

PANTHERIA: Threshold FHE for RLWE-Based Cryptosystems

Preview Talk for NIST MPTC

Yuriy Polyakov (Duality & OpenFHE)
Chris Peikert (University of Michigan & Fhenix)
Zeyu Liu (Yale University & OpenFHE)

PANTHERIA Team

Team members: Andreea Alexandru ^{i1,a1,a2}, Ahmad Al Badawi ^{i2,a1,a2}, Daniel Apon ^{i3,a3}, Jean-Philippe Bossuat ^{i4,a4,a9}, Sylvain Chatel ^{i5,a5}, Ben Fisch ^{i6,a6}, Nicholas Genise ^{i7,a2,*}, Shai Halevi ^{i8,a7,a2}, Loïs Huguenin-Dumittan ^{i9,a8,a9}, Guy Itzhaki ^{i10,a11}, Andrey Kim ^{i11,a2}, Yongwoo Lee ^{i12,a13,a14,a2}, Zeyu Liu ^{i13,a6,a2}, Janmajaya Mall ^{i14,a4}, Christian Mouchet ^{i15,a15,a9}, Carlo Pascoe ^{i16,a1,a2}, Chris Peikert ^{i17,a10,a11}, Kabir Peshawaria ^{i18,a12}, Yuriy Polyakov ^{i19,a1,a2}, Saraswathy R.V. ^{i20,a2,*}, Sarabjeet Singh ^{i21,a1,a2}, Yongsoo Song ^{i22,a16}, Eran Tromer ^{i23,a12}, Vinod Vaikuntanathan ^{i24,a17,a1}, Vincent Zucca ^{i25,a2}, Guy Zyskind ^{i26,a11}

^{a1} Duality Technologies @ Hoboken, NJ, USA

^{a2} OpenFHE team

^{a3} Anduril Industries @ Costa Mesa, CA, USA

^{a4} Poulpy team

^{a5} CISA Helmholtz Center for Information Security @ Saarbrücken, Germany

^{a6} Yale University @ New Haven, CT, USA

^{a7} AWS @ New York, NY, USA

^{a8} Tune Insight @ Lausanne, Switzerland

^{a9} Lattigo team

^{a10} University of Michigan @ Ann Arbor, MI, USA

^{a11} Fhenix @ Tel Aviv, Israel

^{a12} Boston University @ Boston, MA, USA

^{a13} Inha University @ Incheon, South Korea

^{a14} DESILO Inc. @ Seoul, South Korea

^{a15} Hasso-Plattner-Institute, University of Potsdam @ Potsdam, Germany

^{a16} Seoul National University @ Seoul, South Korea

^{a17} EECS & CSAIL, Massachusetts Institute of Technology @ Cambridge, MA 02139, USA

Agenda

Foundations and Background

- Background on conventional FHE cryptosystems
- Background on Threshold FHE cryptosystems

Proposal and Methodology

- What cryptosystems we propose (at a high level)
- Our approach for reference implementations
- Our approach for performance evaluation

Key Components

- (Decomposed) BFV
- MPC-based threshold decryption
- Verifiable homomorphic evaluation

Motivation for standardizing FHE

- Fully Homomorphic Encryption (FHE) is a powerful cryptography method that supports computations over encrypted data (without intermediate decryption)
- FHE has already been successfully used for a number practical use cases, e.g., PIR, private database query, encrypted statistics, image classification
 - There are many startups developing solutions based on FHE
 - Large organizations launched some products based on FHE, e.g., Microsoft, Apple, etc.
- All common FHE schemes are considered quantum-resilient: based on the same/related lattice problems as NIST-certified PQC algorithms, such as ML-KEM (Kyber) and ML-DSA (Dilithium)
- Multiple open-source libraries with FHE scheme implementations are available

Conventional FHE cryptosystems

All conventional FHE schemes are based on the (Ring / Module) Learning With Errors (**LWE**) problem. The main FHE schemes are:

- Brakerski-Gentry-Vaikuntanathan (BGV) [BGV11],
- Brakerski/Fan-Vercauteren (BFV) [Bra12; FV12],
- Gentry-Sahai-Waters (GSW) [GSW13] and its variants/enhancements:
 - “FHEW”: Ducas-Micciancio [DM15],
 - “TFHE”: Chilotti-Gama-Georgieva-Izabachene (CGGI) [CGGI16],
 - Lee-Micciancio-Kim-Choi-Deryabin-Eom-Yoo (LMK+) [LMKCDEY23] FHEW improvement
- Cheon-Kim-Kim-Song (CKKS) [CKKS17] “approximate” FHE

All these schemes support **bootstrapping** and can be used for performing an arbitrary homomorphic computation.

