H. Park, D. Stehlé. "HERMES: Efficient Ring Packing Using MLWE Ciphertexts and Application to Transciphering". *CRYPTO 2023*. 2023, pp. 37–69. https://doi.org/10.1007/978-3-031-38551-3_2.

[BGL+23] J. Bell, A. Gascón, T. Lepoint, B. Li, S. Meiklejohn, M. Raykova, C. Yun. "ACORN: Input Validation for Secure Aggregation". *USENIX Security 23*. 2023, pp. 4805–4822.

[BBG+20] J. H. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, M. Raykova. "Secure Single-Server Aggregation with (Poly)Logarithmic Overhead". ACM CCS '20. 2020, pp. 1253–1269. https://doi.org/10.1145/3372297.3417885.

[BD10] R. Bendlin, I. Damgård. "Threshold Decryption and Zero-Knowledge Proofs for Lattice-Based Cryptosystems". *TCC 2010*. 2010, pp. 201–218. https://doi.org/10.1007/978-3-642-11799-2_13.

[BGG+18] D. Boneh, R. Gennaro, S. Goldfeder, A. Jain, S. Kim, P. M. R. Rasmussen, A. Sahai. "Threshold Cryptosystems from Threshold Fully Homomorphic Encryption". *CRYPTO 2018*. 2018, pp. 565–596. https://doi.org/10.1007/978-3-319-96884-1_19.

[BS23] K. Boudgoust, P. Scholl. "Simple Threshold (Fully Homomorphic) Encryption from LWE with Polynomial Modulus". *ASIACRYPT 2023*. 2023, pp. 371–404. https://doi.org/10.1007/978-981-99-8721-4_12.

[BS24] K. Boudgoust, P. Scholl. *Simple Threshold (Fully Homomorphic) Encryption From LWE With Polynomial Modulus*. ePrint 2023/016. 2024. https://eprint.iacr.org/2023/016. Version 2024-07-16.

[BGV12] Z. Brakerski, C. Gentry, V. Vaikuntanathan. "(Leveled) Fully Homomorphic Encryption without Bootstrapping". *ITCS 2012*. ACM, 2012, pp. 309–325. https://doi.org/10.1145/2090236.2090262.

[BP23] L. T. Brandão, R. Peralta. *NIST IR 8214C ipd: NIST First Call for Multi-Party Threshold Schemes (Initial Public Draft)*. 2023. https://doi.org/10.6028/NIST.IR.8214C.ipd.

[CHK+18] J. H. Cheon, K. Han, A. Kim, M. Kim, Y. Song. "Bootstrapping for Approximate Homomorphic Encryption". *EUROCRYPT 2018*. 2018, pp. 360–384. https://doi.org/10.1007/978-3-319-78381-9_14.

[CKKS17] J. H. Cheon, A. Kim, M. Kim, Y. Song. "Homomorphic Encryption for Arithmetic of Approximate Numbers". *ASIACRYPT 2017*. 2017, pp. 409–437. https://doi.org/10.1007/978-3-319-70694-8_15.

[CGGI20] I. Chillotti, N. Gama, M. Georgieva, M. Izabachène. "TFHE: Fast Fully Homomorphic Encryption Over the Torus". *J. Cryptol.* 33.1 (2020), pp. 34–91. https://doi.org/10.1007/s00145-019-09319-x.

[DEG+18] C. Dobraunig, M. Eichlseder, L. Grassi, V. Lallemand, G. Leander, E. List, F. Mendel, C. Rechberger. "Rasta: A Cipher with Low ANDdepth and Few ANDs per Bit". *CRYPTO 2018*. 2018, pp. 662–692. https://doi.org/10.1007/978-3-319-96884-1_22.

[GGHR14] S. Garg, C. Gentry, S. Halevi, M. Raykova. "Two-Round Secure MPC from Indistinguishability Obfuscation". *TCC 2014*. 2014, pp. 74–94. https://doi.org/10.1007/978-3-642-54242-8_4.

[GMPW20] N. Genise, D. Micciancio, C. Peikert, M. Walter. "Improved Discrete Gaussian and SubGaussian Analysis for Lattice Cryptography". *PKC 2020*. 2020, pp. 623–651. https://doi.org/10.1007/978-3-030-45374-9_21.

