Privacy-Enhancing Technologies: Challenges and Practical Solutions with FHE

White Paper

Tolun Tosun, Erkay Savaş

December 2025

Abstract

The use of big data has led to many breakthroughs in the Artificial Intelligence (AI) space. However, it also raised privacy concerns. The data being processed are often personal and sensitive. Privacy-Enhancing Technologies (PETs) help to keep data safe while still giving the intended service. Consequently, PETs play a critical role in complying with data protection regulations such as GDPR. PETs are developed under various trust models. Key challenges in those designs include preventing data leakage during computation, performance overhead, development complexity. Fully Homomorphic Encryption (FHE) stands out as a promising technology for enabling PETs. In this white paper, we evaluate FHE in the context of PETs, highlighting its strengths and discussing the open challenges that remain for its practical deployment.

1 Introduction

The growth of data-driven systems increased the need for protecting the privacy of individuals from whom the data are collected. Modern applications in healthcare, finance, cloud computing, machine learning utilize sensitive data that cannot be freely shared. For example, a lung x-ray data set collected in a hospital from the patients are highly sensitive information, while it is extremely valuable for developing statistical models for medical diagnosis. Classical data protection approaches such as encryption does not solve these issues. Because the data is typically decrypted for processing, so the data is exposed to the potential adversaries. For example, a cloud service provider which hosts the application can directly access data in plaintext form.

Privacy-Enhancing Technologies (PETs) address these challenges by enabling data processing while minimizing the leakage of sensitive information. PETs generally provide either cryptographic or statistical methods to reduce or eliminate the need for accessing plaintext data and relying on trusted third parties. A common statistical technique that is considered as a PET is data

anonymization [16]. Regulations such as GDPR impose stricter requirements on data handling. Consequently, practical PET constructions are getting increasing attention from both researchers and industry, and they are being considered for real-world deployment.

Among PETs, Fully Homomorphic Encryption (FHE) is one of the outstanding solutions. However, it is also a technically demanding approach. FHE is a special encryption technique, which enables arbitrary computation on encrypted data. The results of these computations remain in encrypted form that can only be decrypted by authorized parties. This capability immediately changes the trust model of outsourced computation, as the data owners can leverage external services such as cloud services, without revealing the content of their data which may hold sensitive information. However, despite significant advances in recent years, FHE continues to face notable challenges such as its performance.

This white paper discusses the key challenges that arise in privacy-preserving computation and evaluates FHE as a practical solution within the PET context. We discuss the security guarantees offered by FHE, analyze its limitations and deployment constraints.

2 PET Problem Space

The design and development of PETs depend on a set of fundamental challenges that arise when sensitive data must be processed beyond trust boundaries. These challenges are not cryptographic in nature. Indeed, they reflect a combination of security and privacy requirements, system constraints, and practical considerations related to the feasibility of the solutions solutions. A good understanding of this problem space is essential for evaluating the different PETs, including FHE.

2.1 Data Exposure During Computation

Traditional security mechanisms are effective for protecting data when it is stored or during its transmittion. However, they generally assume that computations are performed on plaintext data in a trusted environment. This assumption is unrealistic in modern scenarios such as cloud computing, outsourced data analysis, and collaborative processing between different organizations. For example, in AI tasks, training or running inference on machine learning models in the cloud often involves sensitive or proprietary data. In these scenarios, the classical symmetric key and public key cryptography protects the data during its transfer to the cloud server. However, the data becomes vulnerable since it is decrypted for processing.

2.2 Trust Assumptions and Threat Models

A key challenge in PET design is the definition of trust assumptions. Systems may include untrusted computation servers, semi-honest parties, adversaries

with malicious capabilities. A frequently used threat model is the *honest-but-curious* model [20]. In this model, the adversary is assumed to follow the given protocol correctly, while attempting to discover sensitive information by watching intermediate data. Considered threat model affects both the security guarantees and the performance overhead of a given solution. Consequently, PETs must be evaluated only on their theoretical security. Whether their trust assumptions align with real-world deployment scenarios should also be considered.

2.3 Performance and Scalability Constraints

Privacy-preserving computation usually introduces significant computational and communication overhead compared to plaintext processing, such as increased latency, memory usage and energy consumption, and reduced throughput. Because of these overheads, PETs can be impractical for certain applications such as large-scale or real-time ones. Scalability challenges are particularly considered in settings involving complex computations, large datasets, or multiple interacting parties. Performance considerations are often the primary bottleneck in the adoption of PETs, although the solutions achieve strong privacy guarantees.

