

“Preview Writeup”: In anticipation of a package submission to the NIST Threshold Call

Title: Holocron

Subtitle: Threshold K-PKE Encryption and Decryption from FHE

Version: Preview Writeup 1.0 (2026-09-28)¹

Team name: Holocron

Team members: Kyoohyung Han, Byeonghak Lee, Jihoon Kwon, Sangyub Lee, Hyojin Yoon, Jihoon Cho, Jaejun Ko, Seonhong Min, Yongsoo Song, Intak Hwang, Hwajeong Seo, Siwoo Eum

Abstract: This preview writeup presents our planned submission for Category N2 of the NIST Multi-Party Threshold Cryptography (MPTC) Call, proposing Holocron, a practical lattice-based threshold framework for K-PKE that supports dishonest majority t -out-of- n access structures. K-PKE is the core public-key encryption scheme underlying the NIST-specified Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM). Our construction guarantees complete ciphertext interchangeability. To achieve this, our threshold protocols (Holocron) rely on fully homomorphic encryption (FHE). In particular, Holocron.Encrypt leverages ciphertexts from the BFV FHE scheme and short-integer bootstrapping (SI-BTS) from the CKKS FHE scheme to perform distributed centered binomial distribution (CBD) sampling before transitioning to additive shares for linear K-PKE encryption. For decryption, Holocron.Decrypt embeds standard ciphertexts into a higher-degree BGV FHE scheme, utilizing modulus expansion to establish a noise-modulus gap that safely enables Gaussian noise flooding without plaintext corruption. Furthermore, our framework guarantees security-with-abort against static active adversaries by integrating lattice-based non-interactive zero-knowledge (NIZK) proofs to verify the well-formedness of cryptographic outputs.

Proposed crypto-systems:

- (I) Holocron: Threshold K-PKE (Encrypt and Decrypt) from FHE (Categories N2)

Keywords: Threshold Cryptography; NIST Threshold Call; Lattice Cryptography; Homomorphic Encryption; Bootstrapping; Post-Quantum Cryptography; ML-KEM; K-PKE



(Logo AI-generated with ChatGPT)

¹Version submitted to NIST-MPTC for publication

Preview writeup. This document is provided to NIST for online publication, to foster public awareness and support public discussion within the scope of the NIST First Call for Multi-Party Threshold Schemes [NIST-IR8214C]. This “preview writeup” represents a good-faith plan for a subsequent “package submission”. However, until the deadline for package submission, the team may still modify its own composition and the submission plan, including possible changes to the technical scope, and/or the used techniques or achieved results.

Team members: Kyoohyung Han^{i1,a1}, Byeonghak Lee^{i2,a1}, Jihoon Kwon^{i3,a1}, Sangyub Lee^{i4,a1}, Hyojin Yoon^{i5,a1}, Jihoon Cho^{i6,a1}, Jaejun Ko^{i7,a2}, Seonhong Min^{i8,a2}, Yongsoo Song^{i9,a2}, Intak Hwang^{i10,a2}, Hwajeong Seo^{i11,a3}, Siwoo Eum^{i12,a3}

Open Researcher and Contributor Identifiers (ORCID):

i1 (0000-0002-8410-3386); i2 (0000-0003-2736-6830); i3 (0000-0001-9093-4189); i4 (0000-0001-6709-293X); i5 (0009-0005-5914-3676); i6 (0000-0001-9108-4323); i7 (0009-0005-0062-0343); i8 (0000-0002-1151-2453); i9 (0000-0002-0496-9789); i10 (0009-0008-1171-1824); i11 (0000-0003-0069-9061); i12 (0000-0002-9583-5427)

Affiliations:

^{a1} Samsung SDS @ Seoul, South Korea

^{a2} Computer Science and Engineering, Seoul National University @ Seoul, South Korea

^{a3} SW Convergence, Hansung University @ Seoul, South Korea

Main contacts:

- **Team mailing list:** holocron-threshold-kpke-fhe@googlegroups.com
- **Primary technical contact person:** Kyoohyung Han, kh89.han@samsung.com

Produced by humans. The team hereby confirms that the content in this preview writeup: (i) was produced by the team members, and (ii) was not produced by generative artificial intelligence (AI), with the possible exception of AI-proposed grammar improvements, minor integrated suggestions, or some well-identified and short localized portions of auxiliary content (e.g., some illustration); and (iii) was proofread by the team members.