[Gen09] C. Gentry. "Fully Homomorphic Encryption Using Ideal Lattices". *STOC '09*. ACM, 2009, pp. 169–178. https://doi.org/10.1145/1536414.1536440.

[GPV08] C. Gentry, C. Peikert, V. Vaikuntanathan. "Trapdoors for Hard Lattices and New Cryptographic Constructions". *STOC '08*. ACM, 2008, pp. 197–206. https://doi.org/10.1145/1374376.1374407.

[GSW13] C. Gentry, A. Sahai, B. Waters. "Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based". *CRYPTO 2013*. 2013, pp. 75–92. https://doi.org/10.1007/978-3-642-40041-4_5.

[Gol20] O. Goldreich. "On (Valiant's) Polynomial-Size Monotone Formula for Majority". *Computational Complexity and Property Testing: On the Interplay Between Randomness and Computation*. 2020, pp. 17–23. https://doi.org/10.1007/978-3-030-43662-9_3.

[GLS15] S. Dov Gordon, F.-H. Liu, E. Shi. "Constant-Round MPC with Fairness and Guarantee of Output Delivery". *CRYPTO 2015*. 2015, pp. 63–82. https://doi.org/10.1007/978-3-662-48000-7_4.

[HKL+22] J. Ha, S. Kim, B. Lee, J. Lee, M. Son. "Rubato: Noisy Ciphers for Approximate Homomorphic Encryption". *EUROCRYPT 2022*. 2022, pp. 581–610. https://doi.org/10.1007/978-3-031-06944-4_20.

[HJ85] R. A. Horn, C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985. https://doi.org/10.1017/CBO9780511810817.

[LP11] R. Lindner, C. Peikert. "Better Key Sizes (and Attacks) for LWE-Based Encryption". *CT-RSA 2011*. 2011, pp. 319–339. https://doi.org/10.1007/978-3-642-19074-2_21.

[LPR10] V. Lyubashevsky, C. Peikert, O. Regev. "On Ideal Lattices and Learning with Errors over Rings". *EUROCRYPT 2010*. 2010, pp. 1–23. https://doi.org/10.1007/978-3-642-13190-5_1.

[MP13] D. Micciancio, C. Peikert. "Hardness of SIS and LWE with Small Parameters". *CRYPTO 2013*. 2013, pp. 21–39. https://doi.org/10.1007/978-3-642-40041-4_2.

[MR07] D. Micciancio, O. Regev. "Worst-Case to Average-Case Reductions Based on Gaussian Measures". *SIAM J. Comput.* 37.1 (2007), pp. 267–302. https://doi.org/10.1137/S0097539705447360.

[MS25] D. Micciancio, A. Suhl. "Simulation-Secure Threshold PKE from LWE with Polynomial Modulus". *IACR Communications in Cryptology* 1.4 (Jan. 13, 2025). https://doi.org/10.62056/a0zogy4e-.

[NLV11] M. Naehrig, K. Lauter, V. Vaikuntanathan. "Can homomorphic encryption be practical?" *CCSW '11*. ACM, 2011, pp. 113–124. https://doi.org/10.1145/2046660.2046682.

[OT25] H. Okada, T. Takagi. "Gram Root Decomposition over the Polynomial Ring: Application to Sphericalization of Discrete Gaussian". *ICISSP 2025*. 2025, pp. 306–317. https://doi.org/10.5220/0013139700003899.

[OT26] H. Okada, T. Takagi. "Low Communication Threshold FHE from Standard (Module-)LWE". *ASIACRYPT 2025*. Springer Nature Singapore, 2026, pp. 165–198. https://doi.org/10.1007/978-981-95-5122-4_6.

[PS25] A. Passelègue, D. Stehlé. "Low Communication Threshold Fully Homomorphic Encryption". *ASIACRYPT 2024*. 2025, pp. 297–329. https://doi.org/10.1007/978-981-96-0875-1_10.

[Pei10] C. Peikert. "An Efficient and Parallel Gaussian Sampler for Lattices". *CRYPTO 2010*. 2010, pp. 80–97. https://doi.org/10.1007/978-3-642-14623-7_5.

[PKM+24] R. del Pino, S. Katsumata, M. Maller, F. Mouhartem, T. Prest, M.-J. Saarinen. "Threshold Raccoon: Practical Threshold Signatures from Standard Lattice Assumptions". *EUROCRYPT 2024*. 2024, pp. 219–248. https://doi.org/10.1007/978-3-031-58723-8_8.

