**HOMOMORPHIC ENCRYPTION**

**CHAPTER ONE: INTRODUCTION**

* 1. **Background to the Study**

Homomorphic Encryption is a form of encryption that allows computation on ciphertexts, generating an encrypted result which, when decrypted, matches the result of the operations as if they had been performed on the plaintext (Yousuf *et al.,* 2021). Homomorphic Encryption schemes are widely used in many interesting applications, such as private information retrieval, electronic voting, multiparty computation, and cloud computing (Cramer *et al.,* 2021). Abstract algebra, it is a structure-preserving map between two algebraic structures, such as groups (Yousuf *et al.,* 2021). There are four types of Homomorphic Encryption. Homomorphic Encryption (HE), Partially Homomorphic Encryption (PHE), Somewhat Homomorphic (SWHE), and Fully Homomorphic Encryption (FHE). HE is a form of encryption that allows searching the encrypted data without the need to decrypt it first and hence leaving it (Greenberg, 2017). Morris argues that there are many forms PHE that allow for some specific operations to be performed (namely addition and multiplication).

A cryptosystem is considered PHE if it represents either addition or multiple homomorphisms, but not both (Morris, 2023). It can also work with Big Integers of hundreds of bits, considering that the multiplications do not exceed the value of the module used in the encryption. Thus, it is very practical in this regard SWHE is considered a more generic form of Homomorphic Encryption in the sense that it can work on both addition and multiple Homomorphism. Its limitation though lies in the fact that it works with a limited number of bits (Fan & Vercauteren, 2022). Craig Gentry was the first to develop a Fully Homomorphic Cryptosystem in 2009 (Gentry, 2019). FHE is considered that way, since it presents both multiplicative and additive Homomorphism, with an unlimited number of bits (Morris, 2023). It is the most important advantage is its enhanced privacy and retrieval of private information in the sense that it enables computing on encrypted data without first having to decrypt it, which in turn allows data to be transferred to an untrusted cloud application without fear of data leakage (Cramer *et al.,* 2021), thus, it is seen as a safe storage state. Gai used a method of blending arithmetic operations over real numbers and proposed a novel tensor-based Fully Homomorphic Encryption solution. Gai’s proposed scheme (Gaj & Oiu, 2017). Despite how great this system is, yet one of its biggest drawbacks is the complexity of the system and the noise that it creates in the system. Yet, it can be applied on large scale in all segments of the governments like in the financial sector, the voting system, or even on the information of a large enterprise (Morris, 2023). Polynomial is a term that consists of variables and coefficients, involved in addition, multiplication, and subtraction-based operations (Chen *et al.,* 2018). Polynomials increase rapidly in all arithmetic equations since each operation leads to the creation of another polynomial. Thus, an infinite number is created within a single calculation. This is why it is called noise. This noise in major cipher text operations needs to be less than 1/2 to ensure that the decryption is correct (Chen *et al.,* 2018). To explain this further, with any mathematical operation, the noise created by the output is mostly larger than the noise of the input. Many schemes have been studied to decrease the noise made, yet many of them were very costly, like the BFV scheme and logistic Regression tool, and both are commonly used and add great value. With the focus of HE present a detailed examination of Homomorphic Encryption, exploring its principles, applications, advancements, and challenges. We delve into the underlying mathematics, cryptographic techniques, and recent developments in HE schemes. While also showing some points on the other various types of Homomorphic encryption as a whole.