What these FHE cryptosystems are used for

- BGV and BFV are typically used for *finite-field/ring arithmetic over vectors*
 - BGV and BFV are best suited for Private Information Retrieval and Private Set Intersection; they are also used for blockchain applications.
- CGGI and DM/LMK+ are commonly used for low-latency operations over *(few) small integers*; they also feature low-latency lookup table evaluation
 - GCGI and DM/LMK+ are often used for blockchain applications and Boolean circuit evaluation.
- CKKS is often used for applications that deal with (approximate) arithmetic *over vectors of real/complex numbers*
 - CKKS is the main scheme for machine learning/AI applications.

Open-source libraries maintained by PANTHERIA team

- OpenFHE (C++, Python)
 - Implements all common FHE schemes
 - BSD-2 license
- Lattigo (Go)
 - Implements BGV, BFV, and CKKS
 - Apache 2.0 license
- Poulpy (Rust)
 - Implements LMK+
 - Apache 2.0 license

The current FHE scheme implementations in the libraries target *passive security*, i.e., honest-but-curious models.

Threshold FHE

Threshold FHE cryptosystems are built from conventional FHE schemes by

- replacing single-party key generation with **distributed key generation**,
- replacing single-party decryption with **distributed decryption**.

OpenFHE and Lattigo provide implementations of threshold BGV, BFV, and CKKS.

The current implementations target passive security.

A modular view: Building blocks for threshold FHE

Leaving aside parameter selection, a Threshold FHE (Th-FHE) scheme is composed of the following building blocks:

- A distributed key generation protocol **KeyGen**;
- A (typically non-interactive) public-key encryption algorithm **Encrypt**;
- A (typically non-interactive) homomorphic evaluation algorithm **Evaluate**; and
- A distributed decryption protocol **Decrypt**.

The **Encrypt** and **Evaluate** algorithms are typically the same as in the case of conventional FHE schemes—with the exception that *larger parameters* are sometimes needed in the threshold setting, for security of the **Decrypt** protocol.

KeyGen and **Decrypt** are specific to the threshold instantiation.

Distributed key generation

We consider the following options:

1. trusted dealer,
2. private channel for initial Shamir secret sharing and homomorphic addition of secret key shares
3. homomorphic addition of key secret shares
4. MPC key generation.

Options (2) and (3) are currently implemented in Lattigo and OpenFHE.

Distributed decryption

The two main options in the literature to achieve passive security for distributed decryption are **MPC** and **noise flooding**.

The current implementations in OpenFHE and Lattigo rely on noise flooding, i.e., addition of large noise to “erase” the traces of the noise from prior encryption/computation steps.

Main challenges in standardizing (Th)-FHE

- The current implementations target passive security, e.g., IND-CPA, both for conventional and threshold FHE instantiations
- NIST-standardized schemes, including PQC KEM and DSA, target active adaptive security (IND-CCA2)
- Due to malleability of FHE, IND-CCA2 cannot be achieved for end-to-end FHE cryptosystems
 - However, new notions that are stronger than IND-CCA1 have been proposed for FHE, e.g., vCCA
- Active security models can be achieved for individual components of Th-FHE schemes, e.g., distributed decryption
 - This will be the focus of our proposed cryptosystems

What we propose

- We propose three threshold FHE cryptosystems that build upon existing FHE schemes but augment them with new properties: **Th-(d)BFV**, **Th-FHEW**, and **Th-BGV**
- **Th-(d)BFV** adds a recent *decomposed* variant of BFV [PZZ25] and uses a recent efficient active-security MPC threshold decryption method [ZZLP25]
- **Th-FHEW** implements LMK+ with smaller key generation noise and uses the MPC-based active-security MPC threshold decryption method from [ZZLP25]
- For **Th-BGV**, we will also adapt the MPC-based threshold decryption method from [ZZL25]

Additional desired properties in proposed cryptosystems

In addition to actively-secure threshold decryption (**Decrypt**), we are planning to explore

- the polynomial oracle ZKP approach from [HLSS25] for actively-secure distributed key generation (**KeyGen**),
- the Laminate approach [PLFT25] for verifiable computation (**Evaluate**).

We will decide later whether these modules can be included in the proposed cryptosystems.

All three capabilities will be developed in a modular way, e.g., it should be possible to instantiate one of the schemes using MPC threshold decryption without using actively-secure **KeyGen** and **Evaluate**.