[Reg09] O. Regev. "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography". *J. ACM* 56.6 (2009). https://doi.

org/10.1145/1568318.1568324. Preliminary version appeared in STOC '05.

[Sha79] A. Shamir. "How to share a secret". *Commun. ACM* 22.11 (1979), pp. 612–613. https://doi.org/10.1145/359168.359176.

[Val84] L. Valiant. "Short monotone formulae for the majority function". *Journal of Algorithms* 5.3 (1984), pp. 363–366. https://doi.org/10.1016/0196-6774(84)90016-6.

A Definition of Pseudorandom Function

In this section, we provide the (standard) definitions of pseudorandom function which is deferred from Sect. 3. We rely on the multi-instance variant of the pseudorandom functions PRF given in [PKM+24]. Here, we refer to the definition for the completeness.

Definition A.1 (Pseudorandom function). *Let κ be a security parameter and $n, l \in \mathbb{N}$. We say that a deterministic PPT algorithm $\mathsf{PRF} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^l$ is pseudorandom function if for any PPT algorithm $\mathcal{A}$*

$$\mathrm{Adv}^{\mathsf{PRF}}_{\mathcal{A}}(\kappa) := |\Pr[\mathsf{Game}^{\mathsf{PRF}}_{\mathcal{A}}(\kappa) = 1] - \tfrac{1}{2}| = \mathrm{negl}(\kappa)$$

holds, where $\mathsf{Game}^{\mathsf{PRF}}_{\mathcal{A}}$ is defined as in Algo. 8.

Algorithm 8: Security game of PRF (Def. A.1).

$\underline{\mathsf{Game}^{\mathsf{PRF}}_{\mathcal{A}}(\kappa)}$:
1 $\mathsf{List}[\cdot] := \emptyset$
2 $b \leftarrow \{0,1\}$
3 $b' \leftarrow \mathcal{A}^{\mathsf{O}_{\mathsf{PRF}}}(\kappa)$
4 **if** $b = b'$ **then return** 1
5 **else return** 0

$\underline{\mathsf{O}_{\mathsf{PRF}}(i, x \in \{0,1\}^n)}$:
6 **if** $\mathsf{List}[i] = \emptyset$ **then**
  $\mathsf{seed} \leftarrow \{0,1\}^\kappa$, $\mathsf{List}[i] \leftarrow \mathsf{seed}$
7 $\mathsf{seed} \leftarrow \mathsf{List}[i]$
8 $y_0 \leftarrow \mathcal{U}(\{0,1\}^l)$, $y_1 \leftarrow \mathsf{PRF}(\mathsf{seed}, x)$
9 **return** y_b

B Full Proof of Theorem 5.5

We provide the full formal proof of Thm. 5.5; the claim holds by the subsequent Lemmas B.1–B.7. We define hybrid experiments $\mathsf{Hyb}_1, \ldots, \mathsf{Hyb}_6$ in Algorithms 9,...,14, respectively. We now prove the deferred lemmas, i.e., the indistinguishability of the hybrid experiments. As a shorthand, for any algorithm $\mathcal{D}$ and experiments $\mathsf{H}_0(1^\kappa)$ and $\mathsf{H}_1(1^\kappa)$, we define $\mathrm{Adv}^{\mathsf{H}_0\text{-}\mathsf{H}_1}_{\mathcal{D}}(\kappa) := |\Pr[\mathcal{D}(\mathsf{H}_0(1^\kappa)) = 1] - \Pr[\mathcal{D}(\mathsf{H}_1(1^\kappa)) = 1]|$.

Lemma B.1 ($\mathsf{Expt}_{\mathcal{A},\mathsf{Real}} \approx_{\mathsf{s}} \mathsf{Hyb}_1$). *Assume $\epsilon, p, q, Q, \chi_{\mathsf{lwe}}, B_{\mathsf{lwe}}, B_{\mathsf{pub}}, \sigma_{\mathsf{flood}}, \sigma_0, \sigma_1$ are selected as in Thm. 5.3. Then, we have $\mathrm{Adv}^{\mathsf{Hyb}_1\text{-}\mathsf{Hyb}_2}_{\mathcal{D}}(\kappa) = \mathrm{negl}(\kappa)$.*

Proof. The claim holds by Thm. 5.3. □

Algorithm 9: Hyb_1: Differences from $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$ are highlighted.