1. Introduction

This preview writeup presents our planned submission to the NIST Multi-Party Threshold Cryptography (MPTC) Call [NIST-IR8214C], proposing a practical, lattice-based threshold framework for K-PKE, constructed via fully homomorphic encryption (FHE) primitives. K-PKE is the public-key encryption (PKE) component scheme specified in FIPS 203 [FIPS203], upon which the Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM) is built: ML-KEM.Encaps and ML-KEM.Decaps invoke K-PKE.Encrypt and K-PKE.Decrypt internally. Our scheme is specifically tailored for Category N2 (threshold schemes for K-PKE.Encrypt and K-PKE.Decrypt), achieving full interoperability and compatibility with the standardized FIPS 203 specification.

A defining mandate of Category N2 is ciphertext interchangeability. The ciphertexts produced by our threshold encryption protocol (Π_{ThEnc}) are fully valid standard K-PKE ciphertexts, meaning they can be seamlessly decrypted by any standard single-key K-PKE.Decrypt algorithm. Conversely, standard ciphertexts generated by any third-party single-key implementation can be directly threshold-decrypted via our decryption protocol (Π_{ThDec}) without requiring any structural or parameter modifications.

Our construction leverages FHE as a thresholdizer. For encryption, parties jointly sample the centered binomial distribution (CBD) randomness of K-PKE under BFV [FV12; Bra12] and CKKS [CKKS17] short-integer bootstrapping (SI-BTS) [BKSS24], extract additive shares via message flooding, and complete the encryption with local linear operations. For decryption, standard K-PKE ciphertexts are embedded into a higher-degree BGV [BGV14] scheme, whose modulus expansion opens a noise-modulus gap for Gaussian noise flooding that hides the secret-key shares. Active security is obtained by attaching a sublinear Buckler non-interactive zero-knowledge (NIZK) proof [HLSS25] to every broadcast message, which enforces well-formedness, correct use of the secret-key shares, and flooding noise bounds, and yields security-with-abort.

From threshold K-PKE to ML-KEM. Our proposal targets the K-PKE primitives in Category N2, not the higher-level ML-KEM (see Table 7 of [NIST-IR8214C]). ML-KEM.Encaps derives plaintext and randomness from a hashed seed, whereas our protocol treats the plaintext as a secret-shared input and jointly generates CBD randomness. Consequently, the ciphertext output by our threshold encryption protocol (Holocron.Encrypt) can be decrypted by the conventional K-PKE.Decrypt algorithm.

2. Specification

2.1. Organization

Our planned specification presents two modular threshold protocols for the K-PKE component scheme underlying ML-KEM in [FIPS203]: the threshold encryption protocol Π_{ThEnc} for K-PKE.Encrypt and the threshold decryption protocol Π_{ThDec} for K-PKE.Decrypt. As illustrated in Figure 1, Π_{ThEnc} produces ciphertexts compatible with conventional K-PKE, while Π_{ThDec} directly processes ciphertexts generated by conventional K-PKE.Encrypt.

