Computing with Fully Homomorphic Encryption

Constructions, Characteristics and Comparisons

Alexandru Eugen BULBOACA[1]
Supervisor: Lilika MARKATOU[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Alexandru Eugen BULBOACA
Final project course: CSE3000 Research Project
Thesis committee: Lilika MARKATOU, Tim COOPMANS

Abstract

In the digital economy, individual data sovereignty is a requirement for sustainable and secure development. With an increase in the number of outsourced computations, due to the adoption of web services, artificial intelligence and research-based innovation, privacy became one of the main concerns for individuals and third parties alike. Fully Homomorphic Encryption (FHE) addresses this gap by enabling secure computations on encrypted data. This paper reviews and presents the FHE literature in a structural manner, covering the topics of mathematical constructions, security, efficiency and functionality. In doing so, it highlights the current state-of-the-art techniques and provides a systemic comparison with other privacy enhancing mechanisms. In addition, it discusses its applicability in the practical domain, such as Confidential Machine Learning, Medical Data Analysis and Recommender Systems, mentioning potential impediments and their solutions for mass adoption.

1 Introduction

Privacy is one of the most prominent concerns when it comes to storing, sharing and processing data. To protect against unauthorized access, encryption is used to render the data unrecognizable during storage and transfers. However, in order to process it, the data first needs to be decrypted, and then used, leaving it exposed during computations.

The earliest formulation of functions enabling operations on encrypted data dates back to 1978, under the name of *privacy homomorphism* [1]. Until 2009, only functions with limited homomorphic capabilities were discovered. These were called Partially Homomorphic Encryption (PHE), as they allowed only one operation to be performed - either addition [2–4] or multiplications [5,6]. Attempts were made to extend this functionality to include both operations [7], which led to the creation Somewhat Homomorphic Encryption (SwHE) schemes. These were still impractical, as they only allowed a limited number of operations to be performed.

Craig Gentry developed the first theoretical construction of a Fully Homomorphic Encryption (FHE) scheme [8], through a process called *bootstrapping* that transforms a SwHE into a FHE. This enables an arbitrary number of simple operations to be performed without decrypting first, which ensures no information leakage occurs in the process [9]. It was a monumental breakthrough in the field of cryptography, as FHE has numerous applications in various fields [10]. Encrypted information forms a key part of scientific projects that involve human health [11], cloud computing [12], and economic, political and social phenomena [13].

The implications of FHE reach beyond theoretical computer science. Its functionality enables secure delegation of computation of sensitive data and to untrusted environments, such as cloud servers. However, usability remains a challenge due to *high resource demands*, *lack of interoperability*, and *slow computation*. As for *integration*, while progress has been made, real-world deployments remain constrained to specific, low-depth applications. Nonetheless, FHE research is promising for privacy-preserving analytics, encrypted machine learning, and multi-party (MKFHE) [14] computation.

This paper will present a literature review on the topic of *Fully Homomorphic Encryption* schemes. It will examine the best-known construction mechanisms, their characteristics, and a comparison with other privacy-enhancing techniques.

2 Preliminaries

2.1 Background

The technological gap that Homomorphic Encryption (HE) covers is the ability to delegate computation to untrusted environments, while maintaining the security and integrity of the data. This system consists of two parties: the user who uploads the encrypted data and the server, which executes the computations and manages the encrypted data. Compared to other privacy-enhancing technique, HE does not require multiple communications, shared states, nor support the use of data encrypted under different keys. Instead, it facilitates the server to compute any given function without decrypting in the process, or knowing the secret key.

A central concept in HE is *noise* - a small random *error* intentionally added during encryption to ensure semantic security. Over the course of multiple additions and multiplications with ciphertexts, this noise accumulates. When it is small, it can be removed in the decryption process using some prior knowledge of the hidden "error correcting codes". Otherwise, if it grows beyond a certain threshold, the decryption will return the wrong plaintext or fail altogether. Many resources were dedicated to the study of noise growth [15], since it determines the use cases and effectiveness of the schemes.

Craig Gentry introduced the first theoretical method of *refreshing* the accumulated noise of a SwHE scheme [8], which he later adapted to make it practical [16]. This breakthrough was facilitated by introducing the concept of *bootstrapping*, turning a *depth bounded* [17] HE scheme into a FHE scheme.

The construction of a FHE, based on Gentry's framework, includes two major steps. First, a SwHE is designed to perform a limited number of operations homomorphically. Initial works [8, 16] had decryption circuits larger than their respective SwHE schemes could safely compute. As such, for a short period of time, an additional step was added to the framework - *squashing* - which was used to shorten the decryption circuit up to a manageable depth, which resulted in a "bootstrappable" SwHE scheme. Finally, apply bootstrapping to transform the "bootstrappable" SwHE scheme into a FHE, by reducing the noise to its baseline level.

Any HE construction has 2 fundamental properties: *circuit privacy* and *multi-hop homomorphism*. The *circuit privacy* property guarantees that the cloud server inputs, such as the function that needs to be computed, do not leak information, while the user is only allowed to know few elements about them. Whereas *multi-hop homomorphism* permits the output of the evaluation function to be use by another homomorphic evaluation function [9] for a virtually unlimited number of times. This property refers to the ciphertext bit-length expansion with each operation [18]. A scheme which presents an error growth independent of the degree of the function under evaluation is called *compact*. Generally, FHE schemes are compact, while SwHE schemes may not be.

Following Gentry's breakthrough, numerous constructions have improved efficiency, security, and applicability. Based on the taxonomy of FHE schemes presented by Chris Peikert [19], there are 4 generations of schemes, using different underlying hardness assumptions, such as: *Ideal Lattices*, *Approximate Greatest Common Divisor (AGCD)*, *Learning with Errors (LWE)* and *Ring-LWE (RLWE)*. Each has their State-Of-The-Art (SOTA) constructions, which are: BGV [20], B/FV [21,22], FHEW [23], TFHE [24] and CKKS [25].

2.2 Methodology

The central question of the research paper is *"What are the State-Of-The-Art Fully Homomorphic Encryption schemes and their computational characteristics?"*. Subsequent guiding questions can be found in Appendix C.2. In order to address these questions, the topic of FHE was analyzed on a 3-axis plane: background, technical aspects, characteristics and contextualized in a comparison with other privacy-enhancing techniques. For the comparison, Tables 4 and 5 were co-created by the collective knowledge of fellow students, under the supervision of Assoc. Prof. PhD. Lilika Markatou.

The paper has followed the structure of a literature review. The central reference Craig Gentry's "Fully Homomorphic Encryption using Ideal Lattices" [8] was followed by the representative publications for each generation of FHE schemes, as per the taxonomy of Chris Peikert [19]. The research technique called *forward reference searching* was used to identify publications that cite the original articles. Auxiliary materials were found using search engines, such as Google Scholar and Scopus, by applying Queries and Filters, such as in Appendix C.1. The resulting papers have be ranked based on the number of citations and their influence on subsequent works.

3 Technologies

This section displays a comprehensive account of the constructions of the most relevant FHE schemes. These will be discussed in relation to each other and to their respective generation design.

Most FHE constructions are naturally based on *asymmetric* (public key) encryptions, given the necessity of proper integration with real-world environments and lower complexity of key management. Alternative constructions, purely based on symmetric key mechanisms have been proposed [26–28], along with a way to convert a symmetric key to a public key scheme [29], but many suffer form security design flaws [30]. Moreover, most relevant constructions to date are *probabilistic*, highlighting the non-deterministic properties of the encryption function. As such, encrypting the same plaintext twice will not return the same ciphertexts, which accredits them *semantic security*.

Ideal Lattices

Many cryptographic systems use a mathematical concept called *lattice* as a basis for their security. Specifically, cryptographic constructions leverage the *hardness*[1] properties of *ideal lattices* based problems. Broadly speaking, ideal lattices are a set of infinitely repeating points in a n-dimensional space, represented by the set of all integer combinations of n linearly independent vectors.

The most common hard lattice problem is the *shortest vector problem (SVP)*. The best known algorithm for solving such lattice search problems is the *LLL*-algorithm, which achieves a complexity of an approximation factor of $2^{O(n)}$ [31], where n is the dimension of the lattice [32]. The conjecture that there is no non-quantum (perhaps even quantum) polynomial time algorithm that solves this kind of problems [33] confers the basis for the security analysis of many cryptographic systems. The research publications of Micciancio, Goldwasser and Regev cover analysis of *Lattice-based Cryptography* [32,34,35] in great depth.

Following this conceptualization, the first generation FHE constructions were mainly "based on ad-hoc average case assumptions about ideal lattices" [19], such as Sparse Subset Sum Problem (SSSP) [36], Polynomial Coset Problem (PCP) [37], Ideal-Shortest Independent Vector Problem (I-SIVP) [38], Bounded Distance Decoding Problem (BDDP) [39]. The nature of these schemes required high computational efficiency and large ciphertext sizes, which led to bottlenecks in bandwidth required to transfer the ciphertexts [40]. Most relevant subsequent constructions (Generation 2 to 4) leveraged the *(Ring) Learning with Errors* hardness properties of Lattices to address these issues.

(Ring) Learning With Errors

Oded Regev first introduced the concept of Learning with Errors (LWE) in 2005 as an "extension of the *learning from parity with error* problem" [33], introduced by Blum, Furst, Keans and Lipton [41]. Informally, LWE problems ask to recover a secret $s \in \mathbb{Z}_q^n$ given a set of random, approximate (*noisy*) linear equations on s [42]. Regev demonstrated that LWE assumes similar hardness assumptions as worst-case lattice problems, such as *GapSVP* (the decisional version of SVP) and *SiVP* [33,43], by using a *quantum*[2] reduction. In turn, this implied that solving the LWE problem in polynomial time, leads to a solution of the SVP-like problems in *quantum* polynomial time[3]. This ensured the security of the LWE based schemes, and marked the beginning of a new generation of FHE schemes. Later the concept of Ring-LWE (RLWE) has been introduced by Stehlé under the name of *Ideal-LWE* [45], and further defined by Lyubashevsky [46] under the current notation. RLWE is an algebraic version of LWE, which manifests the same strong security guarantees (no attack so far leveraged the "ring" property), while being more efficient for applications.