▷ Expt, OEnc, OEval and ODec are identical to those in $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$

$\underline{\mathsf{ServerDec}(\mathsf{pk},\mathsf{ct},\mathsf{act},\{\mathsf{seed}_i\}_{i\in\mathsf{act}})}$:

▷ Perform lines 9–12 of ServerDec in Algo. 3

$\mathsf{ct}_{\mathsf{dec},0} := \left\lfloor \frac{1}{p}\cdot \mathsf{ct}_{\mathsf{in},0} \right\rceil_{\sigma_0} \bmod q$

$e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z},\sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma_{\mathsf{flood}}^2}},\ \mathfrak{e} := \lfloor \frac{1}{p}e_0 + \{\frac{q}{2}\}\cdot\mu \rceil_{\sigma_1}$

$\mathsf{ct}_{\mathsf{dec},1} = \mathsf{ct}_{\mathsf{dec},0}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \frac{q}{2} \rfloor \cdot \mu + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q,$ where μ is the plaintext of ct

return $(\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid})$

Algorithm 10: Hyb_2: Differences from Hyb_1 are highlighted.

▷ Expt, OEnc, OEval and ODec are identical to those in $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$

$\underline{\mathsf{ServerDec}(\mathsf{pk},\mathsf{ct},\mathsf{act},\{\mathsf{seed}_i\}_{i\in\mathsf{act}})}$:

$\mathsf{sid} \leftarrow \mathsf{sid}+1$, **assert** $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$

for $i \in \mathsf{act}$ **do** $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid}) \in \mathbb{Z}_q$

$\mathsf{ct}_{\mathsf{dec},0} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$

$e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z},\sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma_{\mathsf{flood}}^2}},\ \mathfrak{e} := \lfloor \frac{1}{p}e_0 + \{\frac{q}{2}\}\cdot\mu \rceil_{\sigma_1}$

$\mathsf{ct}_{\mathsf{dec},1} := \mathsf{ct}_{\mathsf{dec},0}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \frac{q}{2} \rfloor \cdot \mu + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q$, where μ is the plaintext of ct

return $(\mathsf{ct}_{\mathsf{dec}} := (\mathsf{ct}_{\mathsf{dec},0}, \mathsf{ct}_{\mathsf{dec},1}) \in \mathcal{C}_{\mathsf{dec}}, \mathsf{sid})$

Lemma B.2 ($\mathsf{Hyb}_1 \approx_{\mathsf{c}} \mathsf{Hyb}_2$). $\mathrm{Adv}_{\mathcal{D}}^{\mathsf{Hyb}_1\text{-}\mathsf{Hyb}_2}(\kappa) \le \mathrm{Adv}_{\mathcal{B}_{\mathsf{LWE}}}^{\mathsf{LWE}(m,n,Q,\chi_{\mathsf{lwe}})}(\kappa)$.

Proof. Let $\mathsf{ct}_{\mathsf{fresh}} := (\mathbf{a}, b) \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$ in the original ServerDec. Recall $\mathbf{a} := \mathbf{r}^{\intercal}\mathbf{A} + \mathbf{f}$ as defined in Algo. 1, then we have $\mathbf{a} \approx_{\mathsf{c}} \mathcal{U}(\mathbb{Z}_q^n)$ under the LWE assumption. Since $\mathsf{ct}_{\mathsf{fresh}}$ is not used elsewhere, we have $\mathsf{ct}_{\mathsf{in},0} \approx_{\mathsf{c}} \mathcal{U}(\mathbb{Z}_Q^n)$. Similarly, $\mathsf{ct}_{\mathsf{dec},0} := \lfloor \frac{1}{p}\cdot \mathsf{ct}_{\mathsf{in},0} \rceil_{\sigma_0} \bmod q \approx_{\mathsf{c}} \mathcal{U}(\mathbb{Z}_q^n)$ also holds. □

Lemma B.3 ($\mathsf{Hyb}_2 \approx_{\mathsf{c}} \mathsf{Hyb}_3$). *We have* $\mathrm{Adv}_{\mathcal{D}}^{\mathsf{Hyb}_2\text{-}\mathsf{Hyb}_3} \le \mathrm{Adv}_{\mathcal{B}_{\mathsf{PRF}}}^{\mathsf{PRF}}(\kappa)$.