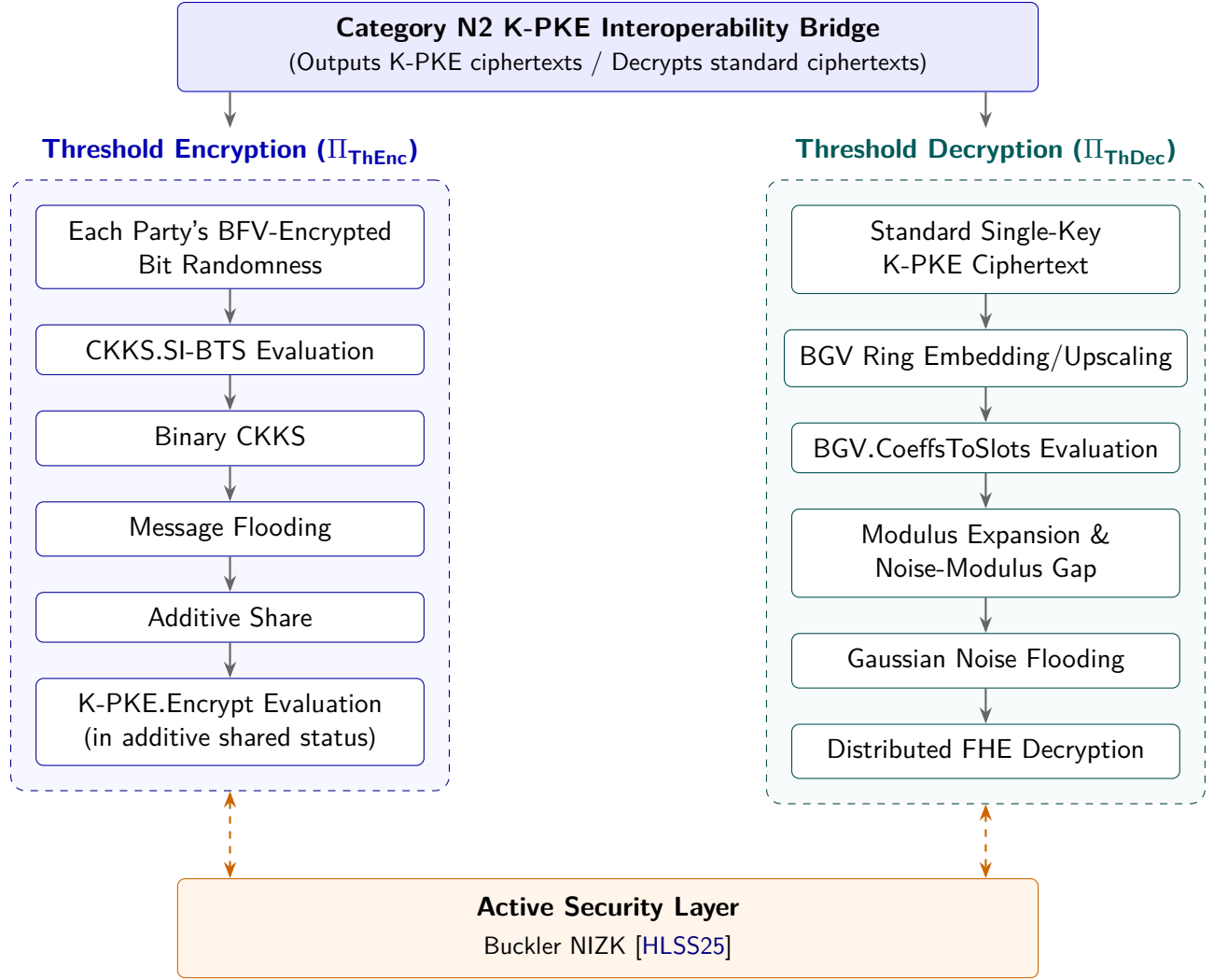


Figure 1: Modular Architecture of the Proposed Threshold K-PKE Scheme

2.1.1. Threshold Encryption Pipeline

The threshold encryption pipeline constructs a conventional K-PKE ciphertext from jointly generated randomness. To initiate this protocol, each participating party begins with a Shamir secret share of the message (using a modulus q equal to the K-PKE modulus) and a Shamir secret share of the FHE (BFV/CKKS) secret key (using the modulus required for FHE partial decryption).

- **Encrypted joint randomness.** Each party samples secret random bits and encrypts them into low-level BFV ciphertexts with a plaintext modulus of 2. After aggregation, CKKS SI-BTS converts the aggregated BFV ciphertext into a binary CKKS ciphertext. This conversion enables homomorphic integer additions over the ciphertexts, facilitating the distributed Centered Binomial Distribution (CBD) sampling required for the joint randomness.

- **Additive-share extraction and encryption.** Message-flooded partial decryption converts the encrypted randomness into $(t\text{-out-of-}t)$ additive shares without revealing the underlying values. Each party locally evaluates the linear portion of `K-PKE.Encrypt` using its additive share. Aggregating the resulting encryption shares cancels the masking terms and adds message terms (obtained from the Shamir shares via Lagrange coefficients), and yields a conventional K-PKE ciphertext.