[1]Hardness refers to the difficulty of finding an efficient - polynomial time algorithmic complexity solution to a problem.

[2]This reduction assumes that the attacker has quantum computing capabilities, which are more powerful than classical adversaries.

[3]Refers to the time complexity of algorithms that can be executed by a quantum computer in polynomial time with a bounded error probability ($<$1/3) (BQP). Quantum algorithms, such as Shor's [44], can break many classical schemes, while lattice-based assumptions like LWE remain hard even for quantum computers.

Prerequisites

The central feature of cryptographic scheme is the *security parameter* λ. Complexity-wise all the functions, the noise growth, and the length of the secret values can be expressed in terms of λ. In order for the scheme to be considered *provably secure*, the probability of a successful attack (i.e. scheme failure) should be *negligible* in the security parameter: $P(failure) < \frac{1}{poly(\lambda)}$. This implies that as λ increases, the probability of a successful attack decreases faster than the inverse of any polynomial, making the attack computationally infeasible. However, with the security increase, so does the complexity of all the other HE functions.

A HE scheme consists of 3 probabilistic algorithms [8,18]:

- Key Generation - Gen: For a given security parameter λ, outputs a pair of public - private keys: pk, sk and the (public) evaluation key evk which is needed for the error reduction procedure
- Encrypt - Enc: Takes the public key pk and a message m (from the message space), and outputs a ciphertext c
- Decrypt - Dec: Takes the secret key sk and a ciphertext c, and outputs the message or a special character in case of failure

Additionally, any FHE construction requires a general purpose interface for applying functions (i.e. an arithmetic or boolean circuit of *permitted length* [8]) to a set of ciphertexts:

- Evaluation - $Eval$: Takes as input the evaluation key evk, a function f and a tuple of ciphertexts $c_1, ...c_t$. The output of this function should decrypt to the output of the function applied to the set of plaintexts, specifically: $Dec_{sk}(Eval_{evk}(f, (c_1, ...c_t))) = f(m_1, ...m_t)$.

It's worth mentioning that while $Enc_{pk}(f(m1, ...m_t))$ and $Eval_{evk}(f, (c_1...c_t))$ both decrypt to the same message, they are not equal - their constructions and noise levels differ [18]. This algorithm is the core construction behind *bootstrapping*.

Bootstrapping

Based Gentry's framework, bootstrapping is a special use case of $Eval$, which "evaluates the decryption circuit homomorphically" [8,16]. As such, the function used in this interface is the decryption circuit, along with an encryption of the secret key: $c_f = Eval_{evk}(Dec_{Enc_{pk}(sk)}, (c_1...c_t))$.

Fundamentally, the bootstrapping procedure uses $Eval$ interface to Decrypt and Recrypt simultaneously, which produces a new (fresh) ciphertext and doesn't expose any data in the process [18]. However, this approach has 2 underlying assumptions. Firstly, that the Dec circuit has a depth small enough so that it can be evaluated homomorphically. This was addressed in the initial constructions through the *squashing* mechanism, which is used before bootstrapping, to express Dec as a function with a lower degree (decreasing the circuit depth). This was made possible by adding "extra information" [8] about the secret key sk to the evaluation key $eval$, without compromising security[4]. Secondly, *circular security* - that the secret key can be safely encrypted under the public key. Most constructions also implicitly assume this as a requirement for bootstrapping [20].

[4]An additional assumption was made to maintain security, namely the hardness of the average-case SSSP

3.1 First Generation

Based on Ideal Lattices

First generation FHE schemes debuted with Gentry's framework [8, 16] based on Ideal Lattice assumptions. Initially it was "conceptually and practically not a realistic scheme" [47], given the complexity of the bootstrapping procedure. Nevertheless, it was first implemented by Smart and Vercauteren [48]. They proposed slight modifications to the encryption Enc function that produced smaller key sizes, "whilst ciphertexts are the same size". Noteworthy is the introduction of the *Single Instruction Multiple Data Processing (SIMD)* in this implementation, which was achieved by using the *Chinese Remainder Theorem (CRT)*[5]. This technique set the foundation of *batching*, which permits packing multiple plaintext to be encrypted in the same ciphertext. It significantly reduced the time for processing sequences of messages, by allowing parallel computation, at a cost of a highly complex and slow key generation process.

Meanwhile, Stehlé and Steinfeld reduced the complexity of the refreshing procedure, by reducing the number of vectors in the hard problem assumption instance and introducing "a probabilistic decryption algorithm that can be implemented with an algebraic circuit of low multiplicative degree" [49]. A subsequent improvement was later made by Gentry and Halevi [50], who relied on the previous construction concepts [8, 16, 48, 49] to reduce the complexity of the $KeyGen$ method, and expand on the SIMD batch encryption processing. While naturally this key generation construction doesn't present the same batching capabilities, these can be preserved by selecting a specific set of parameters. Smart and Vercauteren explored this topic further in another dedicated publication [51].

All of the First Generation constructions are based on *principal ideals* that have "*relatively short* generators, which serve as secret keys" [52]. As such, their "hardness" and security is based on efficiently finding one such *short generator*. This is known as *Short Generator of a Principal Ideal Problem (SG-PIP)* and it was broadly considered a hard problem. However, more recent works initiated by Dan Bernstein's attack on the Campbell-Groves-Shepherd *SOLILOQUY* asymmetrical cryptosystem [53] have given shape to potentially efficient attacks against this specific problem. A concrete proof of such attacks was later provided [52], which, coupled with the difficulty to implement these mathematical constructions, had marked the end of the First Generation of FHE schemes based Ideal Lattices. Specifically, they provided a key-recovery attack in *quantum polynomial time* or exponential classical time ($2^{n^{\frac{2}{3}-\epsilon}}$) for finding "short generators" [18].

[5]CRT states that given k divisors (or moduli) $n_1, ...n_k$ and k remainders $a_1, ...a_k$ with $a_i < n_i$, there is only one number (x) smaller than the product of the divisors $N = n_1 \cdot ... \cdot n_k$ with the property that it presents the remainder a_t to the division with n_t: $x \equiv a_t \pmod{n_t}$, for any $t \leq k$

Based on Approximate Greatest Common Divisor

A significant contribution to the first generation of FHE was the DGHV scheme [54] due to its simplicity and applicability to integers. The underlying security assumptions are based on the hardness of the SSSP and *Approximate Greatest Common Divisor*[6] *(AGCD)* [55] problems. Initially the proposed DGHV scheme was symmetric, but the authors also presented the way to convert it to asymmetric, by computing many "encryptions of zero", which are shared as a public key. Then, encrypting relies on adding to the message a subset sum of these encryptions. To achieve a FHE, the bootstrapping method was coupled with the public key DGHV's SwHE construction:

- $KeyGen$: based on λ output the secret key p and the public key $(x_0, ...x_n)$, such that $x_i = p \cdot q_i + r_i$, where $x_0 > x_i \, \forall i > 0$ and q_i, r_i are random integers
- Enc : for a message $m \in \{0, 1\}$, create the ciphertext $c = m + 2r + 2\sum_{i \in S} x_i$, where r is a random integer and S is a subset of $\{1, ..n\}$
- Dec : for a ciphertext c compute $(c \, mod \, p) \, mod \, 2$. The $c \, mod \, p$ eliminates the multiples of p from x_i, leaving $m + 2 * noise$, which is reduced $mod \, 2$, as long as $2r + 2\sum_{i \in S} x_i < p/2$

Despite the simplicity of the scheme, it presents large key sizes ($\mathcal{O}(\lambda^{10})$) and high computational costs. As such, subsequent publications have addressed the issue of public key sizes: Coron et al. reduced it to $\mathcal{O}(\lambda^7)$ [56], then again to $\mathcal{O}(\lambda^5)$ [57] and Yang et al. brought it to $\mathcal{O}(\lambda^3)$ [58]. Their approach was based on the idea that only a subset of elements of the public key should be stored, since the rest can be safely recovered. However, Chen et al. [59] presented an new approach to the *exhaustive search attack* over the hardness of AGCD, "whose running time is essentially the "square root" of that", which reduced the security of the scheme.

This scheme was further improved by research addressed in the Section 3.2. For example, it's worth noting that Coron et al. [57] additionally used the *modulo switching* technique, introduced by Brakerski and Vaikuntanathan [60]. This further reduced the key size by scaling the equation with a fraction composed of the new smaller modulus base p' divided by the secret key: $\frac{p'}{p}$, implicitly scaling down the noise as well. Moreover, a more recent publication [61] has leveraged Brakerski's [22] *scale-invariant* property to address the limitations of the modulus switching technique. Additionally, batch versions of DGHV were independently introduced [62, 63] and merged in a final version [64] to handle multiple ciphertexts and re-encryption operations in parallel [47].