Targeted threshold profiles

For proposed cryptosystems, we will consider both n -out-of- n and t -out-of- n scenarios.

We are planning to focus on threshold profiles (2), (3), (S), and (M) from the NIST MPTC call, i.e., the cases where n does not exceed 64.

In terms of corruption proportion, we will target dishonest majority (D) (n -out-of- n and $(n - 1)$ -out-of- n) and two flavors of honest majority (h) ($t > N/2$) and (H) ($t > 2N/3$).

Reference implementations

- We will develop a reference implementation for each proposed cryptosystem using at least one of the libraries maintained by the PANTHERIA team, i.e., OpenFHE, Lattigo, or Poulpy.
- The reference implementations will be compilable, executable, and testable on the Baseline Platform specified in Section 7.1 of the Final NIST-MPTC call.
- The actual implementations will be published in publicly-accessible github repositories.
- The reference implementations will be available under the open-source licenses approved by OSI, e.g., BSD 2-clause and Apache 2.0 licenses.

Performance evaluation methodology

- We will include performance results for conventional FHE cryptosystems (under passive security), as suggested in Section 11.5 of the Final NIST-MPTC call
- For threshold cryptosystems, we will typically report the runtime and size measurements w.r.t. conventional FHE cryptosystems
- We will also use the slowdown and size expansion w.r.t. conventional FHE cryptosystems as the main metrics to evaluate the costs of achieving active security for specific building blocks of threshold FHE cryptosystems,
 - e.g., the complexity of verifiable FHE evaluation will be compared to the complexity of passive-security FHE evaluation

Recent Work: *decomposed* BFV (dBFV) [PZZ'25]

- dBFV is a new “exact” FHE scheme that **operates on large (plaintext) numbers much more efficiently** than prior schemes like BGV, BFV, TFHE.
- In BGV/BFV, ciphertext **error grows proportionally to the plaintext modulus p** , which necessitates large parameters for security.
 - Ciphertexts and operations scale ***quadratically*** with the “bit length” $\log p$.
- In TFHE, large plaintexts are encrypted bit by bit \Rightarrow ***quadratic*** runtimes.
- dBFV encodes plaintexts in “**digit decomposed**” form, as **short vectors** (polynomials) modulo a special lattice.
 - Ring isomorphism induces modulo- p arithmetic
 - **Error grows proportionally to norm of short plaintext vector: $b * \log_b(p)$ versus p**
 - Ciphertexts and operations **scale only *~linearly*** with the bit length

Advantages of dBFV

1. **General:** works for **any plaintext modulus p** , with **any underlying LWE ring** (or none at all)
 - a. Cf. [GC14, CLPX18, GV25, BFGGMS25]: tight “coupling” between p and ring structure
2. **Fully Compatible** with standard FHE techniques and implementations: fast ring arithmetic, SIMD plaintext packing, bootstrapping, ...
3. **Flexibly parameterizable:** trade off error growth and size/speed via base b
4. **Fast:** prototype takes $\sim 10\text{-}100\text{ms}$ for $\text{mod-}2^{64}$ mult (vs. several sec for TFHE)

MPC-Based Threshold Decryption

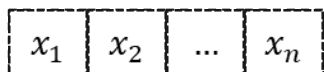
- In BGV/BFV, decryption is a “**rounded (mod-q) dot product**” with secret key:
$$\text{msg} = \lceil \langle c, s \rangle * (p/q) \rceil$$
- In threshold setting, parties have *linear shares* s_i of s . Could publish $\langle c, s_i \rangle$ and reconstruct $\langle c, s \rangle$ — but totally insecure! Reveals error in c .
- Typical solution: “**noise flooding**”. Publish $\langle c, s_i \rangle + E_i$, recon $\langle c, s \rangle + E$. Requires huge $E \Rightarrow$ huge $q \Rightarrow$ tiny “error rate” \Rightarrow **large params** for security.
- **Our solution** [ZZLP25]: *efficient* specialized MPC rounding protocol: Instead of *flooding* the noise, securely *remove* it before revealing anything!

Advantages of Our MPC-based Rounding

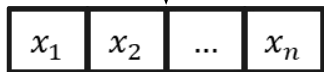
1. **Modular:** plugs into to any (exact) LWE/FHE, w/ no change to parameters
2. **Flexible and composable:** can use any UC-instantiation of the “ABB model”:
(pro)active security, (dis)honest majority, etc.
3. **Fast and high throughput:** offline *preprocessing* for fast, low-interaction *online* phase (once ciphertext is known).
Up to 20,000x better throughput and 37x better latency than prior SoTA.