2.1.2. Threshold Decryption Pipeline

The threshold decryption pipeline evaluates the non-linear rounding operations required by `K-PKE.Decrypt`. At the onset of this pipeline, each party begins with a Shamir secret share of the BGV secret key (using the FHE partial decryption modulus). Furthermore, the protocol relies on a dealer-provided Key-Switching Key to convert the target K-PKE ciphertext into a BGV ciphertext.

- **BGV embedding and homomorphic evaluation.** The input K-PKE ciphertext is embedded into a BGV cyclotomic ring of a higher degree (e.g., 2^{16}) than the standard degree-256 K-PKE ring, utilizing a plaintext modulus of $p_{\text{BGV}} = 7681$. Because $7681 \equiv 1 \pmod{512}$ but $7681 \not\equiv 1 \pmod{1024}$, the plaintext ring splits into exactly 256 slots for any power-of-two degree $N \geq 256$, each slot being an extension field of degree $N/256$; this enables the entire K-PKE ciphertext to be packed into a single HE ciphertext. The `CoeffsToSlots` algorithm and subsequent homomorphic operations then extract the most significant bits necessary for decryption.
- **Modulus expansion and Gaussian flooding.** Homomorphic bootstrapping is leveraged to expand the ciphertext modulus, artificially constructing a massive gap between the inherent algorithmic noise and the modulus boundary. Thanks to securing this noise-modulus gap via bootstrapping, it becomes possible to safely inject massive Gaussian flooding noise during the FHE partial decryption process. This safely masks each party's BGV secret-key shares without overflowing or corrupting the recovered plaintext.
- **Output mode.** Our threshold decryption protocol targets the non-secret-shared (NSS) output mode of [NIST-IR8214C], where the aggregated partial decryptions reveal the plaintext in the clear. Since the final step is a linear aggregation of partial decryptions, the protocol naturally extends to the secret-shared output (SSO) mode. We leave the specification of the SSO variant to future work.

2.1.3. Active Security Layer

Whenever a party broadcasts a newly generated ciphertext or a partial decryption share, it attaches a Buckler NIZK proof. These proofs verify the well-formedness of the cryptographic outputs while ensuring that the HE ciphertexts and partial decryptions are correctly computed using the valid HE public key and the party's HE secret-key share, respectively.

For instance, the NIZK statements include that the randomness used to generate a homomorphic ciphertext lies within a specified bound and that the encrypted message has binary coefficients;

for partial decryptions, that the share is computed with the party's secret-key share and that the flooding noise lies within the prescribed bound.

2.2. System Model

This section formalizes the system entities, threshold access structure, communication network assumptions, and operational phases governing our threshold K-PKE framework.

2.2.1. Parties and Threshold Access Structure

We consider a set of n distinct parties, denoted by $\mathcal{P} = \{P_1, P_2, \dots, P_n\}$. Our framework operates under a standard (t, n) -threshold access structure ($1 < t \leq n$): Any subset $\mathcal{S} \subseteq \mathcal{P}$ of at least t parties ($|\mathcal{S}| \geq t$) can collaboratively execute the threshold encryption (Π_{ThEnc}) or threshold decryption (Π_{ThDec}) protocol.

2.2.2. System Entities and Operational Roles

Our architecture defines two primary cryptographic roles:

- **Trusted Key Dealer:** Responsible for executing the one-time offline setup phase (Setup). The dealer generates and securely distributes the following cryptographic materials:
 - The standard K-PKE secret key and the K-PKE public key. [common]
 - The BFV secret key (and its corresponding t -out-of- n Shamir secret shares) and public key, alongside the public keys required for CKKS SI-BTS. (Note that the BFV and CKKS schemes are fully compatible and share the exact same secret key.) [Π_{ThEnc}]
 - The BGV secret key (and its corresponding t -out-of- n Shamir secret shares), public key, and public keys for BGV bootstrapping. [Π_{ThDec}]
 - A public Key-Switching Key designed to seamlessly convert K-PKE ciphertexts into BGV ciphertexts. [Π_{ThDec}]

This entity can be replaced by an interactive distributed key generation (DKG) protocol in future extensions.