3.2 Second Generation

Brakerski and Vaikuntanathan developed a couple of SwHE scheme based on LWE [60] and RLWE [65], which marked the starting point of the Second Generation of schemes [19]. They had introduced two central concepts: *re-linearization* (known as *key switching*) and *dimension-modulus reduction* (known as *mod switching*). Given the n-dimensional "random" vectors representation of the secret key, homomorphic operations introduced the problem of exponential growth of the dimension of the ciphertexts: a single multiplication grows the ciphertext size to $\frac{n^2}{2}$. *Re-linearization* addresses this problem: reducing the quadratic equation $c_1 \cdot c_2$ to a linear one by means of encrypting all the terms of the secret key under a new key [18], scaling the size back to $n + 1$. *Mod switching* replaced *squashing*[7] with a technique that mathematically scales down the complexity (depth) of the decryption circuit (degree of at least $max(n, \log(q))$ [60]) from polynomial in the bits of the secret key ($poly(\lambda)$), by some polynomial factor $poly(n)$. Simultaneously, this technique also reduces the *absolute error* to some fixed polynomial bound [19, 60].

The key and mod switching techniques were highly utilized in the construction of the BGV [20] scheme (RLWE). It was constructed as a *Leveled*-FHE, which displays homomorphic capabilities up to any predefined depth. As such, it removes the costly bootstrapping procedure, and is much more efficient ($\mathcal{O}(\lambda \cdot d^3)$) in circuits that have a depth d known apriori. The authors also presented a faster bootstrapping technique($\mathcal{O}(\lambda^2)$) that can transform it into a FHE when necessary. This scheme is highly used in the industry, having some of its optimizations been implemented in the IBM HElib [66]. The most relevant optimization [67] marked an important step towards fast bootstrapping, and used Smart-Vercauteren [48, 51] batch processing on the BGV [20] construction to achieve one of the fastest schemes, with polylogarithmic overhead ($polylog(\lambda)$) for each gate. The BGV [18, 20] scheme design can be found in Appendix D.

Based on this technique, Brakerski [22] proposed a *scale invariant* LWE scheme, which removes the need for modulus switching. In this approach, the error growth rate is bound at a $poly(n)$ rate per multiplication. Fan and Vercauteren (FV) implemented this scheme [22] over RLWE [21], optimizing the computation and the key switching approach. These two constructions, referred to as B/FV, are implemented in Microsoft's Simple Encrypted Arithmetic Library (SEAL) [68]. Later publications have presented optimizations to the bootstrapping procedure for large moduli [69], modifications to the plaintext space [70], and to the modulus used [71]. Regardless, both B/FV and BGV are considered SOTA schemes in the context of the Second Generation. These have been studied in comparison, especially how they manage error growth. Given the fact that decryption is only possible if the "small error" remains smaller than the modulus q, the representations of BGV-B/FV-inspired constructions are faced with a security - error margin tradeoff that influences parameter selection [18]. Performance-wise, Costache and Smart [72] found that for a fixed number of levels and small plaintext B/FV is preferable to BGV, whereas the latter outperforms all the other schemes as the plaintext modulus increases. Subsequently, Costache et al. [15] have aug-

[6] AGCD asks to determine a secret integer p from a set of "noisy" equations of the form $x_i = p * q_i + r_i$, with given conditions. Specifically, p is a η bit prime, x_i have γ bits, r_i have ρ bits and ρ significantly smaller than η [55].

[7] Implicitly removed the need for an additional SSSP hardness assumption

mented this comparison with a noise growth analysis. They concluded that there is a discrepancy between the theoretical (assumed) and practical (observed) noise growth, in addition to the fact that for large moduli BGV outperforms B/FV.

3.3 Third Generation

Gentry, Sahai and Waters (GSW) [73] proposed a scheme based on LWE called *the approximate eigenvector* method. The scheme has no evaluation key and knowing the public key is no longer required, except for basic parameters. In this approach additions and multiplications are treated as matrix operations, resulting in an "asymptotically faster" scheme. It removed the need for key and modulus switching, further reducing the complexity of the operations. Due to this fact, GSW presents a *polynomial* ($L\ poly(n)$) growth in error rate rather than the *quasi-polynomial* ($poly(n)^{log(L)}$) that relinearization-based schemes exhibit [74] for a maximum multiplicative depth L and a scheme dimension n. Moreover, it reduced the space complexity from quasi-cubic to quasi-quadratic, although the authors state that the RLWE version performs worse than the best known RLWE schemes by a log factor. Additionally, it is worth noting that the main drawback of this method, as stated by the authors is the costly bootstrapping procedure: "our scheme loses some of its advantages once bootstrapping is used" [73]. Following the notation [75, 76], the construction of GSW [18, 73]:

- $KeyGen$: outputs random $sk = (1, s_2, ...s_n) \in \mathbb{Z}_q^n$, and $pk = A \in \mathbb{Z}_q^{n \times n}$, such that $A \cdot sk = e \approx 0$
- Enc : computes $C = mI_n + RA$, where $m \in \mathbb{Z}_q$ is the message, I_n the identity matrix, R a $n \times n$ matrix with random binary elements
- Dec : outputs first element of the n-dimensional vector $Csk = mI_nsk + RAsk \approx mI_nsk = (m, ms_2, ...ms_n)$. R is small, so $RAsk = Re \approx 0$

While this scheme benefits from better error management and a more "natural" procedure, the performance suffers when used on complex large operations. To address this issue, multiple bootstrapping optimizations have been proposed, the most remarkable of which is the Alperin-Peikert (*AP*) [75, 77]. In fact, these consist of a new approach to evaluate decryption as an *arithmetic circuit*, rather than a boolean one. Interestingly, the AP approach enables $Eval$ to be performed in the same time as the noise reduction operation, later refered to as Programmable Bootstrapping (PBS). The outcome is a *quasi-optimal* construct - *quasi-linear* ($\tilde{\mathcal{O}}(\lambda)$) in the security parameter λ operations to bootstrap any 2^{λ} secure LWE system.

Gama et al. [78] proposed *GINX* as an alternative to AP [75, 77], that generalizes over RLWE problems with polynomial noise overhead. Similar to the AP approach, GINX is also considered a PBS technique, and represents an important step towards fast bootstrapping. In terms of distinctions, as shown by Micciancio and Polyakov [79] GINX in practice is preferable for binary secrets, while AP outperforms it for larger secret sizes[8]. Following this line of research, a significant number of papers were published to improve or extend the functionality of these bootstrapping procedures [80–84], starting either from AP or GINX. These were monumental in constructing the SOTA Third Generation schemes [23, 24].

Ducas and Micciancio [23] proposed a new RLWE scheme, FHEW, which leverages the Ring variant of the AP's quasi-optimal bootstrapping procedure [77]. Their main contribution, however, consists of a new method to compute the *NAND gate*, which is functionally complete[9]. Thus, instead of boolean or arithmetic circuits, FHEW uses NAND circuits to compute, which "introduces a much lower level of noise than previous techniques" [23]. FHEW operates, like the other Third Generation schemes, at a bit-level, which, coupled with the NAND gate computation achieves much better performance. However, in this construction bootstrapping must be performed after every operation.

Alternatively, Chillotti et al. [24] have proposed a generalized FHE method based on GINX. As a consequence, they vastly reduced the bootstrapping running time and key sizes. Technically, they proposed 3 FHE schemes over Torus referred to as *TFHE* [18]: TLWE as a generalized version of LWE, TRLWE its RLWE equivalent and TRGSW the generalized version of the ring equivalent of the GSW [73]. Being generalized versions, one could switch between them, by changing the domain of the application.

3.4 Forth Generation

The Forth Generation of schemes was characterized the breakthrough of computing on floating point numbers. Cheon, Kim, Kim and Song [25] proposed a new method of performing *approximate arithmetic*, initially known as *HEAAN*, by treating the noise as "part of error occurring during approximate computations". As such, the decryption Dec circuit has predefined level of precision when outputting the approximate value of the plaintext. This approach suffers from a small accuracy loss, which is bounded by the depth of the circuit. Their discovery was treated as a starting point for approximate arithmetic schemes, in which they laid out the foundational *LFHE* construction. They also proposed a new batching (SIMD) technique for RLWE constructions, which preserves the set precision while not increasing error growth.

The basic example they provide to illustrate this scheme is that the ciphertext will be decrypted as $Dec : \langle c, sk \rangle = m + e\ (mod\ q)$, using the same notion as before for (R)LWE problems. Additionally, for a small error compared to the message, that equation can become the new approximate message $m' = m + e$. Their construction can be extended to more complex functions, but in this form it presents an exponential bit size growth in terms of circuit depth. Consequently, to address this issue and obtain a linear ciphertext modulus growth, they proposed a technique referred to as *rescaling*, very similar to BV's [60, 65] *key switching*. It is realized by multiplying the decryption circuit by $1/p$, for some number p after multiplications, obtaining a valid encrypting of $\frac{m}{p}$ with a lower noise $\frac{e}{p}$ under $mod\ \frac{q}{p}$, hence the small accuracy loss. Over the course of multiple operations, the modulus decreases until operations are no longer feasible.

[8]Secret keys drawn from distributions (e.g. Gaussian [79]) over large intervals, as opposed binary or ternary secrets (e.g. $0, 1, -1$)

[9]Can represent any circuit using NAND gates

The following year, they augmented this approach with a bootstrapping method to transform the scheme into a FHE [85]. They used a *scaled sine function* to evaluate the decryption circuit approximately and obtain the original message in a "large ciphertext modulus". Subsequent constructions followed this blueprint to present optimization methods. Notably, Boemer et al. [86] improved the ciphertext additions and multiplications, as well as the runtime of scalar encoding, through the use of complex packing [18]. Moreover, Kim et al. [87] adapted the rescale technique to be used before computations, leading to a smaller error growth.

It's important to mention that to address the issue of versatility, CHIMERA [88] scheme was developed. The authors presented a method to switch between the 3 generations of RLWE representative schemes: TFHE [24], FV [21, 22] and CKKS [25, 85]. FV is by necessity the middle scheme, the one which can be switched in either of CKKS and TFHE, and vice-versa. This way, one could obtain the advantages of each these constructions in less defined or changing environment.