2.4 Accuracy and Functional Effectiveness

Many applications need precise numerical calculations, complex control flow, or advanced machine learning operations. PETs can sometimes reduce the precision, limit the types of computations that can be done efficiently, require approximations or even changes how computations are performed. These limitations create a trade-off between privacy and accuracy. A stronger privacy is often achieved with lower accuracy. Differential privacy [9] is a well known example such trade-off. This approach protects privacy by perturbing the query results to the database by intentionally adding noise. The added noise is configured by a privacy parameter. Understanding these trade-offs is important to decide if a PET is suitable for a specific application or not.

2.5 Development Complexity

Beyond security and privacy feasibility, PETs must integrate with existing software stacks, and data pipelines. Processes such as key management, developer tools, and debugging support have a high impact on the practicality of deployment. Solutions, which require specialized expertise or extensive system redesign, may face barriers for adoption of them, particularly in industrial environments as reliability and maintainability are critical.

2.6 Evaluation Criteria for PETs

The challenges outlined above introduces a set of evaluation criteria for PETs:

- Trust assumptions: reliance on single or multiple parties, and whether these are semi-honest or malicious.
- Privacy: protection level against defined adversaries.
- Performance: throughput, latency, scalability, and also energy efficiency drawbacks.
- Accuracy: whether the results of computations are correct.
- Deployability: development and integration effort, development tools, and operational cost.

These criteria offer a way to analyze PETs and show that no one method provides a universal solution. In the next sections, we explore FHE with this perspective, focusing on its distinct advantages and its practical drawbacks in the wider PET context.

3 Fully Homomorphic Encryption (FHE)

Homomorphic Encryption (HE) enables computation on encrypted data. It produces encrypted outputs, which match the encryptions of the result of computations if they are performed on the corresponding plaintexts. This property of HE affects the trust model of computation by eliminating the need for exposing sensitive data to the party that performs the computation. As a result, HE is particularly well suited for outsourced computation, untrusted cloud environments, and privacy-preserving data analytics. *Fully* Homomorphic Encryption (FHE) supports arbitrary computation on the data while the types of computations in HE are limited. FHE offers some of the strongest privacy promises among existing PETs, as it ensures that data remains encrypted throughout computation and does not rely on trust in the compute provider. Its trust assumptions are typically minimal, involving a single untrusted server and a well-defined cryptographic adversary model.

A simplified model of encrypted data processing in a client–server setting is illustrated in Figure 1. In this model, client encrypts its input data and sends it to cloud server. Then, the cloud server performs the computationally intensive task directly on the encrypted data, without ever decrypting it. During the execution of this task, homomorphic evaluation keys are used which are a special type of public keys in FHE context. The cloud returns the resulting output to the client in ciphertext form. This model is also referred to as end-to-end security, as the data remains encrypted through both transmission and processing. It is only decrypted by the client, who is considered as the owner of the input and output data. As a result, the PET concern discussed in Section 2.1 is naturally addressed.

FHE was considered as the holy grail of cryptography. The first FHE construction was introduced by Craig Gentry in his PhD thesis [11]. Despite the previous HE schemes, Gentry's solution was capable of supporting arbitrary

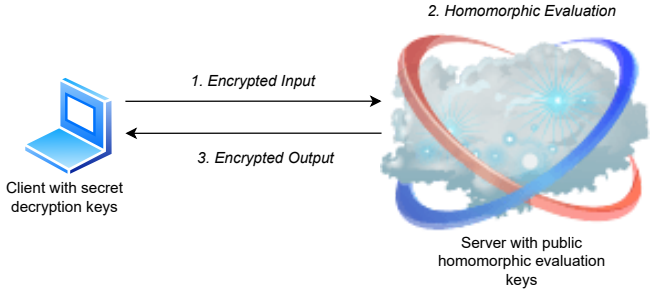

Figure 1: Simplified encrypted data processing model using homomorphic encryption.

computation over encrypted data. Gentry's approach achieved that by combining the so-called somewhat homomorphic encryption with a bootstrapping procedure to reduce accumulated noise. As a result, evaluation of circuits with unbounded multiplicative depth became possible while preserving correctness and security.

Gentry's original construction was mainly of theoretical interest as its computational overhead was extreme. However, it established the feasibility of FHE and led to extensive research aimed at improving efficiency, security assumptions, and practicality of FHE. Subsequent schemes shifted toward more efficient lattice-based constructions, utilizing hardness of Learning With Errors (LWE) or Ring-LWE.

In todays world, the most widely used schemes are Brakerski–Fan–Vercauteren (BFV) [10, 4], BGV [5], CKKS (Cheon–Kim–Kim–Song) [12], and TFHE (Fast Fully Homomorphic Encryption over the Torus) [7]. The BFV scheme supports exact arithmetic over integers and is well suited for applications requiring precise computation, such as encrypted database queries [8, 15] and integer-based analytics.