- **Threshold Parties:** Individual processing nodes that securely store their assigned Shamir secret key shares. In the online phase, parties independently perform homomorphic evaluations, generate Buckler NIZK proofs, and broadcast their local partial evaluation shares.

2.2.3. Communication Network Assumptions

We assume a synchronous network model with the participating set $\mathcal{S} \subseteq \mathcal{P}$, fixed and identified before execution. Formally, we assume the following networking infrastructure:

- **Broadcast-with-Abort:** Parties communicate over authenticated point-to-point channels. We do not assume a reliable broadcast channel. Instead, we assume the weaker *broadcast-with-abort* primitive, which can be instantiated over point-to-point channels via the standard echo-broadcast protocol of Goldwasser and Lindell [GL05]; any inconsistency among the messages received by honest parties causes the detecting parties to abort.

2.3. Security

We consider a standard static active (malicious) adversary corrupting up to $t - 1$ parties. Corrupted parties may arbitrarily deviate from the protocol (e.g., by sending malformed shares, injecting noise, submitting invalid NIZK proofs, or equivocating messages—addressed via the broadcast-with-abort in Section 2.2.3).

Our framework does not require an honest majority, supporting a dishonest majority up to $t - 1$ corruptions (including up to $n - 1$ in the full-threshold case). Against such adversaries, we guarantee security-with-abort. While corrupted parties can force the protocol to abort (preventing guaranteed output delivery), doing so leaks no information about the underlying plaintexts or honest parties’ secret key shares.

3. Open-Source Implementation

We plan to provide a complete, open-source software implementation of our framework in Go (Golang). Our software relies on two state-of-the-art Go cryptographic libraries:

- **Lattigo [MBTH20]:** Used for the multiparty homomorphic encryption modules (threshold CKKS and BGV) to execute distributed randomness generation and homomorphic evaluations.
- **ringo-snark [Hwa25]:** A zero-knowledge PIOP toolkit providing the Buckler NIZK primitives to prove the well-formedness of HE ciphertexts and partial decryptions, thereby guaranteeing active security.

4. Experimental Performance Evaluation

In this section, we present a preliminary performance evaluation of our threshold K-PKE framework. As of this preview writeup, our internal implementation covers the semi-honest protocol without zero-knowledge proofs. We are actively developing the fully secure open-source implementation that achieves security-with-abort by integrating the Buckler NIZK framework. We expect the integration of the Buckler NIZK proofs to add a constant-factor overhead to the per-party computation for a fixed threshold profile, with proof sizes sublinear in the witness size [HLSS25]. The NIZK relations are stated over the HE ciphertext modulus, whose bit-length grows logarithmically in t (to accommodate the aggregated flooding noise), so the proving and verification costs scale

as $O(\log t)$ through the modulus size. Comprehensive experimental results for the active-secure setting will be provided in the final package submission.

We benchmarked the Go implementation of the protocols targeting the ML-KEM-1024 parameter set on a single thread of an Intel Xeon Platinum 8268 Linux machine. Table 1 reports the average execution time over 10 runs and the communication size per party for the current semi-honest implementation, excluding the one-time offline setup.

Protocol	Execution Time	Communication Size
Π_{ThEnc}	12.13 s	3.502 MiB
Π_{ThDec}	27.01 s	1.000 MiB

Table 1: Measured online computation and total communication (sent and received) per party with $n = 16$ and $t = 10$ (semi-honest implementation, without NIZK proofs).

The following observations highlight the efficiency and scalability of our protocols based on these preliminary results:

- **Computation-Bound Online Phase:** The protocol Π_{ThEnc} (resp. Π_{ThDec}) requires only three (resp. one) online broadcast rounds,² making it highly advantageous in real-world high-latency networks. The overall online latency is dominated by local computations: CKKS SI-BTS for Π_{ThEnc} and BGV modulus expansion for Π_{ThDec} .
- **Offline-Heavy, Online-Light Architecture:** Our framework relies on threshold FHE primitives that inherently require large public evaluation keys (e.g., relinearization and automorphism keys). However, the generation and distribution of these heavy cryptographic assets are strictly relegated to the one-time offline Setup phase, keeping the online threshold operations lightweight and highly scalable.