4 Characteristics

This section addresses the characteristics of FHE, examining the differences between the generations of constructions, on the Functionality, Efficiency, Usability and Security axis.

4.1 Functionality

Homomorphic Encryption first evolved in the form of PHE as a direct response to specific problems. As such, these were developed with low versatility in mind, high speed, no interactions with other parties and very precise use cases. Most of these schemes are already used in the real world scenarios [1,4,6,89], such as: electronic voting protocols [90], smart grid services [91] and biometric applications [40] through the Pai scheme [3]. However, the need for a general purpose encryption technique had intensified in the following period, due to technological advancements. Then, the topic of FHE became of interest since it can cover all the drawbacks of PHE schemes. The key differences between FHE and PHE can be found in Table 3. FHE was developed so that it can enhance data secrecy during any computations. The main goal is to become a general purpose tool, that can be used for ad-hoc applications, as well as, underlying secure infrastructure. In terms of reliability, FHE is only bounded by the hardness of the problems is based on, not circuit depth or types of operations, as LFHE or PHE are.

Especially in the age of Artificial Intelligence, when data is capital, superior security measures have to be implemented not only to prevent intrusions, but also to protect data in such event. Specifically for these kinds of applications FHE was developed. For example, Munjal et Al. [11] have analyzed the benefits FHE can bring to the healthcare sector via secure Electronic Health Record (EHRs), where data leaks carry high risks. Moreover, the applicability of FHE is still under study in relation to the 5/6G technology, in the process called *fog computing* [92]. Specifically, FHE has a role to play in the data aggregation services used via fog computing in smart grids (smart cities' infrastructures) [93]. Additionally, protected data could be used in the training of Machine Learning (ML) models without exposing sensitive information in the process, regardless of the ML model. Architectures for ML applications, such as Neural Networks (CNN) [94], Decision Trees (DT) [95] and Deep Learning [96] are being developed. However, the adoption process requires overcoming some major obstacles in terms of *performance*, *standardization* and *interoperability*, further discussed in Section 7.

4.2 Usability

Each generation of schemes has their respective practical features, presented in Table 2, which highlight advantages, disadvantages, which lead to specific use cases in practice, as displayed by Table 1 in Appendix B.

The second generation of schemes, such as BGV [20] and B/FV [21, 22], introduced integers operations, packing via CRT, which allowed parallel computations to be performed, but present a slow bootstrapping mechanism. However, for circuits with predefined circuit length, these can display an efficient leveled design, which circumvent the need for noise reduction. These are appropriate for exact operations on large arrays (batches) of data, as these can be parallelized.

Similarly, the forth generation of schemes via CKKS [25, 85], improved the batching framework to an exponential number of entries, which can lead to an amortized speed of the slow bootstrapping technique. Remarkably, these introduced approximate arithmetic FHE, allowing operations to be performed with real numbers. Moreover, they optimized specific computation procedures used in Machine Learning, Data Streaming and Analysis, such as polynomial approximation, logistic regression and multiplicative inverse [18].

However, the second and forth generations are not well equipped to perform non-linear operations. Instead, the third generation via FHEW [23] and TFHE [24] can outperform the others, as long as the computations can be expressed as boolean circuits. The main two limitations for the third generation are that bootstrapping has to be performed after every operation/NAND gate, and that these don't support batching.

4.3 Efficiency

FHE introduces an overhead in terms of performance, compared to plaintext computations, due to multiple main factors: Key Generation (order of minutes), encrypted-domain representation, ciphertext sizes, Bootstrapping procedures, which differ between constructions. Generally, the operands are ciphertexts with abstract algebraic structures, such as polynomials over rings, not integers or floats, thus being orders of magnitude larger than the plaintext.

The Bootstrapping procedures differ between generations, the most efficient achieving amortized[10] polynomial overhead. These might be repeated frequently, depending on the error growth, number of operations and the depth limit. In addition, some schemes require the use of key or mod switching, or rescaling, that increase overall computation time. The subject of efficiency was further examined in depth in Sections 3.2 -3.4 with regard to each construction in particular.

[10]The 2nd and 4th generations, which offer support for batching (SIMD), amortize their very slow (order of minutes) bootstrapping procedures through parallel computing.

4.4 Security

In terms of security, all FHE schemes use non-deterministic encryption algorithms, which ensure that the same plaintext encrypted twice does not yield the same result. As such, FHE schemes benefit from the security level *IND-CPA:* Indistinguishability under Chosen Plaintext Attack, also known as semantic security. It implies that an adversary with access to the public parameters cannot distinguish between ciphertexts that result from encrypting two adversarially chosen plaintexts [97].

However, many applications require a stronger security level, known as Indistinguishability under Non-adaptive Chosen Ciphertext Attack (*IND-CCA1*). To achieve this level, an adversary with polynomial-bounded access to an encryption and decryption oracle, cannot reliably guess the message when given its encryption. A superior security level is *IND-CCA2*, in which the attacker is given access to the same oracle even after receiving the encryption of the message. In real-world scenarios, IND-CCA2 security level is required, as it prevents any meaningful modification of a given ciphertext [97]. FHE is malleable by design, so it cannot achieve IND-CCA2 security, as an attacker could simply use the homomorphic property of the scheme to add to the ciphertext the encryption of a known number, decrypt the resulting ciphertext, and subtract that number to obtain the plaintext. However, some schemes can achieve IND-CCA1 security [97]. Moreover, some attempts to improve the security level have been made [98], but were proven to be unsuccessful [99].

5 Comparison

To further clarify the role of FHE as a cryptographic technique, this section will conduct a wider comparison with other privacy-enhancing techniques. The key aspects are also highlighted by the Tables 4 and 5 in Appendix B.

Multi-Party Computation

Secure MPC facilitates distributed computing capabilities with the underlying assumption of malicious party behavior profiles, such as: *malicious* - can compromise correctness, and *honest-but-curious* - follows the protocol but attempts to gain information. If the number of corrupted parties forms a minority, MPC can achieve fairness and guarantee output delivery, assuming access to a broadcast channel [100, 101]. Such protocols support innovation in the fields of healthcare, government technology and advertisement. For example, these can facilitate secure genome comparisons across institutions' databases to detect cancer predispositions [102].

Like FHE, MPC supports general purpose computations, but requires interactive protocols for each gate evaluation, limiting scalability in asynchronous or high-latency settings. Most MPC consists of three phases, namely: Input Sharing (such as Shamir's Secret Sharing [103]), Circuit Evaluation and Output Reconstruction [102]. Relevant semi-honest constructions are Yao's GC [104], GMW [105], BGW [106], and BMR [107]. These differ in circuit types (Boolean or Arithmetic), Round Complexity (Linear or Constant) and adversarial models, making them suitable for different use cases. For example, Yao's GC encodes a boolean circuit in *truth tables* and allows one party to evaluate it, whereas GMW represents the circuit as a *series of gates* and evaluates them collaboratively via bitwise secret sharing and oblivious transfer.

The collaborative environment and speed in computing with a limited number of parties makes them distinctive from FHE, which does not require coordination, trust or communication rounds between the parties. However, MPC shifts the cost from computation to communication, requiring multiple rounds of exchange per gate. Nevertheless, MPC can benefit from the development of FHE by enabling encrypted data computation on each parties' secret share.

Structured Encryption

Structured Encryption (StE) enables queries to be performed on *structured* data (as matrices, arrays, graphs, labeled data etc.) stored in untrusted servers, without revealing information about the operations or the data [108]. StE has evolved as a generalization of the Symmetric Searchable Encryption (SSE) proposed in 2000 [109]. SSE facilitated keyword searches on encrypted documents, further refined in 2003 [110] by introducing Secure Indexes. These enabled finding documents which included certain words in constant time, while ensuring the pointers are encrypted, through the use of Bloom filters. The following publications on SSE [111] and on secure keyword searches [112] gave stronger security definitions and more efficient constructions.

At a high level, StE takes as input structured data and outputs an encrypted structure and the respective ciphertexts. StE can be viewed as a functional encryption which uses a token, generated with the secret key, that can recover the pointers to the ciphertexts from the encrypted structure. To achieve this, there is an *induced permutation* that shuffles the data, acting as a mapping function, which inevitably leaks access pattern data. This vulnerability is leveraged in the IKK-type attacks [113], where attackers can accurately infer sensitive information, given some prior knowledge about the data.

StE offers greater performance, but is less versatile than FHE, which supports any computation. Instead StE is designed for specific use cases, where encrypted data structures represent client data stored on the server, enabling efficient searches and updates, at the cost of access pattern leakage.

Oblivious RAM

Integrating cryptosystems, such as FHE, secures data content, but using it may leak access patterns to memory (sequences of addresses or operations). ORAM complements them by protecting against software-level leaks, hiding access patterns. It facilitates secure data storage on untrusted or outsourced servers, by ensuring attackers cannot observe or infer memory accessing operations.

The initial proposal by Goldreich and Ostrovsky [114] assumed a client-server scenario, where the server stores information, while the client is the one performing operations on it. Subsequent works [115] have extended the capabilities of the server to also compute, which make use of PHE or LFHE [116, 117], to obtain confidentiality of both, *data content* and *access patterns*, enhancing the capabilities of servers to achieve trusted executions. These are evaluated in terms of *bandwidth, client* and *server storage* [118], highlighting

SOTA ORAM techniques, such as: SSS [119, 120], Burst ORAM [115], Ring ORAM [121], and Path ORAM [122].

Generally speaking, ORAM divides data into blocks and stores it in graph-like structures. It continuously shuffles and re-encrypts it as it's being accessed, thus offering *indistinguishability* between objects accessed, and parties that access them. However, typical ORAM constructions are vulnerable to side-channel (hardware-level) attacks that leverage execution time [123, 124], power consumption [125], and electromagnetic radiation [126]. By measuring the output from these side-channels and correlating them with private information, attackers can learn the underlying secret in the system [126].