* 1. **Statement of the Problem**

Homomorphic Encryption (HE) faces several challenges that hinder its widespread adoption and practical implementation.HE schemes often incur significant computational overhead due to the complex mathematical operations involved. The encryption, computation, and decryption processes can be resource-intensive, leading to increased latency and reduced performance. Despite significant advancements in homomorphic encryption (HE) schemes, there exists a substantial difference between theoretical proposals and practical implementations in real-life applications. There is a need to address the security concerns associated with HE schemes, ensuring that they remain robust against known attacks and provide strong guarantees for confidentiality. Comparisons among various HE schemes should be conducted across diverse use cases to determine the best fit for specific applications. Developing efficient and scalable HE schemes capable of handling large datasets and complex operations is crucial for widespread adoption in industries like finance, health care, and cloud computing. Investigating novel techniques to reduce the overhead inherent in current HE schemes would enable more cost-effective and faster processing times. Exploring hybrid approaches combining traditional encryption methods with HE to achieve better trade-offs between security, performance, and usability. Assessing the impact of HE on privacy protection mechanisms in databases and other data-intensive systems. The security of HE schemes relies on certain mathematical assumptions, such as the hardness of specific computational problems. Adversarial advancements and breakthroughs in cryptography could potentially undermine these assumptions, leading to vulnerabilities in HE schemes.HE schemes typically involve complex mathematical operations to enable computations on encrypted data.

* 1. **Objectives of the Study**

1. To develop a framework for homomorphic encryption.
2. To evaluate the performance, security, and usability of homomorphic encryption schemes.
3. To examine the hardware acceleration of homomorphic encryption, balance, security, and accuracy.
4. To assess the user perceptions and usability.
   1. **Significance of the Study**

The study on homomorphic encryption is multifaceted and extends across various domains, including cryptography, data security, privacy protection, and the broader digital ecosystem. This is significance in Advancing Data Security and Privacy offers a revolutionary approach to data security by allowing computations to be performed directly on encrypted data, without the need for decryption. This capability ensures that sensitive information remains protected throughout the entire computation process, mitigating the risk of data breaches and unauthorized access. These studies will also enabling secure cloud computing, widespread adoption of cloud computing services, the need for robust data protection mechanisms has become increasingly critical. Homomorphic encryption enables organizations to securely outsource data processing tasks to the cloud while maintaining full control over their sensitive data, thus fostering trust and confidence in cloud-based services.

Facilitating privacy-preserving data analytics, in an era of big data analytics, preserving data privacy is paramount, especially when dealing with sensitive or personally identifiable information. Homomorphic encryption allows organizations to perform advanced analytics and machine learning tasks on encrypted data, thereby protecting individual privacy while still deriving meaningful insights from the data. Supporting secure multiparty computation**,** Homomorphic encryption plays a crucial role in enabling secure multiparty computation, where multiple parties can jointly compute a function over their private inputs without revealing any sensitive information to each other. This capability has applications in areas such as collaborative research, confidential data sharing, and secures voting systems.

* 1. **Definition of Key Terms**

**Advancing Data Security and Privacy:** Homomorphic encryption offers a revolutionary approach to data security by allowing computations to be performed directly on encrypted data, without the need for decryption. This capability ensures that sensitive information remains protected throughout the entire computation process, mitigating the risk of data breaches and unauthorized access.

**Enabling Secure Cloud Computing:** With the widespread adoption of cloud computing services, the need for robust data protection mechanisms has become increasingly critical. Homomorphic encryption enables organizations to securely outsource data processing tasks to the cloud while maintaining full control over their sensitive data, thus fostering trust and confidence in cloud-based services.

**Facilitating Privacy-Preserving Data Analytics:** In an era of big data analytics, preserving data privacy is paramount, especially when dealing with sensitive or personally identifiable information. Homomorphic encryption allows organizations to perform advanced analytics and machine learning tasks on encrypted data, thereby protecting individual privacy while still deriving meaningful insights from the data..

**Supporting Secure Multiparty Computation:** Homomorphic encryption plays a crucial role in enabling secure multiparty computation, where multiple parties can jointly compute a function over their private inputs without revealing any sensitive information to each other. This capability has applications in areas such as collaborative research, confidential data sharing, and secure voting systems.

**Enhancing Regulatory Compliance:** In light of increasingly stringent data protection regulations such as GDPR, HIPAA, and CCPA, the adoption of homomorphic encryption can help organizations comply with regulatory requirements while still leveraging data for analysis and decision-making. This ensures that sensitive data is handled in a manner that respects individual privacy rights and regulatory obligations.