5. Licensing, Patent Claims, and Funding

We plan to release the entire open-source software implementation of our proposed threshold K-PKE scheme under the MIT License.

We hereby disclose that there are pending patent applications related to the core cryptographic techniques and homomorphic encryption-based thresholdization methodologies utilized in this scheme. The known issued or pending patents will be listed in the Notes on Patent Claims of the package submission, as required by the Threshold Call [NIST-IR8214C].

This work did not receive any dedicated external funding; the research was supported by the authors' respective institutions.

²Round counts exclude the echo messages of the broadcast-with-abort instantiation, which can be piggybacked on the following round.

6. Comparison with Other Submissions

To our knowledge, no other publicly available proposal addresses Category N2 for K-PKE or ML-KEM. Since our framework is built on threshold FHE, we instead position it relative to the two Phase-1 Category S5 submissions that specify threshold FHE: Zama’s Nexus suite [Tea26b] and Pantheria [Tea26a].

6.1. Scope and Key Setup

Both are general-purpose threshold FHE systems: Nexus provides robust key generation and decryption for BGV, BFV, and TFHE via generic MPC over Galois rings, and Pantheria proposes actively secure Th-(d)BFV, Th-FHEW, and Th-BGV with MPC-based distributed decryption [ZZLP25]. Our proposal uses threshold FHE for a single fixed circuit, the standardized K-PKE algorithms, and treats key setup as out of scope. Our trusted dealer is the trusted-dealer option that Pantheria lists among its key-generation options, and a DKG from either submission could replace it, since the dealer only outputs Shamir-shared RLWE keys and public evaluation keys.

6.2. Use of Bootstrapping and Distributed Decryption

Our technical core is the use of bootstrapping in two places: CKKS SI-BTS to generate the CBD randomness of K-PKE. Encrypt distributively, and BGV bootstrapping to expand the modulus of an embedded K-PKE ciphertext, creating the noise-modulus gap that makes flooding possible without altering the standardized parameters or ciphertext format. The distributed decryption step itself is textbook Gaussian noise flooding, with active security from Buckler NIZK proofs—the proof system Pantheria also plans to use for key generation—rather than from MPC. The advanced distributed decryption techniques of the S5 submissions could replace this step to allow partial decryption at a smaller modulus, which we regard as a natural direction for future work.

6.3. Threshold Profiles and Network Model

Flooding imposes no inherent restriction on the threshold profile: our protocols support any (t, n) access structure with $1 < t \leq n$, where t is the number of parties required to encrypt or decrypt, and remain secure-with-abort against up to $t - 1$ corruptions, including a dishonest majority, in a synchronous network. Nexus achieves guaranteed output delivery over an asynchronous network at the cost of an honest super-majority, tolerating at most t_N corrupted parties with $t_N < n/3$ (here t_N denotes Nexus’s corruption bound, which is distinct from our access threshold t), while Pantheria scopes its profiles to $n \leq 64$ and targets active adaptive security for decryption.