Trusted Execution Environment

TEEs denote secure enclaves in hardware, where code and data are isolated and protected from the rest of the system (e.g. OS, Hypervisor or applications). This technology relies on a minimal Trusted Computing Base (TCB) and is heavily used in servers that require secrecy, at no additional complexity overhead, which alternative approaches, such as FHE, would introduce. As such, these are well suited for low latency - high throughput scenarios, such as file-based encryption, biometric authentication, and anti-piracy controls for video streaming [127]. There are two architecture designs for TEEs, namely *enclave-based*, with relevant examples being Intel SGX [128] and Keystone [129], and *VM-based*, with Intel TDX [130], and AMD SEV. The former one offers tighter control at the cost of platform fragmentation and limited memory models.

Given their reliance on the code TCB, all TEEs try to minimize it for enforcing the security guarantees. Nevertheless, they are vulnerable to micro-architectural design flaws and side-channel attacks, such as cache timing and speculative execution [131,132]. Hardware bugs (e.g. Foreshadow for SGX - L1TF [133]) can bypass enclave protections, highlighting the necessity of supply chain trust. Compared to FHE, TEEs have an entirely different usability model. TEEs shift the responsibility to the manufacturers, facilitating a secure computation space, isolated memory and trusted Input/Output. Conversely, FHE enhances the security of the computations with a layer of encryption, which expands the feasibility of secure operations to untrusted servers as well.

6 Responsible Research

This section will dive deeper into how the research endeavor was performed in terms of material choices and careful representation of information, with the aim of ensuring the transparency, trustworthiness, and reproducibility.

Code of Conduct

The research was conducted in accordance to the following established ethical frameworks: Association for Computing Machinery's (ACM) *Code of Ethics and Professional Conduct* [134], TU Delft's *Code of Conduct* [135] and National Society of Professional Engineers' (NSPE) *Code of Ethics for Engineers* [136]. These have guided source selection, comparative methodology, and the framing of claims throughout the paper.

Ethical Consideration and Privacy

Although this research project does not involve data collection or experimentation of any form, it unequivocally supports the principles of Privacy by design. European regulations, such as General Data Protection Regulation (GDPR) highlight the ideas of data collection minimization and maximal privacy, which are technologically realizable through FHE. In addition, it emphasizes that FHE can augment privacy-sensitive applications to enable compliance in computations, without compromising user choice of participation. As such, the ethical discussion around data handling is not empirical, but conceptual and normative. Thus, FHE serves not only as a research subject but also as an enabler of privacy-compliant data computation.

Replicability, Reproducibility and FAIR Data

This paper does not present new implementations of the exposed methods, nor data collection from already implemented libraries. Instead it composes an analytical and comparative endeavor of theoretical concepts, based only on public data - freely available, with no restrictions on access or usage. The cited sources are Findable and Accessible on the internet, without the need for authentication, and are used to create Interoperable and Reusable data points through this research project. Thus, the principle of *Reproducibility* is strongly emphasized - another researcher given the listed open references can trace the rationale behind technical or comparative claims made. These claims are often supported by previous publications, which are also cited. To specify:

- The core of the narrative presentation, the FHE scheme timeline, is agreed upon by the scientists and engineers in the field of encryption, having been published about 10 years ago [19], highly reviewed and widely adopted in all subsequent works.
- The mathematical constructions of the FHE schemes were analyzed starting from their publication source and displayed in light of other cited sources, such as surveys or reviews.
- The comparison tables are based on peer reviewed survey metrics and benchmark data from primary sources, co-authored by fellow research project students, which abide by the same principles.
- Other discussed items of interest, such as security claims, are grounded in mathematical provable concepts, otherwise cited or intentionally excluded.

Replicability is highlighted by this paper's use of primary source data. As such, it would be infeasible to construct an equivalent survey using entirely disjoint sources. The majority of the sources cited are the foundational layer of the subject. Any additional sources were used to support the claims or comparisons made by these central publication.

It is worth mentioning that the Reproducibility and Replicability of this research are only limited by interpretative lens of the literature, especially in the context of "Functionality", "Ethical Considerations" and "Discussion and Future Work". It is possible that other scientists may weigh in differently, but the supporting materials offered them the transparency required to retrace the rationale behind those claims.

Research Integrity

The core principle of the paper pertains to integrity. All actions taken in the research must be justifiable in terms of maintaining integrity. As such, original formal FHE constructions (primary sources) were given priority above auxiliary, processed information. For auxiliary sources, priority was given to the most cited among them or those that were produced by some of the most cited authors in the field. Related claims of reviews or surveys were verified in a forward referencing manner and cross-validated wherever possible, or excluded otherwise. Moreover, the message gathered from all the sources was reproduced as faithfully and truthfully as possible to its original source, and adheres to the same truth-seeking ethos as the original theoretical framing. To highlight this effort, the terminology used was always the same as that of the cited sources, and it coincides with the domain-wide terminology and methodology consensus. Moreover, it was further emphasized through clearly separating claims from mathematically-grounded construction and implications.

Bias

Bias cannot be avoided, yet it must be recognized. This report may be affected by two types of bias, namely *Selection* and *Interpretation* Bias. By acknowledging these biases, the paper invites critical reflection and further inquiry.

Selection Bias refers to the fact that the inclusion criteria was based on the most prominent (most cited or impactful) publications in the field, which may lead to overlooking lesser known, but highly relevant publications. However, one core axiom which this research operated under was the fact that in a quantifiable scientific field, such as Cryptography, the most relevant publications given enough time will have the most citations relative to their niche. Moreover, the taxonomy of the FHE schemes was adopted from Piekert [19], which is the only officially recognized one, but it may not be exhaustive. Consequently, certain schemes and constructions were deliberately excluded, given the fact that these do not overlap well with the classification presented or that these have not been peer reviewed.

Interpretation Bias refers to the choices the researcher has made in analyzing or framing the presented information. This, too, was minimized by continuous efforts of adopting what would be considered as general comparison metrics, such as ”Ciphertext Size”, ”Noise reduction mechanism” etc. Nevertheless, it is worth emphasizing that the interpretation of qualitative aspects, such as ”usability” or ”efficiency” was portrayed with interpretation bias. Such categories are dependent on the research scope - attempted to be exhaustive - and the personal understanding of the researcher. Similarly the conclusions are based on both, the quantifiable data and the limited experience and understanding of the subject.

Artificial Intelligence

The increasing accessibility of generative AI tools, such as Large Language Models (LLMs), presents new opportunities for scientific research. However, under no circumstance, should AI substitute critical thinking and analysis. As such, this paper did not rely on generative AI for content creation, citations, or technical explanations. However, it did make limited use of freely available ChatGPT (OpenAI) to improve clarity and consistency in language, in accordance with TU Delft's Research Project policy on AI use. It facilitated improvements to surface-level grammar and spelling in the final revision stage, not for analytical reasoning or source evaluation. A representative example was the prompt: ”Identify in the following paragraphs any writing mistakes without reformulating or adding any other information”. The resulting feedback was used to correct typographic errors and ensure writing consistency, while all analytical, comparative, and interpretive work remained the sole responsibility of the author.

Closing Remarks

This research was conducted with a commitment to ethical clarity, reproducibility, and fair representation of knowledge. Through rigorous source citation and cross-validation, transparent methodology, and bias acknowledgment, the project aims to adhere to responsible knowledge development within the fields of scientific and engineering research.

7 Discussion and Future Work

FHE has the potential to become the general tool for underlying security even in the event of intrusions. However, at the moment there are 3 characteristics that impede its mass adoption: *performance*, *standardization* and *interoperability*.

Firstly, FHE computations are still relatively slow [19]. Reducing the noise in any FHE demands evaluating the decryption circuit homomorphically, which impedes the efficiency of the scheme. Despite support for parallel computing (SIMD) and various optimizations, the required improvement for adoption might not be achievable from software-based solutions alone [18]. Instead, FHE hardware accelerators have been developed, such as Doroz et al. [137], Cousins et al. [138] and, more recently the F1 accelerator [139]. These have achieved up to 4 orders of magnitude speed-ups (the F1) in the SOTA schemes, such as BGV [20], GSW [73] and CKKS [25, 85]. Recent hardware-focused technologies are required to fill in the performance gap that would make FHE a standard practice.

Secondly, resources have been devoted for deep theoretical understanding and improvements, but standardization is required for large-scale usability. In turn, this implies establishing benchmarks in terms of FHE schemes and their respective parameters and abstracting away the theoretical hurdles through dedicated SDKs and APIs. Considerable progress had been made on this front by a consortium consisting of large companies, such as Microsoft and IBM, academic researchers and government representatives. They have made remarkable progress by paving the way to focused discussion in the direction of standardization, publishing standards of FHE schemes parameters [140].

Thirdly, in order to become a general-purpose tool, FHE has to enable for data interaction. By design, FHE cannot compute on data encrypted with a different key. This prevents its deployment in collective environments, such as distributed machine learning or decentralized systems. With this goal in mind, the extension of FHE to a Multi-Key FHE (MKFHE) [14] has been developed, but is less efficient than other MPC protocols. Moreover, the ciphertexts obtained this way cannot be decrypted if one of the parties doesn't deliver its share of the output. The subsequent publication of Threshold MKFHE (TMFHE) [141] has addressed this issue, at the cost of superior trust requirements between the parties. Nevertheless, industry adoption of FHE schemes requires data interoperability, along with active zero-knowledge proofs [18].