Proof. In Hyb_3, the challenger samples $\mathfrak{m}_i \leftarrow \mathcal{U}(\mathbb{Z}_q)$ for all $i \in \mathsf{hon}_{\mathsf{act}}$ instead of the output of PRF. It is easy to see that this hybrid is indistinguishable from Hyb_2 under the assumption on the pseudorandomness of the PRF (Def. A.1). □

Lemma B.4 ($\mathsf{Hyb}_3 \approx \mathsf{Hyb}_4$). *The distributions of* Hyb_3 *and* Hyb_4 *are identical.*

Proof. We analyse the distribution of the partial decryption of the party P_h in Hyb_4; $\mathsf{pd}_h := \lambda_{\mathsf{act},h}\cdot\bar{\mathbf{a}}^{\intercal}\mathbf{s}_h + \mathfrak{m}_h$. By the correctness of Shamir secret sharing, we have $\sum_{i\in\mathsf{act}} \mathsf{pd}_i = \bar{\mathbf{a}}^{\intercal}\mathbf{s} + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q$. Furthermore, the output $(\bar{\mathbf{a}}, \bar{b})$ of ServerDec (in both Hyb_4 and Hyb_5) satisfies $\bar{b} := \bar{\mathbf{a}}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \frac{q}{2} \rfloor \cdot \mu + \sum_{i\in\mathsf{act}} \mathfrak{m}_i \bmod q$. Hence, we obtain $\sum_{i\in\mathsf{act}} \mathsf{pd}_i = \bar{b} - \lfloor \frac{q}{2} \rfloor \cdot \mu - \mathfrak{e}$, which is equivalent to $\mathsf{pd}_h = \bar{b} - \lfloor \frac{q}{2} \rfloor \cdot \mu - \sum_{i\in\mathsf{act}\setminus\{h\}} \mathsf{pd}_i - \mathfrak{e}$. □

Algorithm 11: Hyb_3: Differences from Hyb_2 are highlighted.

▷ Expt, OEnc and OEval are identical to those in $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$

ODec(act):

▷ Perform lines 9–10 of $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$.ODec(act) in Algo. 5

1 $\mathsf{sid} \leftarrow \mathsf{sid}+1$, **assert** $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$

2 **for** $i \in \mathsf{corr}_{\mathsf{act}}$ **do** $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$

3 **for** $i \in \mathsf{hon}_{\mathsf{act}}$ **do** $\mathfrak{m}_i \leftarrow \mathcal{U}(\mathbb{Z}_q)$ ▷ Sample uniformly

4 $\overline{\mathbf{a}} \leftarrow \mathcal{U}(\mathbb{Z}_q^n), e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z},\sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma_{\mathsf{flood}}^2}}$, $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$

5 $\overline{b} := \overline{\mathbf{a}}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \frac{q}{2} \rfloor \cdot \mu + \sum_{i \in \mathsf{act}} \mathfrak{m}_i \bmod q$, where μ is the plaintext of ct

6 **for** $i \in \mathsf{act}$ **do** $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot \overline{\mathbf{a}}^{\intercal}\mathbf{s}_i + \mathfrak{m}_i$

7 **return** $(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}), \{\mathsf{pd}_i\}_{i \in \mathsf{act}})$

Algorithm 12: Hyb_4: Differences from Hyb_3 are highlighted.

▷ Expt, OEnc and OEval are identical to those in $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$

ODec(act):

▷ Perform lines 9–10 of $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$.ODec(act) in Algo. 5

1 $\mathsf{sid} \leftarrow \mathsf{sid}+1$, **assert** $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$

2 **for** $i \in \mathsf{corr}_{\mathsf{act}}$ **do** $\mathfrak{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$

3 **for** $i \in \mathsf{hon}_{\mathsf{act}}$ **do** $\mathfrak{m}_i \leftarrow \mathcal{U}(\mathbb{Z}_q)$

4 $\overline{\mathbf{a}} \leftarrow \mathcal{U}(\mathbb{Z}_q^n), e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z},\sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma_{\mathsf{flood}}^2}}$, $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$