For applications involving real-valued or approximate computation, the CKKS scheme is a dominant option. CKKS supports approximate arithmetic on encrypted floating-point–like values, allowing efficient evaluation of linear algebra operations with controlled numerical error. This design makes CKKS particularly attractive for privacy-preserving neural network inference and signal processing tasks, where exact integer arithmetic is not needed.

In contrast to arithmetic-circuit–oriented schemes such as BFV, BGV, and CKKS, TFHE focuses on efficient evaluation of Boolean gates with fast bootstrapping. TFHE enables low-latency evaluation of binary circuits by performing bootstrapping at the level of individual gates. Consequently, it is well suited for applications involving control logic, comparisons, and branching. While TFHE is an outstanding solution in Boolean algebra and latency for small circuits, it is generally less efficient for large-scale numerical computation compared to arithmetic HE schemes.

4 Open Challenges & Research Gaps

Despite the substantial progress in the design and implementation of FHE, several open challenges continue to limit its widespread adoption. The most important limitation of FHE is its high computational cost relative to plaintext computation. In particular, FHE-based data processing can introduce overheads of up to six orders of magnitude compared to plaintext processing [6]. This overhead arises primarily from data expansion of ciphertexts, expensive polynomial arithmetic, and noise management operations such as bootstrapping. Although advances such as batching, and optimized bootstrapping have significantly improved performance, the evaluation of deep or complex circuits remains computationally demanding.

The impact of computational overhead depends on the application domain. While such overhead may be a blocker for real-time applications, it can be tolerable in offline or batch-processing settings. For example, in the context of AI, inference is typically a real-time operation where the latency is critical. However, model training is generally performed offline and can therefore accommodate relatively higher computational delays.

To address practical issues in FHE, existing applications are usually adapted to the constraints of FHE-friendly arithmetic. Developers usually redesign algorithms to reduce multiplicative depth, avoid control flow branching, and operate within restricted numeric domains, such as [21]. While compilers and high-level frameworks, such as [22], aims to automate parts of this process, there is still a gap between cryptographic primitives and efficient, general-purpose programming models for FHE. Bridging this gap requires further research for structuring computational tasks to minimize homomorphic cost.

Another asset for tackling the practical issues is hardware acceleration. Existing studies on hardware acceleration for Fully Homomorphic Encryption are already rich. A wide range of acceleration approaches have been explored, including the use of GPUs, FPGAs (field programmable gate arrays), and custom hardware accelerators through ASICs (application specific integrated circuits) [17, 2], with examples [24, 19], [1, 13, 23], and [18, 17, 2], respectively.

Energy consumption is another issue in practical FHE solutions, which is closely related to computational overhead. Aligned with performance considerations, FHE-based systems consume substantially more energy than their plaintext counterparts. This fact raises concerns about scalability, cost, and also environmental impact. In cloud and data-center environments, energy efficiency is a direct operational expense. There is a notable research gap in systematic evaluation of energy-per-operation metrics for FHE and in the development of energy-aware parameter selection, scheduling, and hardware–software co-optimization strategies. Addressing energy consumption will be critical for making FHE viable beyond niche or high-value use cases.Key management is another fundamental challenge in FHE deployments.

Managing the keys securely across distributed systems, and multiple users, introduces significant complexity. Public evaluation keys can reach tens of gigabytes in size for certain operations, such as bootstrapping. From a sys-

tems perspective, the absence of well-defined and standardized key management frameworks represents a critical gap between cryptographic theory and practical, deployable solutions. More particularly, in multi-party computation settings, the mathematical structure of public and secret keys are revised, leading to approaches such as threshold FHE [3]. While these methods address some trust concerns, they often require a trusted third party for key generation and distribution. Therefore, they introduce additional trust assumptions that are undesirable in many deployment scenarios. A well known application in the multi-party FHE setting is federated learning [14].

5 Conclusion

We formalized the PET problem space and discussed the strengths of FHE together with open challenges. Among PETs, FHE provides exceptional privacy guarantees and minimal trust requirements, enabling end-to-end security.However, these advantages co-exist with practical challenges, including notable computational and memory overheads, leading to increased latency and energy consumption. Overall, FHE is an outstanding option for use cases where the associated performance and deployment costs can be tolerated in order to leverage its strong privacy guarantees.

6 Acknowledgments

This document was prepared in the scope of the European Union Twinning Project 101079319 (acronym enCRYPTON).

This document was prepared with the assistance of OpenAI's ChatGPT for language refinement and structural editing. The authors take full responsibility for the technical content and conclusions.