**Fostering Innovation and Technological Advancement:** Research and development in homomorphic encryption drive innovation in cryptographic techniques, algorithms, and protocols. By addressing challenges related to efficiency, scalability, and usability, studies in this area pave the way for the development of more practical and widely applicable encryption solutions, fostering technological advancement and innovation in the field of data security.

**Empowering Data-Centric Applications:** The ability to perform computations on encrypted data opens up new possibilities for data-centric applications in various domains, including healthcare, finance, telecommunications, and government. By ensuring the confidentiality and integrity of sensitive data, homomorphic encryption enables the development of secure and privacy-preserving applications that empower individuals and organizations to leverage the full potential of their data.

**Contributing to the Digital Economy:** As data continues to be recognized as a valuable asset in the digital economy, the adoption of homomorphic encryption can help build trust and confidence in data-driven technologies and services. By protecting data privacy and security, studies in this area contribute to the growth and sustainability of the digital economy, fostering innovation, competitiveness, and economic development.

**CHAPTER TWO: LITERATURE REVIEW**

**2.1 Introduction**

This subsection is outlined in detail for an in-depth understanding of the subject matter.

**2.2 Conceptual Review**

**2.2.1 Introduction: Homomorphic Encryption (HE)**

Homomorphic Encryption (HE) is a particular type of encryption that maintains certain algebraic structure between the plaintext and ciphertext. One example of HE is where the product of any two ciphertexts is equal to the ciphertext of the sum of the two corresponding plaintexts, when all the encryptions use the same key. This is represented as: E(m1) \* E(m2) = E(m1+m2) (Tilborg & Jajodia, 2019). Homomorphism therefore simply means that the algebraic operations can be performed on the encrypted text, and when decrypted, the result will be the same as that of performing those operations on the unencrypted text. Public key cryptosystems such as the RSA, ElGamal, Paillier, etc. are homomorphic. HE is computationally indistinguishable because it is secure against passive attacks. Homormorphism in encryption correspond with mathematical concepts of homomorphism.RSA encryption for example is multiplicatively homomorphic. This is because of the property, for any m1,m2, ϵ Z\*n, (me1 mod n ) \* (me2 mod n) = (m1m2)e mod n.  
The ElGamal encryption is also multiplicatively homomorphic; it can however also be formulated to be additively homomorphic. The Pallier encryption is additively homomorphic – where r1, r2, ϵ RZ\*n Fontaine and Galand (2017) define homomorphism with the following notation:

For all a and b in P and k in K, if Ek(a) ○ Ek b) = Ek (a ◊ b) Then the encryption system can be said to be homorphic. Other types of homomorphisms also exist. For example the Goldwasser-Micali encryption is based on quadratic residues and is XOR-homormorphic.  
HE is important in cryptography because it preserves the relationships between elements across the encryption transformation. This means that the natural structure of the elements, and the relationships between them, is preserved in spite of the elements being encrypted.  
Fully homomorphic encryption is a strictly more powerful type of HE. It relates to the mathematical concept of ring homomorphism. It involves two operations such as addition and multiplication, and a finite ring. Fully HE is both additively and multiplicatively homomorphic (Smart, 2015). This means that any computing or mathematical function can be evaluated solely on the encrypted values, with knowledge of the public key (Tilborg & Jajodia, 2019).the following example. If Alice encrypts some information m and stores the ciphertext c on a remote server, she can perform mathematical functions on the ciphertext as if it were done on the unencrypted text, without having to decrypt the ciphertext. This saves on time and computational power. This is a relevant issue when the ciphertext is a huge database and decrypting would take a long time. With HE, the function can be executed on the ciphertext; even more notably, the fact that makes this characteristic useful is that every mathematical function can be expressed as a series of additions and multiplications over a ring. This means that the mathematical function F can be expressed as a series. A variation of Fully HE is SHE,or somewhat homomorphic encryption, where a limited number of additions and multiplication operations can be evaluated. Beyond the specific threshold of number of operations done on the ciphertext for the encryption system, the ciphertext in SHE (or SWHE, somewhat homomorphic encryption) becomes too noisy and the decryption will fail. Most SHEs can be converted to FHEs with bootstrapping.