5 $\overline{b} := \overline{\mathbf{a}}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \frac{q}{2} \rfloor \cdot \mu + \sum_{i \in \mathsf{act}} \mathfrak{m}_i \bmod q$, where μ is the plaintext of ct

6 Fix some $h \in \mathsf{hon}_{\mathsf{act}}$

7 **for** $i \in \mathsf{act} \backslash \{h\}$ **do** $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot \overline{\mathbf{a}}^{\intercal}\mathbf{s}_i + \mathfrak{m}_i$

8 $\mathsf{pd}_h := \overline{b} - \lfloor \frac{q}{2} \rfloor \cdot \mu - \sum_{i \in \mathsf{act} \backslash \{h\}} \mathsf{pd}_i - \mathfrak{e}$

9 **return** $(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}), \{\mathsf{pd}_i\}_{i \in \mathsf{act}})$

Lemma B.5 ($\mathsf{Hyb}_4 \approx \mathsf{Hyb}_5$)**.** *The distributions of* Hyb_4 *and* Hyb_5 *are identical.*

Proof. In Hyb_4, $\mathfrak{m}_h \leftarrow \mathcal{U}(\mathbb{Z}_q)$ is used only for calculating $\overline{b} := \overline{\mathbf{a}}^{\intercal}\mathbf{s} + \mathfrak{e} + \lfloor \frac{q}{2} \rfloor \cdot \mu + \sum_{i \in \mathsf{act}} \mathfrak{m}_i \bmod q$ since pd_h in Hyb_4 no longer uses $\mathfrak{m}_h$. Thus, we have $\overline{b} \sim \mathcal{U}(\mathbb{Z}_q)$, and it is identically distributed as $\overline{b}$ in Hyb_5. □

Lemma B.6 ($\mathsf{Hyb}_5 \approx \mathsf{Hyb}_6$)**.** *The distributions of* Hyb_5 *and* Hyb_6 *are identical.*

Proof. In Hyb_5, $\mathfrak{m}_i \leftarrow \mathcal{U}(\mathbb{Z}_q)$ for $i \in \mathsf{hon}_{\mathsf{act}} \backslash \{h\}$ are used only for calculating pd_i. Hence, we have $\mathsf{pd}_i \sim \mathcal{U}(\mathbb{Z}_q)$, which are identically distributed as in Hyb_6. □

Lemma B.7 ($\mathsf{Hyb}_6 \approx \mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$)**.** *The distributions of* Hyb_6 *and* $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ *are identical.*

Proof. Let the PPT simulator Sim of $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ in Algo. 5 as the procedures of lines 2-9 of Hyb_6. Note that $\mathfrak{e} \leftarrow D_{\mathbb{Z},\sqrt{(\sigma_0\|\mathbf{s}\|)^2+(\sigma_{\mathsf{flood}}/p)^2+\sigma_1^2}}$ in line 8 is simulatable since pk contains $\|\mathbf{s}\|$ (see line 3 in Algo. 3). The only difference between

Algorithm 13: Hyb_5: Differences from Hyb_4 are highlighted.

▷ Expt, OEnc and OEval are identical to those in $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$
ODec(act):
▷ Perform lines 9–10 of $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$.ODec(act) in Algo. 5
1 $\mathsf{sid} \leftarrow \mathsf{sid}+1$, **assert** $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$
2 **for** $i \in \mathsf{corr}_{\mathsf{act}}$ **do** $\mathtt{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$
3 Fix some $h \in \mathsf{hon}_{\mathsf{act}}$
4 **for** $i \in \mathsf{hon}_{\mathsf{act}} \backslash \{h\}$ **do** $\mathtt{m}_i \leftarrow \mathcal{U}(\mathbb{Z}_q)$ ▷ $\mathtt{m}_h$ is no longer sampled
5 $\overline{\mathbf{a}} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$, $\overline{b} \leftarrow \mathcal{U}(\mathbb{Z}_q)$ ▷ Sample uniformly
6 **for** $i \in \mathsf{act} \backslash \{h\}$ **do** $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot \overline{\mathbf{a}}^{\intercal}\mathbf{s}_i + \mathtt{m}_i$
7 $e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma_{\mathsf{flood}}^2}}$, $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$
8 $\mathsf{pd}_h := \overline{b} - \lfloor \frac{q}{2} \rfloor \cdot \mu - \sum_{i \in \mathsf{act} \backslash \{h\}} \mathsf{pd}_i - \mathfrak{e}$
9 **return** $(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}), \{\mathsf{pd}_i\}_{i \in \mathsf{act}})$

Algorithm 14: Hyb_6: Differences from Hyb_5 are highlighted.