Computation Outsourcing via FHE (Honest Server)

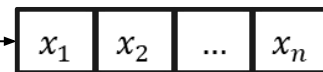
Client



FHE.Enc



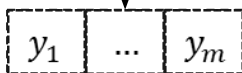
Server



FHE.Eval(C)



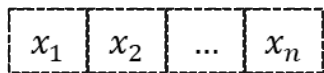
FHE.Dec



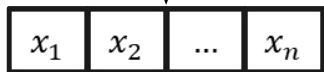
where $y_1, \dots, y_m = C(x_1, x_2, \dots, x_n)$

Computation Outsourcing via FHE (Malicious Server)

Client



FHE.Enc



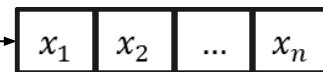
FHE.Dec



No Integrity!

- Wrong LLM models used
- Wrong transactions fetched
- Incorrect health reports

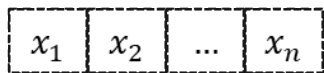
Server



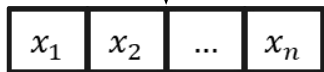
where $g_1, \dots, g_m \neq C(x_1, x_2, \dots, x_n)$ may just be some garbage values

Computation Outsourcing via FHE (Integrity Guarantee)

Client



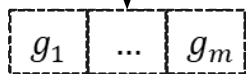
FHE.Enc



Verify



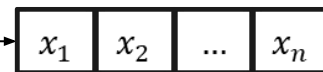
FHE.Dec



where $g_1, \dots, g_m = C(x_1, x_2, \dots, x_n)$ iff $\text{Verify}(\pi) = 1$

Active security
for homomorphic evaluations

Server



GenProof



Laminate: Adding Integrity to FHE [PLFT25]

Client



Encrypt

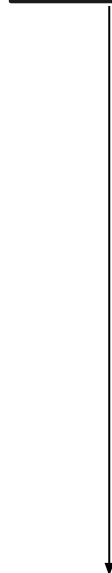


Input

Server



FHE.Eval(C)



Decrypt



Output

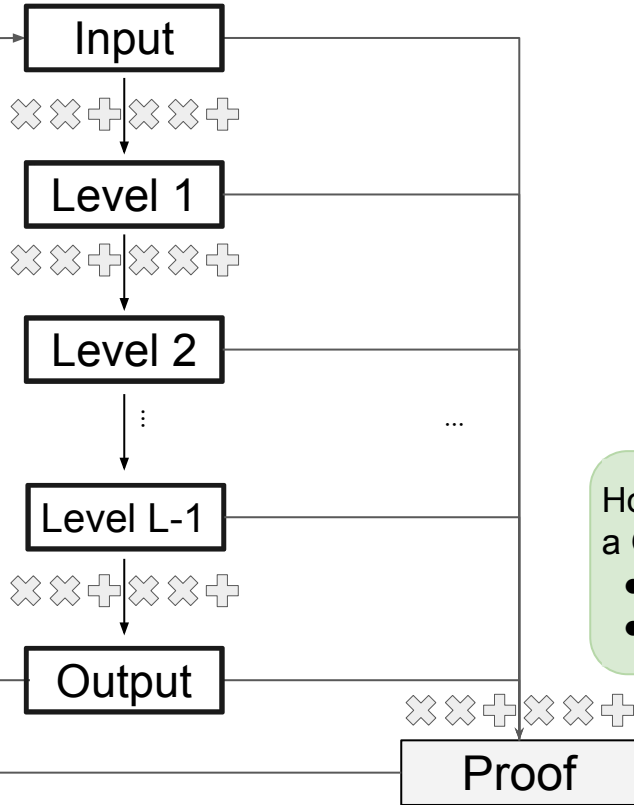
Laminate: Adding Integrity to FHE [PLFT25]

Client



Encrypt

Server



- Succinct verification

Homomorphically compute a GKR proof of integrity.

- 1 extra mult. level
- Quasilinear time

Verify

Decrypt

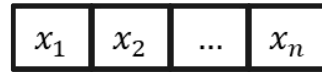
Proof

Laminate (A Very High-level Summary)

Server



Intermediate values
(including input/output)



FHE.Eval($\times / +$)



FHE.Eval($\times / +$)



...