**2.2.2 Practical Problems**

HE was recognized as far back as in the 1970s, in the properties of public key cryptosystems. The potential to use these for advanced privacy protection was also recognized. This is because if the encryption is Homomorphic, then fundamental factors (primitives) are enabled, such as oblivious transfer, and consequence on oblivious transfer, secure multiparty computation. HE combined with threshold cryptography makes electronic elections possible (Tilborg & Jajodia, 2019).

One of the main problems with HE systems such as RSA is that RSA has to pad the message with random bits to achieve semantic security, and this padding makes the encryption system lose its homomorphism (Fontaine & Galand, 2017). RSA is deterministic, which means that the encrypted message can be computed whereas in probabilistic encryption systems the outcome is not necessarily computable directly, but can be guessed. Adeterministic public key encryption system necessarily leaks information to the adversary. The adversary can intercept the ciphertext c, and know that the message m is from a small set of {m1, m2….mn) of possible messages. He can then find out m by computing the ciphertexts and comparing them with C. The implication of this possibility is that encryption systems have to have a degree of randomness in order to be truly or fully secret. The adversary should not be able to predict the ciphertext of a message. The introduction of randomness makes the encryption system probabilistic rather than deterministic (Delfs & Knebl, 2017). However, Zhou and Young (2020) suggest that it is possible to build a semantically secure public key encryption scheme that is also additively homomorphic. Bogos *et al* (2017) however contest this claim, publishing details of their attack on a HE that was claimed to be semantically secure, but found to be not semantically secure after the attacks.  
Moore *et al* (2018) explain that current HE schemes are still not efficient enough for real time application, as the execution can take too long. For example they point out that the key generation in Gentry and Halevi’s (2021) scheme can take anywhere between 2.5 seconds to 2.2 hours – this is too long for the encryption program to be practically useful. Waiting for 2.2 hours to encrypt a message will cripple modern communication systems which rely on almost instantaneous communication. Furthermore, FHE schemes also have heavy memory usage. In order to provide security guarantees, very large ciphertext and public keys are required. Finally, the requirement for bootstrapping to reduce and manage noise that is created in the execution of homomorphic operations on ciphertexts. Overall they suggest that improvements in HE can come from improvements in the theory as well as implementation of HE.

**2.2.3 HE Schemas**

A good classification of HE schemas is presented by dos Santos *et al* (2019).

**2.2.3.1 Gentry**

The Gentry 2019 HE schema is considered to be the earliest HE schema that has been developed. Gentry and Halevi (2021) explain that the construction of the FHE begins with the construction of a SHE scheme that supports evaluating low degree polynomials on the encrypted data. After this the decryption procedure had to be modified so that it can be expressed as a low degree polynomial. Bootstrapping completed the process to make the SHE into an HE. Once the degree of polynomials that can be evaluated by the scheme exceeds twice the degree of the decryption polynomial, the scheme is boots trappable and can be converted into an FHE. The modification of the decryption procedure is done by adding to the public key an additional hint about the secret key. This is called the sparse subset sum problem (SSSP). The public key is enhanced with a large set of vectors, and a sparse subset of the public keys adds up to the secret key. A ciphertext of the underlying scheme is post-processed using the additional hint. This allows it to be decrypted with a low degree polynomial. Overall this encryption method is efficient. The most expensive operation during the encryption is evaluating the degree (n-1) polynomial u at the point r.

**2.2.3.2 Helib**

The main characteristic of SHEs are that they have noise that grows with each homomorphic encryption, until it is so large that it causes decryption errors. Halevi and Shoup (2017) present Helib, which is a software library that implements the BGV HE. This HE focuses on the use of Smart-Vercauterenciphertext packing techniues and Gentry-Halevi-Smart optimisations. The system defines the set of operations that can be applied homomorphically and the cost of each operation.