8 Conclusion

Fully Homomorphic Encryption (FHE) represents a crucial development in privacy-preserving computation, enabling data to be processed while remaining encrypted. This paper provided a technical and functional characterization of FHE, analyzing its evolution through mathematical design, efficiency constraints, and real-world relevance. Particular emphasis was placed on the limitations of current schemes, such as bootstrapping overhead, the features each iteration presents, and the role of FHE in a wider ecosystem of privacy-enhancing technologies.

Through the theoretical constructions, FHE helped lay the groundwork of zero-knowledge computing for untrusted, unsecured environments. In doing so, it displays a monumental potential in maintaining individual privacy, while ensuring ethical innovation. As such, its interaction with other privacy-enhancing techniques is a topic of interest, which can bring this vision closer to becoming a reality. However, obstacles pertaining to efficiency, applicability and interactivity have to be overcome in order for mass adoption to be achievable. To address these, this paper recommended enhancing the performance through dedicated hardware architecture, standardization through public-private partnerships and interoperability through integration with other privacy-enhancing purpose-built techniques.

A Appendix A: FHE Characteristics Comparison Tables

Table 1: High-level usability comparison of FHE schemes generations, extending the table in [18]

Characteristic	1st Gen	2nd Gen	3rd Gen	4th Gen
Schemes	Gentry, DGHV	BGV [20], B/FV [21, 22]	GSW [73], FHEW [23], TFHE [24]	CKKS [25, 85]
Operation domain	Ideal Lattices	Integers	Bitwise (Integers)	Real Numbers
Advantages	Support for packing (SIMD) Simplicity	Packing (SIMD) Fast linear functions Exact Arithmetic	Efficient boolean circuits Fast Bootstrapping Fast comparisons	Efficient packing (SIMD) Fast Linear Computation Fast Approximation
Disadvantages	Not secure Very slow	Slow Bootstrapping Slow complex functions	Frequent Bootstrapping No batching	Slow bootstrapping Accuracy Loss
Applications	Impractical	Large array operations	Many complex operations	Machine Learning

Table 2: Key characteristics of generations of FHE schemes

Generation	Operation Gates	Bootstrapping Speed	Packing (SIMD)	Error Growth Rate	Ciphertext Reduction	Bootstrapping Key size
2nd	Modular Arithmetic	Low	◑ ≈ 1000 slots	Quasi-polynomial High Linear	Key & Mod Switching Scale invariant	Very Large
3rd	Arbitrary Boolean Gates via PBS	Very High	○	Constant Polynomial (GSW)	Scale invariant	Large initially Smaller
4th	Approximate Arithmetic	Low High If Amortized	● $\approx 2^{15}$ slots	Linear	Rescaling	Large

Table 3: Comparison between types of Homomorphic Encryption constructions, expanding table in [40]

Attribute	Operation Type	Circuit Depth	Versatility	Speed	Use Cases
PHE	Additions or Multiplications	Unlimited	Low	Very High	Voting Protocols Biometric applications
SwHE	Additions and Multiplications	Limited	Medium	High	Medical Information Bioinformatics data
FHE	Additions and Multiplications	Unlimited	High	Low	Big Data Processing Sensitive Data Analytics Machine Learning

B Appendix B: Comparison with other Privacy Enhancing Techniques

Table 4: Functionality and Usability Comparison of Privacy-Enhancing Techniques

Technique	Computation Type	Parties Communication	Applicability	Use Cases
FHE	Any computation	Non-interactive Client–server	Available in open-source libraries	Medical data analysis Recommender systems Confidential ML
MPC	General computation (excluding specialized protocols)	Multiple clients or distributed parties	Used in practice but with limitations	Secure auctions DNA comparison Collaborative research
ORAM	Data access	Non-interactive Client(s)–server(s)	Used in secure processors and oblivious DBs	SGX integration ObliDB, Signal protocol
StE	Specific data access on encrypted structures	Non-interactive Client–server	Practical protocols for specific structures	Encrypted DBMS (e.g. MongoDB)
TEE	Any computation	Interactive Client–server with attestation service	Optional in real world cloud deployment	Data analytics Trusted AI workloads Medical Federated Learning

Table 5: Security and Performance Comparison of Privacy-Enhancing Techniques

Technique	Threat Model	Information Leakage	Performance Overhead
FHE	IND-CCA1 Non-adaptive attack	None by itself	High: Key Generation & Polynomial Operations
MPC	Semi-honest or malicious	Nothing beyond function output	Constant or Linear
ORAM	Semi-honest or malicious	Leakage through side-channel attacks	Logarithmic
StE	Semi-honest	Access pattern sometimes response volume	Sublinear
TEE	Malicious actor controlling server	Access patterns, plaintext in CPU	Generally near-native, bottleneck in I/O heavy

C Appendix C: Methodology

C.1 Search Queries

An example of a *query* used to retrieve relevant literature publications: TITLE-ABS-KEY((”homomorphic encryption*” OR ”FHE” OR ”LFHE” OR ”bootstrapping*”) AND NOT ”multi-party*”).

Then, the following *filters* were applied, defining the inclusion and exclusion criteria for the resulting papers:

- Time frame: between 2009 and 2024
- Subject area: computer science, cryptography
- Document type: conference paper, article, journal, book
- Language: English
- Publication stage: Final

C.2 Research Questions

1. What are Homomorphic Functions and what are the different types of such functions?
2. What are the functional differences between FHE and PHE and SwHE?
3. What are the construction methods of a FHE?
4. How does Fully Homomorphic Encryption integrate into the current systems in terms of key sizes, overhead and threat model?
5. What are the most relevant (SOTA) FHE schemes?
6. How do the differences of construction manifest in the use case scenarios of these SOTA schemes?
7. What are the differences in terms of functionality, usability, security, and efficiency of the most relevant FHE schemes?
8. How does it compare to other privacy techniques such as Multi-Party Computation, Structured Encryption, Trusted Execution Environment and Oblivious RAM?

D Appendix D: BGV Mathematical Construction

Following the notation and construction of the BGV [18, 20] scheme:

Let d be a power of 2, q an odd positive integer modulus, $\mathcal{R} = \mathbb{Z}[x]/\langle f(x) \rangle$ a ring of polynomials with integer coefficients modulo the polynomial $f(x)$, $\mathcal{X}$ an error distribution over $\mathcal{R} = \mathbb{Z}[x]/\langle x^d + 1 \rangle$, and B a bound on the length of the elements outputted by $\mathcal{X}$.

- $KeyGen$: takes as input λ, randomly selects a small error e, chooses an element $s, e \in \mathcal{X}$, and sets the secret key $sk = (1, s) \in \mathcal{R}_q^2$. It generates $a \in \mathcal{R}_q$ uniformly at random and computes $b = as + 2e$, setting $pk = (b, -a)$
- Enc : takes as input pk and a message m, converting it in (m,0)$\in$ $\mathcal{R}_q^2$, and randomly selects $r, e_0, e_1 \in \mathcal{X}$. Outputs ciphertext $c = (c_0, c_1) = m+2(e_0, e_1)+pk\, r = (m + 2e_0 + br, 2e_1 - ar) \in \mathcal{R}_q^2$
- Dec : takes as input sk and a ciphertext c. Outputs $\langle c, sk \rangle = c_0 + c_1 s = (m + 2e_0 + 2e_1 s + 2er)$ mod q mod 2, which equals m
- *Addition:* Performed component-wise: $c + c' = (m + m') + 2(e_0 + e_0', e_1 + e_1') + pk\,(r + r') = c''$. It decrypts to $m + m'$ as long as the resulting error is smaller than q
- *Multiplication:* $\langle c, s \rangle \cdot \langle c', s \rangle = (c_0 + c_1 s)(c_0' + c_1' s) = d_0 + d_1 s + d_2 s^2$. Thus it leads to the extension of the secret key vector to $(1, s, s^2)$, for which key switching is proposed. This encrypts s^2 under s [11] and transforms the resulting ciphertext to $c_0'' + c_1'' s$.

[11]Operation requires circular security assumption

References

[1] Ronald L. Rivest and Michael L. Dertouzos. ON DATA BANKS AND PRIVACY HOMOMORPHISMS. *Foundations of secure computation*, 1 1978.

[2] David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. pages 59–66, 11 1998.

[3] Pascal Paillier. *Public-Key Cryptosystems based on composite degree residuosity classes*. 10 2007.

[4] Josh Benaloh. Dense probabilistic encryption. *Selected Areas in Cryptography*, 2 1999.

[5] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 2 1978.

[6] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 7 1985.

[7] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. *Evaluating 2-DNF formulas on ciphertexts*. 1 2005.

[8] Craig Gentry. Fully homomorphic encryption using ideal lattices. volume 9, pages 169–178, 05 2009.

[9] Nesrine Kaaniche, Maryline Laurent, and Sana Belguith. Privacy enhancing technologies for solving the privacy-personalization paradox: Taxonomy and survey. *Journal of Network and Computer Applications*, 171:102807, 8 2020.

[10] Zarina Chokparova and Leon Urbas. *Application of multiplicative homomorphic encryption in process industries*. 1 2022.

[11] Kundan Munjal and Rekha Bhatia. A systematic review of homomorphic encryption and its contributions in healthcare industry. *Complex Intelligent Systems*, 9(4):3759–3786, 5 2022.

[12] Ayantika Chatterjee and Khin Mi Mi Aung. *Fully homomorphic encryption in real world applications*. 1 2019.

[13] Javier Sanchez, Ronny Correa, Hernando Buenano, Susana Arias, and Hector Gomez. Encryption techniques: A theoretical overview and future proposals. pages 60–64, 3 2016.

[14] Pratyay Mukherjee and Daniel Wichs. *Two round multiparty computation via multi-key FHE*. 1 2016.