▷ Expt, OEnc and OEval are identical to those in $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$
ODec(act):
▷ Perform lines 9–10 of $\mathsf{Expt}_{\mathcal{A},\mathsf{Real}}$.ODec(act) in Algo. 5
1 $\mathsf{sid} \leftarrow \mathsf{sid}+1$, **assert** $\{\mathsf{sid} \le L_{\mathsf{Dec}}\}$
2 **for** $i \in \mathsf{corr}_{\mathsf{act}}$ **do** $\mathtt{m}_i := \mathsf{PRF}(\mathsf{seed}_i, \mathsf{sid})$
3 Fix some $h \in \mathsf{hon}_{\mathsf{act}}$ ▷ $\mathtt{m}_i$ is no longer sampled for $i \in \mathsf{hon}_{\mathsf{act}}$
4 $(\overline{\mathbf{a}}, \overline{b}) \leftarrow \mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)$
5 **for** $i \in \mathsf{corr}_{\mathsf{act}}$ **do** $\mathsf{pd}_i := \lambda_{\mathsf{act},i} \cdot \overline{\mathbf{a}}^{\intercal}\mathbf{s}_i + \mathtt{m}_i$
6 **for** $i \in \mathsf{hon}_{\mathsf{act}} \backslash \{h\}$ **do** $\mathsf{pd}_i \leftarrow \mathcal{U}(\mathbb{Z}_q)$ ▷ Sample uniformly
7 $e_0 \leftarrow D_{\frac{1}{2}\mathbb{Z}, \sqrt{(p\sigma_0\|\mathbf{s}\|)^2+\sigma_{\mathsf{flood}}^2}}$, $\mathfrak{e} := \lfloor \frac{1}{p} e_0 + \{\frac{q}{2}\} \cdot \mu \rceil_{\sigma_1}$
8 $\mathsf{pd}_h := \overline{b} - \lfloor \frac{q}{2} \rfloor \cdot \mu - \sum_{i \in \mathsf{act} \backslash \{h\}} \mathsf{pd}_i - \mathfrak{e}$
9 **return** $(\mathsf{ct}_{\mathsf{dec}} := (\overline{\mathbf{a}}, \overline{b}), \{\mathsf{pd}_i\}_{i \in \mathsf{act}})$

Hyb_6 and $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ is that the secret shares $\{\mathsf{sk}_i\}_{i\in[N]} \leftarrow \mathsf{Shamir.Share}_{\mathbb{Z}_q,T,N}(\mathsf{sk})$ in Hyb_6 are replaced with $\{\mathsf{sk}_i\}_{i\in[N]} \leftarrow \mathsf{Shamir.Share}_{\mathbb{Z}_q,T,N}(\mathbf{0})$ in $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$, which contain no information about sk. In Hyb_6 and $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$, only the corrupted secret shares $\{\mathsf{sk}_i\}_{i\in\mathsf{corr}_{\mathsf{act}}}$ are used, and $|\mathsf{corr}_{\mathsf{act}}| \le |\mathsf{corr}| < T$ holds for any sid. Hence, Hyb_6 and $\mathsf{Expt}_{\mathcal{A},\mathsf{Ideal}}$ are identically distributed owing to the privacy of Shamir secret sharing (Constr. 3.17). □

C Proof of Lemma 7.3

Although the proof of Lem. 7.3 is almost identical to that of [OT25, Lemma 5.4], we provide the proof for the completeness:

Proof. By using Cor. 3.9, we first show

$$D_{A_1,\sqrt{\mathbf{\Sigma}_1}} + D_{A_0,\sqrt{\mathbf{\Sigma}_0}} \approx_{\mathsf{s}} D_{\mathbb{Z}^n,\sqrt{\mathbf{\Sigma}_0+\mathbf{\Sigma}_1}}. \tag{32}$$