The most efficient FHEs that are currently available are ring learning with errors (RLWEs). There are different types of RLWEs, such as BGVs, Brakerski, etc. RLWE differs from LWE in that a polynomial ring is used instead. Helib is purely written in C++, implements BGV encryption and supports optimisations such as relinearisation, bootstrapping and packing. The key characteristic of BGV is that it is a levelled homomorphic encryption scheme. This means that the ciphertext space is not fixed. Levels are used to reduce noise inside the ciphertext. As multiplication adds greater noise level is generally changed after multiplication.

Relinearisation is used to reduce the overhead in ciphertext multiplication. Packing is the use of several messages in one ciphertext. It is used to reduce the numbers of ciphertext, and to club together the computation time required for the encryption. Packing can be done into the coefficients, as well as into the subfields (Halevi & Shoup, 2017). Overall the main drawback of Helib is the overhead that is in the order of seconds (Mazonka *et al*, 2016).

**2.2.3.3 Cryptoleq**

SHE is considered to be sufficient for practical applications, on account of the lower overheads. Although they offer fewer operations, and are less powerful than FHEs in terms of their computational completeness, their efficiency makes up for the shortcomings and makes them a very practical choice. Cryptoleq is a SHE that is based on the concept of an abstract machine that is capable of performing general purpose computation on encrypted programs. The operands of this program are protected using the Paillier SHE (or PHE, partially homomorphic encryption), which supports additive homomorphism.Multiplicative homomorphism is achieved by inventing a heuristically obfuscated software re-encryption module. This module is blended into the program. It also supports probabilistic encryption. This means that Cryptoleq allows both encrypted and unencrypted instruction operands in the same program and memory space (Mazonka *et al,* 2016)bBecause Cryptoleq uses a secure probabilistic PHE, there is generally no leakage of plaintext information. However adversaries may analyse the side channel information of the Cryptoleq execution, such as IP trace patterns, memory events, etc. However the code based obfuscation can provide heuristic guarantees (Mazonka *et al*, 2016)b

**2.2.3.4 DGHV Variant**

Dos Santos *et al* (2019) put forward an implementation of a DGHV variant that reduces the prohibitive size of public keys at the cost of computational power. Their implementation is based on Coron’s () variation of Gentry (2019). It uses Python and is found to be slightly faster compared to Coron’s implementation. This implementation does not contribute any change in the algorithm as put forward by Coron () but it contributes in the form of a better implementation. This shows that execution speed can be improved by implementation strategy and choices. Dos Santos et al (ibid) also suggest that further improvements in execution speed can be done using code parallelism, the Pythom-specific Numba library, the use of FPGAs, etc

**2.2.4 FHE with Noise Reduction by Liu (2015)**

Liu (2015) presents a new FHE scheme that he claims is efficient for practical applications. The main advantage of this scheme is that it does not need the noise reduction that is required in other FHE schemes, such as bootstrapping and modulus switching. In essence, it is said that noise reduction is included in the algorithm. This is done by allowing arbitrarily large noises in its ciphertexts. In this method, the dimensions of the ciphertext do not change with homomorphic operations. Furthermore all ciphertext elements are in a finite domain, making the scheme very compact. This method is also semantically secure. Liu (ibid) also implements the proposed algorithm in Java using relevant Java packages such as SunJCE. They find that their scheme is slightly faster than AES for encryption, and much faster than AES for decryption. They also conduct a performance evaluation for homomorphic operations, and find that the time increases with the number of bits; operations with 10,000 bits for example take times of less than 6 seconds. Overall this means that this scheme is useful in data processing applications, and hence can be said to be perhaps the most practical implementation of FHE currently available.