[15] Anamaria Costache, Kim Laine, and Rachel Player. *Evaluating the effectiveness of heuristic Worst-Case noise analysis in FHE*. 1 2020.

[16] Craig Gentry. Computing arbitrary functions of encrypted data. *Communications of the ACM*, 53(3):97–105, 3 2010.

[17] Nigel P. Smart. *Cryptography made simple*. 11 2015.

[18] Chiara Marcolla, Victor Sucasas, Marc Manzano, Riccardo Bassoli, Frank H. P. Fitzek, and Najwa Aaraj. Survey on Fully Homomorphic Encryption, Theory, and Applications. *Proceedings of the IEEE*, 110(10):1572–1609, 10 2022.

[19] Chris Peikert. A decade of Lattice Cryptography. *Foundations and Trends® in Theoretical Computer Science*, 10(4):283–424, 1 2016.

[20] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) Fully Homomorphic Encryption without Bootstrapping. *ACM Transactions on Computation Theory*, 6(3):1–36, 7 2014.

[21] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Paper 2012/144, 2012.

[22] Zvika Brakerski. *Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP*. 1 2012.

[23] Léo Ducas and Daniele Micciancio. *FHEW: Bootstrapping homomorphic encryption in less than a second*. 1 2015.

[24] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1):34–91, 4 2019.

[25] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. *Homomorphic encryption for arithmetic of approximate numbers*. 1 2017.

[26] Andrej Bogdanov and Chin Ho Lee. Homomorphic encryption from codes. Cryptology ePrint Archive, Paper 2011/622, 2011.

[27] Iti Sharma. Fully homomorphic encryption scheme with symmetric keys. *CoRR*, abs/1310.2452, 2013.

[28] Aviad Kipnis and Eliphaz Hibshoosh. Efficient methods for practical fully homomorphic symmetric-key encrypton, randomization and verification. Cryptology ePrint Archive, Paper 2012/637, 2012.

[29] Ron Rothblum. Homomorphic Encryption: From Private-Key to Public-Key. *Lecture notes in computer science*, pages 219–234, 1 2011.

[30] Zvika Brakerski. *When homomorphism becomes a liability*. 1 2013.

[31] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 12 1982.

[32] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):1–40, 9 2009.

[33] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC '05, page 84–93, New York, NY, USA, 2005. Association for Computing Machinery.

[34] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems*. 1 2002.

[35] Daniele Micciancio and Oded Regev. *Lattice-based cryptography*. 1 2009.

[36] Jeffrey Hoffstein, Jill Pipher, and J.H. Silverman. *An Introduction to Mathematical Cryptography*. Springer Publishing Company, Incorporated, 1 edition, 2008.

[37] Sanjeev Arora. Probabilistic checking of proofs and hardness of approximation problems, 9 1995.

[38] Johannes Blömer and Jean-Pierre Seifert. On the complexity of computing short linearly independent vectors and short bases in a lattice. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, STOC '99, page 711–720, New York, NY, USA, 1999. Association for Computing Machinery.

[39] Yi-Kai Liu, Vadim Lyubashevsky, and Daniele Micciancio. *On bounded distance decoding for general lattices*. 1 2006.

[40] Tan Fun and Azman Samsudin. A survey of homomorphic encryption for outsourced big data computation. *KSII Transactions on Internet and Information Systems*, 10(8), 8 2016.

[41] Avrim Blum, Merrick Furst, Michael Kearns, and Richard J. Lipton. *Cryptographic primitives based on hard learning problems*. 1 1994.

[42] Oded Regev. The Learning with Errors Problem, 6 2010.

[43] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. 5 2009.

[44] Peter W. Shor. Polynomial-Time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 10 1997.

[45] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. Cryptology ePrint Archive, Paper 2009/285, 2009.

[46] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. *On Ideal Lattices and Learning with Errors over Rings*. 1 2010.

[47] Abbas Acar, Hidayet Aksu, A. Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *CoRR*, abs/1704.03578, 2017.

[48] N. P. Smart and F. Vercauteren. *Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes*. 1 2010.

[49] Damien Stehlé and Ron Steinfeld. *Faster fully homomorphic encryption*. 1 2010.

[50] Craig Gentry and Shai Halevi. *Implementing Gentry's Fully-Homomorphic encryption Scheme*. 1 2011.

[51] N. P. Smart and F. Vercauteren. Fully homomorphic SIMD operations. *Designs Codes and Cryptography*, 71(1):57–81, 7 2012.

[52] Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. *Recovering short generators of principal ideals in cyclotomic rings*. 1 2016.

[53] Peter Campbell, Michael Groves, Dan Shepherd, and CESG. *SOLILOQUY: a CAUTIONARY TALE*. 2015.

[54] Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. *Fully Homomorphic Encryption over the Integers*. 1 2010.

[55] Steven Galbraith, Shishay Gebregiyorgis, and Sean Murphy. Algorithms for the approximate common divisor problem. *LMS Journal of Computation and Mathematics*, 19:58–72, 08 2016.

[56] Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi. *Fully Homomorphic Encryption over the Integers with Shorter Public Keys*. 1 2011.

[57] Jean-Sébastien Coron, David Naccache, and Mehdi Tibouchi. Public Key Compression and Modulus Switching for Fully Homomorphic Encryption over the Integers. *Advances in Cryptology – EUROCRYPT*, 1 2012.

[58] Hao-Miao Yang, Qi Xia, Xiaofen Wang, and Dianhua Tang. A new somewhat homomorphic encryption scheme over integers. 03 2012.

[59] Yuanmi Chen and Phong Q. Nguyen. Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers. Cryptology ePrint Archive, Paper 2011/436, 2011.

[60] Zvika Brakerski and Vinod Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) LWE. pages 97–106, 10 2011.

[61] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. *Scale-Invariant Fully Homomorphic Encryption over the Integers*. 1 2014.

[62] Jung Hee Cheon, Jinsu Kim, Moon Sung Lee, and Aaram Yun. CRT-based fully homomorphic encryption over the integers. *Information Sciences*, 310:149–162, 3 2015.

[63] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Batch fully homomorphic encryption over the integers. Cryptology ePrint Archive, Paper 2013/036, 2013.

[64] Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrède Lepoint, Mehdi Tibouchi, and Aaram Yun. *Batch Fully Homomorphic Encryption over the Integers*. 1 2013.

[65] Zvika Brakerski and Vinod Vaikuntanathan. *Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages*. 1 2011.

[66] Shai Halevi and Victor Shoup. "HElib" - https://github.com/homenc/HElib.

[67] Craig Gentry, Shai Halevi, and Nigel P. Smart. *Fully Homomorphic Encryption with Polylog Overhead*. 1 2012.

[68] Microsoft SEAL (release 4.0). https://github.com/Microsoft/SEAL, March 2022. Microsoft Research, Redmond, WA.

[69] Hao Chen and Kyoohyung Han. *Homomorphic lower digits removal and improved FHE bootstrapping*. 1 2018.

[70] Hao Chen, Kim Laine, Rachel Player, and Yuhou Xia. *High-Precision arithmetic in homomorphic encryption*. 1 2018.

[71] Carl Bootland, Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren. Efficiently processing Complex-Valued data in homomorphic encryption. *Journal of Mathematical Cryptology*, 14(1):55–65, 6 2020.

[72] Anamaria Costache and Nigel P. Smart. Which ring based somewhat homomorphic encryption scheme is best? Cryptology ePrint Archive, Paper 2015/889, 2015.

[73] Craig Gentry, Amit Sahai, and Brent Waters. *Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based*. 1 2013.

[74] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. Cryptology ePrint Archive, Paper 2013/541, 2013.

[75] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. Cryptology ePrint Archive, Paper 2014/094, 2014.

[76] family=Brakerski given i=Z, given=Zvika. Fundamentals of fully homomorphic encryption - a survey. 25:125–.

[77] Jacob Alperin-Sheriff and Chris Peikert. Practical bootstrapping in quasilinear time. Cryptology ePrint Archive, Paper 2013/372, 2013.

[78] Nicolas Gama, Malika Izabachène, Phong Q. Nguyen, and Xiang Xie. Structural lattice reduction: Generalized worst-case to average-case reductions and homomorphic cryptosystems. page 528–558, Berlin, Heidelberg, 2016. Springer-Verlag.

[79] Daniele Micciancio and Yuriy Polyakov. Bootstrapping in fhew-like cryptosystems. In *Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, WAHC '21, page 17–28, New York, NY, USA, 2021. Association for Computing Machinery.

[80] Sergiu Carpov, Nicolas Gama, Mariya Georgieva, and Juan Ramon Troncoso-Pastoriza. Privacy-preserving semi-parallel logistic regression training with fully homomorphic encryption. *BMC Medical Genomics*, 13(S7), 7 2020.

[81] Yongwoo Lee, Daniele Micciancio, Andrey Kim, Rakyong Choi, Maxim Deryabin, Jieun Eom, and Donghoon Yoo. Efficient FHEW bootstrapping with small evaluation keys, and applications to threshold homomorphic encryption. Cryptology ePrint Archive, Paper 2022/198, 2022.

[82] Antonio Guimarães, Edson Borin, and Diego F. Aranha. Revisiting the functional bootstrap in TFHE. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 229–253, 2 2021.

[83] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. Cryptology ePrint Archive, Paper 2016/870, 2016.

[84] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. *Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE*. 1 2017.

[85] Jung Hee Cheon, Kyoohyung Han, Andrey Kim, Miran Kim, and Yongsoo Song. *Bootstrapping for approximate homomorphic encryption*. 1 2018.

[86] Fabian Boemer, Anamaria Costache, Rosario Cammarota, and Casimir Wierzynski. NGrAPH-HE2: A High-ThroughPut Framework for Neural Network inference on encrypted data. *arXiv (Cornell University)*, 1 2019.