Note that we have $\widehat{\sigma} \geq \sqrt{2}\widehat{B}\|\mathbf{B}_0\|_{\mathrm{len}}\eta_\epsilon^+(\mathbb{Z}^n) \geq \|\mathbf{B}_0\|_{\mathrm{len}}\eta_\epsilon^+(\mathbb{Z}^n) \geq \eta_\epsilon(A_0)$ by the hypothesis (and Fact 3.6). Hence, we also obtain $\sqrt{\mathbf{\Sigma}_0} \geq \eta_\epsilon(A_0)$ since $\sigma_{\min}(\sqrt{\mathbf{\Sigma}_0}) \geq \widehat{\sigma}$. In addition, we have $\sqrt{(\mathbf{\Sigma}_0^{-1} + \mathbf{\Sigma}_1^{-1})^{-1}} \geq \eta_\epsilon(\mathcal{L}_1(\mathbf{B}_1))$ because

$$\begin{aligned}\sigma_{\min}(\sqrt{(\mathbf{\Sigma}_0^{-1} + \mathbf{\Sigma}_1^{-1})^{-1}}) &\geq \tfrac{1}{\sqrt{2}} \min\{\sigma_{\min}(\sqrt{\mathbf{\Sigma}_0}), \sigma_{\min}(\sqrt{\mathbf{\Sigma}_1})\} \\ &\geq \tfrac{1}{\sqrt{2}}\widehat{\sigma} \geq \|\mathbf{B}_1\|_{\mathrm{len}}\eta_\epsilon^+(\mathbb{Z}^n)\end{aligned}$$

by Fact 7.2 and the hypothesis $\widehat{\sigma} \geq \sqrt{2}\widehat{B}\eta_\epsilon^+(\mathbb{Z}^n) \geq \sqrt{2}\|\mathbf{B}_1\|_{\mathrm{len}}\eta_\epsilon^+(\mathbb{Z}^n)$. Therefore, we obtain (32) by Cor. 3.9 since $A_1 \subseteq A_0$. Next, we show

$$D_{A_2,\sqrt{\mathbf{\Sigma}_2}} + D_{A_0,\sqrt{\mathbf{\Sigma}_0+\mathbf{\Sigma}_1}} \approx_{\mathsf{s}} D_{A_0,\sqrt{\mathbf{\Sigma}_0+\mathbf{\Sigma}_1+\mathbf{\Sigma}_2}} \tag{33}$$

via Cor. 3.9 again. By Fact 3.6, Fact 7.2 and $\widehat{\sigma} \geq \eta_\epsilon(A_0)$, we have $\sqrt{\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1} \geq \eta_\epsilon(A_0)$ because we have

$$\sigma_{\min}(\sqrt{\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1}) \geq \min\{\sigma_{\min}(\sqrt{\mathbf{\Sigma}_0}), \sigma_{\min}(\sqrt{\mathbf{\Sigma}_1})\} \geq \widehat{\sigma} \geq \eta_\epsilon(A_0).$$

Furthermore, we have $\sqrt{((\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1)^{-1} + \mathbf{\Sigma}_2^{-1})^{-1}} \geq \eta_\epsilon(\mathcal{L}_1(\mathbf{B}_2))$ because

$$\begin{aligned}\sigma_{\min}(\sqrt{((\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1)^{-1} + \mathbf{\Sigma}_2^{-1})^{-1}}) &\geq \tfrac{1}{\sqrt{2}} \min\{\sigma_{\min}(\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1), \sigma_{\min}(\mathbf{\Sigma}_2)\} \\ &\geq \tfrac{1}{\sqrt{2}} \min\{\sigma_{\min}(\mathbf{\Sigma}_0), \sigma_{\min}(\mathbf{\Sigma}_1), \sigma_{\min}(\mathbf{\Sigma}_2)\} \\ &\geq \tfrac{1}{\sqrt{2}}\widehat{\sigma} \geq \|\mathbf{B}_2\|_{\mathrm{len}}\eta_\epsilon^+(\mathbb{Z}^n)\end{aligned}$$

holds by the hypothesis $\widehat{\sigma} \geq \sqrt{2}\widehat{B}\eta_\epsilon^+(\mathbb{Z}^n) \geq \sqrt{2}\|\mathbf{B}_2\|_{\mathrm{len}}\eta_\epsilon^+(\mathbb{Z}^n)$. Hence, we obtain (33) by Cor. 3.9. Repeating the above, we obtain (31). □