References

[1] Agrawal, R., de Castro, L., Yang, G., Juvekar, C., Yazicigil, R., Chandrakasan, A., Vaikuntanathan, V., Joshi, A.: Fab: An fpga-based accelerator for bootstrappable fully homomorphic encryption. In: 2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA). pp. 882–895. IEEE (2023)

[2] Aikata, A., Mert, A.C., Kwon, S., Deryabin, M., Roy, S.S.: Reed: Chiplet-based accelerator for fully homomorphic encryption. arXiv preprint arXiv:2308.02885 (2023)

[3] Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold fhe. In: Annual International Conference on

the Theory and Applications of Cryptographic Techniques. pp. 483–501. Springer (2012)

[4] Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapsvp. In: Annual cryptology conference. pp. 868–886. Springer (2012)

[5] Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory (TOCT) **6**(3), 1–36 (2014)

[6] Brynds, C., McLeod, P., Caccamise, L., Pal, A., Saiham, D., Rahman, S., Miguel, J.S., Wu, D.: Cryptoracle: A modular framework to characterize fully homomorphic encryption. arXiv preprint arXiv:2510.03565 (2025)

[7] Chilotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast fully homomorphic encryption over the torus. In: Journal of Cryptology. Springer (2019)

[8] Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. Journal of the ACM (JACM) **45**(6), 965–981 (1998)

[9] Dwork, C.: Differential privacy. In: International colloquium on automata, languages, and programming. pp. 1–12. Springer (2006)

[10] Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive (2012)

[11] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the forty-first annual ACM symposium on Theory of computing. pp. 169–178 (2009)

[12] J. H. Cheon, A. Kim, M.K., Song., Y.: Homomorphic encryption for arithmetic of approximate numbers. In: Asiacrypt 2017. pp. 409–437. Springer (2017)

[13] Koçer, E., Kirbiyik, S., Tosun, T., Alaybeyoglu, E., Savas, E.: Io-optimized design-time configurable negacyclic seven-step ntt architecture for fhe applications. In: Proceedings of the Great Lakes Symposium on VLSI 2025. pp. 14–21 (2025)

[14] Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: Challenges, methods, and future directions. IEEE signal processing magazine **37**(3), 50–60 (2020)

[15] Melchor, C.A., Barrier, J., Fousse, L., Killijian, M.O.: Xpir: Private information retrieval for everyone. Proceedings on Privacy Enhancing Technologies pp. 155–174 (2016)

[16] Murthy, S., Bakar, A.A., Rahim, F.A., Ramli, R.: A comparative study of data anonymization techniques. In: 2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS). pp. 306–309. IEEE (2019)

[17] Nabeel, M., Gamil, H., Soni, D., Ashraf, M., Gebremichael, M.A., Chielle, E., Karri, R., Sanduleanu, M., Maniatakos, M.: Silicon-proven asic design for the polynomial operations of fully homomorphic encryption. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **43**(6), 1924–1928 (2024)

[18] Ovichinnikov, D., Kavadia, H., Kudupudi, S.K.C., Rempel, I., Chadha, V., Franz, M., Master, P., Gentry, C., Kindler, D., Reyes, A., et al.: Resource estimation of cggi and ckks scheme workloads on fractlcore computing fabric. arXiv preprint arXiv:2510.16025 (2025)

[19] Özcan, A.Ş., Ayduman, C., Türkoğlu, E.R., Savaş, E.: Homomorphic encryption on gpu. IEEE Access **11**, 84168–84186 (2023)

[20] Paverd, A., Martin, A., Brown, I.: Modelling and automatically analysing privacy properties for honest-but-curious adversaries. Tech. Rep (2014)

[21] Song, C., Shi, X.: Reacthe: A homomorphic encryption friendly deep neural network for privacy-preserving biomedical prediction. Smart Health **32**, 100469 (2024)

[22] Stoian, A., Frery, J., Bredehoft, R., Montero, L., Kherfallah, C., Chevallier-Mames, B.: Deep neural networks for encrypted inference with tfhe. In: International Symposium on Cyber Security, Cryptology, and Machine Learning. pp. 493–500. Springer (2023)

[23] Su, Y., Yang, B., Yang, C., Tian, L.: Fpga-based hardware accelerator for leveled ring-lwe fully homomorphic encryption. IEEE Access **8**, 168008–168025 (2020)

[24] Wang, W., Hu, Y., Chen, L., Huang, X., Sunar, B.: Accelerating fully homomorphic encryption using gpu. In: 2012 IEEE conference on high performance extreme computing. pp. 1–5. IEEE (2012)