FHE.Eval($\times / +$)



FHE performance is sensitive to the number of levels
(prior works require a lot more levels to generate the proof)

L levels of homomorphic circuits

GenProof
by FHE.Eval($C_{Gen} \pi$)



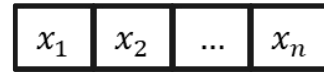
Laminate: only **one more** level
of homomorphic circuit

Laminate (A Very High-level Summary)

Server



Intermediate values
(including input/output)



FHE.Eval($\times / +$)



FHE.Eval($\times / +$)



...



FHE.Eval($\times / +$)



FHE performance is sensitive to the number of levels
(prior works require a lot more levels to generate the proof)

L levels of homomorphic circuits

GenProof
by FHE.Eval($C_{Gen} \pi$)



Verifier: decrypts and verifies the proof

Laminate (Performance Evaluation)

Circuit Type	8 Levels			20 Levels		
	Honest Eval (hrs)	Laminate Payload	Laminate Proving	Honest Eval (hrs)	Laminate Payload	Laminate Proving
<i>Gates distributed uniformly over all multiplicative depths</i>						
Half Adds/Mults	0.93	2.02×	81.4×	2.17	1.54×	34.9×
90% Mults	1.65	2.02×	45.9×	3.85	1.54×	19.7×
90% Adds	0.21	2.02×	360.5×	0.49	1.54×	154.5×
<i>90% of gates at the highest multiplicative depth</i>						
Half Adds/Mults	1.57	1.45×	48.2×	3.92	1.19×	19.3×
90% Mults	2.78	1.45×	27.2×	6.96	1.19×	10.9×
90% Adds	0.35	1.45×	216.3×	0.88	1.19×	86.0×

Evaluated over representative types of circuits:

- Payload overhead: from a single extra level, thus cheap $\approx 2x$
- Proving overhead: on the order of 10x to 100x
- Proof size: only 0.27MB
- Working on further improvement

References

- [BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. “(Leveled) Fully Homomorphic Encryption without Bootstrapping”. In: ACM Trans. Comput. Theory 6.3 (2014), 13:1–13:36. DOI: 10.1145/2633600. Also at ia.cr/2011/277.
- [Bra12] Zvika Brakerski. “Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP”. In: Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings. Ed. by Reihaneh Safavi-Naini and Ran Canetti. Vol. 7417. Lecture Notes in Computer Science. Springer, 2012, pp. 868–886. DOI: 10.1007/978-3-642-32009-5_50. Also at ia.cr/2012/078.
- [CGGI16] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. “Faster Fully Homomorphic Encryption: Bootstrapping in Less Than 0.1 Seconds”. In: Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Vol. 10031. Lecture Notes in Computer Science. 2016, pp. 3–33. DOI: 10.1007/978-3-662-53887-6_1. Also at ia.cr/2016/870.

References (cont'd)

[CKKS17] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yong Soo Song. “Homomorphic Encryption for Arithmetic of Approximate Numbers”. In: *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Vol. 10624. *Lecture Notes in Computer Science*. Springer, 2017, pp. 409–437. DOI: 10.1007/978-3-319-70694-8_15. Also at ia.cr/2016/421.

[DM15] Léo Ducas and Daniele Micciancio. “FHEW: Bootstrapping Homomorphic Encryption in Less Than a Second”. In: *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. *Lecture Notes in Computer Science*. Springer, 2015, pp. 617–640. DOI: 10.1007/978-3-662-46800-5_24. Also at ia.cr/2014/816.

[FV12] Junfeng Fan and Frederik Vercauteren. “Somewhat Practical Fully Homomorphic Encryption”. In: *IACR Cryptol. ePrint Arch.* (2012), p. 144. URL: <http://eprint.iacr.org/2012/144>.

References (cont'd)

[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. 2025, pp. 4049–4063. DOI: 10.1145/3719027.3765229. Also at ia.cr/2024/1879.

[PLFT25] Kabir Peshawaria, Zeyu Liu, Ben Fisch, and Eran Tromer. Laminate: Succinct SIMD-Friendly Verifiable FHE. Cryptology ePrint Archive, Paper 2025/2285. 2025. URL: <https://eprint.iacr.org/2025/2285>.

[PZZ25] Chris Peikert, Doron Zarchy, and Guy Zyskind. High-Precision Exact FHE Made Simple, General, and Fast. Cryptology ePrint Archive, Paper 2025/2321. 2025. URL: <https://eprint.iacr.org/2025/2321>.

[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. 2025, pp. 2339–2353. DOI: 10.1145/3719027.3744884. Also at: ia.cr/2025/1781.