References

- [BGV14]** Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. “(Leveled) fully homomorphic encryption without bootstrapping”. In: *ACM Transactions on Computation Theory* 6.3 (2014), 13:1–13:36. DOI: [10.1145/2633600](https://doi.org/10.1145/2633600). Also at ia.cr/2011/277.
- [BKSS24]** Youngjin Bae, Jaehyung Kim, Damien Stehlé, and Elias Suvanto. “Bootstrapping small integers with CKKS”. In: *Advances in Cryptology – ASIACRYPT 2024*. Vol. 15484. LNCS. Springer, 2024, pp. 330–360. DOI: [10.1007/978-981-96-0875-1_11](https://doi.org/10.1007/978-981-96-0875-1_11). Also at ia.cr/2024/1637.
- [Bra12]** Zvika Brakerski. “Fully homomorphic encryption without modulus switching from classical GapSVP”. In: *Advances in Cryptology – CRYPTO 2012*. Vol. 7417. LNCS. Springer, 2012, pp. 868–886. DOI: [10.1007/978-3-642-32009-5_50](https://doi.org/10.1007/978-3-642-32009-5_50). Also at ia.cr/2012/078.
- [CKKS17]** Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. “Homomorphic encryption for arithmetic of approximate numbers”. In: *Advances in Cryptology – ASIACRYPT 2017*. Vol. 10624. LNCS. Also at ia.cr/2016/421. Springer, 2017, pp. 409–437. DOI: [10.1007/978-3-319-70694-8_15](https://doi.org/10.1007/978-3-319-70694-8_15). Also at ia.cr/2016/421.
- [FIPS203]** National Institute of Standards and Technology. *Module-Lattice-Based Key-Encapsulation Mechanism Standard*. Federal Information Processing Standards Publication NIST FIPS 203. U.S. Department of Commerce, August 2024. DOI: [10.6028/NIST.FIPS.203](https://doi.org/10.6028/NIST.FIPS.203).
- [FV12]** Junfeng Fan and Frederik Vercauteren. *Somewhat practical fully homomorphic encryption*. Cryptology ePrint Archive, Paper 2012/144. 2012. URL: <https://eprint.iacr.org/2012/144>.
- [GL05]** Shafi Goldwasser and Yehuda Lindell. “Secure multi-party computation without agreement”. In: *Journal of Cryptology* 18.3 (2005), pp. 247–287. DOI: [10.1007/s00145-005-0319-z](https://doi.org/10.1007/s00145-005-0319-z).
- [HLSS25]** Intak Hwang, Hyeonbum Lee, Jinyeong Seo, and Yongsoo Song. “Practical zero-knowledge PIOP for maliciously secure multiparty homomorphic encryption”. In: *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS 2025)*. ACM, 2025, pp. 4049–4063. DOI: [10.1145/3719027.3765229](https://doi.org/10.1145/3719027.3765229). Also at ia.cr/2024/1879.
- [Hwa25]** Intak Hwang. *ringo-snark: Zero-knowledge PIOP toolkit for ring-LWE relations in Go*. Open-source software (Apache License 2.0). 2025. URL: <https://github.com/sp301415/ringo-snark> (visited on 09/16/2026).
- [MBTH20]** Christian Vincent Mouchet, Jean-Philippe Bossuat, Juan Ramón Troncoso-Pastoriza, and Jean-Pierre Hubaux. “Lattigo: a Multiparty Homomorphic Encryption Library in Go”. In: *Proceedings of the 8th Workshop on Encrypted Computing and Applied Homomorphic Cryptography (WAHC 2020)*. Software (Apache License 2.0) available at <https://github.com/tuneinsight/lattigo>. 2020, pp. 64–70. DOI: [10.1145/3411501.3419423](https://doi.org/10.1145/3411501.3419423).

[Tea26a] Team PANTHERIA. *PANTHERIA: Threshold FHE for RLWE-Based Cryptosystems*. Preview Writeup submitted to the NIST First Call for Multi-Party Threshold Schemes. January 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/PANTHERIA-PW01.pdf>.

[Tea26b] Team Zama. *TFHE, ZHENith and Nexus: A Suite of Cryptosystems to Enable Fully Homomorphic Encryption Applications*. Preview Writeup submitted to the NIST First Call for Multi-Party Threshold Schemes. January 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Zama-PW01.pdf>.

[ZZLP25] Guy Zyskind, Doron Zarchy, Max Leibovich, and Chris Peikert. “High-Throughput Universally Composable Threshold FHE Decryption”. In: *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS 2025)*. Also at ia.cr/2025/1781. ACM, 2025, pp. 2339–2353. DOI: [10.1145/3719027.3744884](https://doi.org/10.1145/3719027.3744884).

[NIST-IR8214C] Luís T. A. N. Brandão and René Peralta. *NIST First Call for Multi-Party Threshold Schemes*. (National Institute of Standards and Technology) NIST Internal Report (NISTIR) 8214C. 2026. DOI: [10.6028/NIST.IR.8214C](https://doi.org/10.6028/NIST.IR.8214C).