[87] Duhyeong Kim and Yongsoo Song. *Approximate Homomorphic Encryption over the Conjugate-Invariant Ring*. 1 2019.

[88] Christina Boura, Nicolas Gama, Mariya Georgieva, and Dimitar Jetchev. CHIMERA: Combining Ring-LWE-based fully homomorphic encryption schemes. *Journal of Mathematical Cryptology*, 14(1):316–338, 8 2020.

[89] Shafi Goldwasser and Silvio Micali. Probabilistic encryption how to play mental poker keeping secret all partial information. pages 365–377, 1 1982.

[90] Lukas Malina, Jan Hajny, Radek Fujdiak, and Jiri Hosek. On perspective of security and privacy-preserving solutions in the internet of things. *Computer Networks*, 102:83–95, 3 2016.

[91] Z. Erkin, J. R. Troncoso-Pastoriza, R. L. Lagendijk, and F. Perez-Gonzalez. Privacy-preserving data aggregation in smart metering systems: an overview. *IEEE Signal Processing Magazine*, 30(2):75–86, 2 2013.

[92] Marcelo Yannuzzi, Frank Van Lingen, Anuj Jain, Oriol Lluch Parellada, Manel Mendoza Flores, David Carrera, Juan Luis Perez, Diego Montero, Pablo Chacin, Angelo Corsaro, and Albert Olive. A New Era for Cities with Fog Computing. *IEEE Internet Computing*, 21(2):54–67, 3 2017.

[93] Liehuang Zhu, Meng Li, Zijian Zhang, Chang Xu, Ruonan Zhang, Xiaojiang Du, and Nadra Guizani. Privacy-Preserving authentication and data aggregation for FOG-Based smart grid. *IEEE Communications Magazine*, 57(6):80–85, 4 2019.

[94] Ehsan Hesamifard, Hassan Takabi, Mehdi Ghasemi, and Rebecca N. Wright. Privacy-preserving machine learning as a service. *Proceedings on Privacy Enhancing Technologies*, 2018(3):123–142, 4 2018.

[95] Alhassan Khedr, Glenn Gulak, and Vinod Vaikuntanathan. SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers. *IEEE Transactions on Computers*, 65(9):2848–2858, 11 2015.

[96] Joon-Woo Lee, Hyungchul Kang, Yongwoo Lee, Woosuk Choi, Jieun Eom, Maxim Deryabin, Eunsang Lee, Junghyun Lee, Donghoon Yoo, Young-Sik Kim, and Jong-Seon No. Privacy-Preserving machine learning with fully homomorphic encryption for deep neural network. *IEEE Access*, 10:30039–30054, 1 2022.

[97] Ran Canetti, Srinivasan Raghuraman, Silas Richelson, and Vinod Vaikuntanathan. *Chosen-Ciphertext Secure fully homomorphic encryption*. 1 2017.

[98] Linming Gong, Shundong Li, Qing Mao, Daoshun Wang, and Jiawei Dou. A homomorphic encryption scheme with adaptive chosen ciphertext security but without random oracle. *Theoretical Computer Science*, 609:253–261, 10 2015.

[99] Hyung Tae Lee, San Ling, and Huaxiong Wang. Analysis of Gong et al.'s CCA2-secure homomorphic encryption. *Theoretical Computer Science*, 640:104–114, 6 2016.

[100] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority. *Symposium on the Theory of Computing*, pages 218–229, 1 1987.

[101] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, page 73–85, New York, NY, USA, 1989. Association for Computing Machinery.

[102] Yehuda Lindell. Secure multiparty computation. *Communications of the ACM*, 64(1):86–96, 12 2020.

[103] Adi Shamir. How to share a secret. 22(11):612–613, November 1979.

[104] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 162–167, 1986.

[105] Oded Goldreich, S. Micali, and Avi Wigderson. How to play any mental game. pages 218–229, 01 1987.

[106] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). pages 1–10, 01 1988.

[107] D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, STOC '90, page 503–513, New York, NY, USA, 1990. Association for Computing Machinery.

[108] Melissa Chase and Seny Kamara. *Structured encryption and controlled disclosure*. 1 2010.

[109] Dawn Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. pages 44–55, 02 2000.

[110] Eu-Jin Goh. Secure indexes. Cryptology ePrint Archive, Paper 2003/216, 2003.

[111] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, page 79–88, New York, NY, USA, 2006. Association for Computing Machinery.

[112] Yan-Cheng Chang and Michael Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In *Proceedings of the Third International Conference on Applied Cryptography and Network Security*, ACNS'05, page 442–455, Berlin, Heidelberg, 2005. Springer-Verlag.

[113] Fateh Boucenna, Omar Nouali, Kamel Adi, and Samir Kechid. Access pattern hiding in searchable encryption. pages 107–114, 08 2021.

[114] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious rams. *J. ACM*, 43(3):431–473, May 1996.

[115] Jonathan Dautrich, Emil Stefanov, and Elaine Shi. Burst oram: minimizing oram response times for bursty access patterns. In *Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC'14, page 749–764, USA, 2014. USENIX Association.

[116] Craig Gentry, Shai Halevi, Charanjit Jutla, and Mariana Raykova. Private database access with HE-over-ORAM architecture. Cryptology ePrint Archive, Paper 2014/345, 2014.

[117] Daniel Apon, Jonathan Katz, Elaine Shi, and Aishwarya Thiruvengadam. *Verifiable oblivious storage*. 1 2014.

[118] Srinivas Devadas, Marten Van Dijk, Christopher W. Fletcher, Ling Ren, Elaine Shi, and Daniel Wichs. *Onion ORAM: a constant bandwidth blowup oblivious RAM*. 12 2015.

[119] E. Stefanov and E. Shi. ObliviStore: High performance oblivious cloud storage. *IEEE Symposium on Security and Privacy*, 5 2013.

[120] Emil Stefanov, Elaine Shi, and Dawn Xiaodong Song. Towards practical oblivious RAM. *arXiv (Cornell University)*, 1 2011.

[121] Ling Ren, Christopher Fletcher, Albert Kwon, Emil Stefanov, Elaine Shi, Marten Van Dijk, and Srinivas Devadas. Constants count: practical improvements to oblivious RAM. *USENIX security*, pages 415–430, 8 2015.

[122] Emil Stefanov, Marten Van Dijk, Elaine Shi, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. Path ORAM: an extremely simple oblivious RAM protocol. *Other repository*, 11 2013.

[123] Ralf Hund, Carsten Willems, and Thorsten Holz. Practical timing side channel attacks against kernel space aslr. In *2013 IEEE Symposium on Security and Privacy*, pages 191–205, 2013.

[124] Eran Tromer, Dag Arne Osvik, and Adi Shamir. Efficient cache attacks on aes, and countermeasures. *J. Cryptology*, 23:37–71, 07 2010.

[125] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '99, page 388–397, Berlin, Heidelberg, 1999. Springer-Verlag.

[126] Chongxi Bao and Ankur Srivastava. Exploring timing side-channel attacks on path-orams. In *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 68–73, 2017.

[127] Carlton Shepherd and Konstantinos Markantonakis. *Trusted Execution Environments*. 01 2024.

[128] Victor Costan and Srinivas Devadas. Intel SGX explained. Cryptology ePrint Archive, Paper 2016/086, 2016.

[129] Github Open Source. Keystone Enclave - An Open Framework for Architecting TEEs.

[130] Masanori Misono, Dimitrios Stavrakakis, Nuno Santos, and Pramod Bhatotia. Confidential vms explained: An empirical analysis of amd sev-snp and intel tdx. *Proc. ACM Meas. Anal. Comput. Syst.*, 8(3), December 2024.

[131] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. Spectre Attacks: exploiting speculative execution. *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1–19, 5 2019.

[132] Guoxing Chen, Sanchuan Chen, Yuan Xiao, Yinqian Zhang, Zhiqiang Lin, and Ten-Hwang Lai. SGXPectRe: Stealing Intel secrets from SGX enclaves via speculative execution. *IEEE Security Privacy*, 18(3):28–37, 1 2020.

[133] Michael Schwarz, Moritz Lipp, Daniel Moghimi, Jo Van Bulck, Julian Stecklina, Thomas Prescher, and Daniel Gruss. Zombieload: Cross-privilege-boundary data sampling. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 753–768, New York, NY, USA, 2019. Association for Computing Machinery.

[134] Association for Computing Machinery (ACM). Code of Ethics and Professional Conduct.

[135] TU Delft. Code of Conduct.

[136] National Society of Professional Engineers. Code of Ethics for Engineers.

[137] Yarkin Doroz, Erdinc Ozturk, and Berk Sunar. Accelerating fully homomorphic encryption in hardware. *IEEE Transactions on Computers*, page 1, 1 2014.

[138] David Bruce Cousins, Kurt Rohloff, and Daniel Sumorok. Designing an FPGA-Accelerated Homomorphic Encryption Co-Processor. *IEEE Transactions on Emerging Topics in Computing*, 5(2):193–206, 10 2016.

[139] Nikola Samardzic, Axel Feldmann, Aleksandar Krastev, Srinivas Devadas, Ronald Dreslinski, Christopher Peikert, and Daniel Sanchez. F1: A fast and programmable accelerator for fully homomorphic encryption. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '21, page 238–252, New York, NY, USA, 2021. Association for Computing Machinery.

[140] Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin Lauter, Satya Lokam, Daniele Micciancio, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. Homomorphic encryption standard. Cryptology ePrint Archive, Paper 2019/939, 2019.

[141] Eunkyung Kim, Hyang-Sook Lee, and Jeongeun Park. *Towards Round-Optimal secure multiparty computations: multikey FHE without a CRS*. 1 2018.