**2.3 Theoretical Review**

In ancient Greek, the term   (homos) was used to mean “same,” while (morphe) was used for “shape” (Liddell & Scott 1896). Then the term homomorphism was coined and used in different areas. In abstract algebra, homomorphism is defined as a map preserving all the algebraic structures between the domain and range of an algebraic set (Malik *et al*. 2017). The map is simply a function, i.e., an operation that takes the inputs from the set of domains and outputs an element in the range (e.g., addition, multiplication). In the cryptography field, homomorphism is used as an encryption type. Homomorphic Encryption (HE) is a kind of encryption scheme that allows a third party (e.g., cloud, service provider) to perform certain computable functions on the encrypted data while preserving the features of the function and format of the encrypted data. Indeed, this homomorphic encryption corresponds to a mapping in the abstract algebra. As an example for an additively HE scheme, for sample messages m1 and m2, one can obtain E (m1 + m2) by using E (m1) and E (m2) without knowing m1 and m2 explicitly, where E denotes the encryption function. Normally, encryption is a crucial mechanism to preserve the privacy of any sensitive information. However, the conventional encryption schemes cannot work on the encrypted data without decrypting it first. In other words, the users have to sacrifice their privacy to make use of cloud services such as file storing, sharing, and collaboration. Moreover, untrusted servers, providers, and popular cloud operators can keep physically identifying elements of users long after users end the relationship with the services (McMillan 2018). This is a major privacy concern for users. In fact, it would be perfect if there existed a scheme that would not restrict the operations to be computed on the encrypted data while it would be still encrypted. From a historical perspective in cryptology, the term homomorphism was used for the first time by Rivest *et al*. (1978a) in 1978 as a possible solution to the computing without decrypting problem. This given basis in Rivest *et al.* (1978a) has led to numerous attempts by researchers around the world to design such a homomor- phic scheme with a large set of operations. In this work, the primary motivation is to survey the HE schemes focusing on the most recent improvements in this field, including partially, somewhat, and fully HE schemes. An early attempt to compute functions/operations on encrypted data is Yao’s garbled cir- cuit1 study (Yao 1982). Yao proposed a two-party communication protocol as a solution to the 1A circuit is the set of connected gates (e.g., AND and XOR gates in Boolean circuits) where the evaluation is completed by calculating the output of each gate in turn. millionaires’ problem, which compares the wealth of two rich people without revealing the exact amount to each other. However, in Yao’s garbled circuit solution, ciphertext size grows at least lin- early with the computation of every gate in the circuit. This yields a very poor efficiency in terms of computational overhead and too much complexity in its communication protocol. Until Gen- try’s breakthrough in Gentry (2019), all the attempts (Rivest *et al*. 1978b; Goldwasser & Micali 1982; ElGamal 1985; Benaloh 1994; Naccache and Stern 1998; Okamoto and Uchiyama 1998; Paillier1999; Damgård & Jurik 2011; Kawachi *et al*. 2017; Yao 1982; Boneh *et al*. 2015; Sander et al. 1999; Ishai & Paskin 2017) have allowed either one type of operation or a limited number of operations on the encrypted data. Moreover, some of the attempts are even limited over a specific type of set (e.g., branching programs). In fact, all these different HE attempts can neatly be categorized under three types of schemes with respect to the number of allowed operations on the encrypted data as follows: (1) Partially Homomorphic Encryption (PHE) allows only one type of operation with an unlimited number of times (i.e., no bound on the number of usages). (2) Somewhat Homomorphic Encryption (SWHE) allows some types of operations a limited number of times. (3) Fully Homo- morphic Encryption (FHE) allows an unlimited number of operations for an unlimited number of times.

PHE schemes are deployed in some applications like e-voting (Benaloh 1987) or Private Information Retrieval (PIR) (Kushilevitz & Ostrovsky 1997). However, these applications were restricted in terms of the types of homomorphic evaluation operations. In other words, PHE schemes can only be used for particular applications, whose algorithms include only addition or multiplication operations. On the other hand, the SWHE schemes support both addition and multiplication. Nonetheless, in SWHE schemes that are proposed before the first FHE scheme, the size of the ciphertexts grows with each homomorphic operation and hence the maximum number of allowed homomorphic operations is limited. These issues put a limit on the use of PHE and SWHE schemes in real-life applications. Eventually, the increasing popularity of cloud-based services accelerated the design of HE schemes that can support an arbitrary number of homomorphic operations with random functions, i.e., FHE. Gentry’s FHE scheme is the first plausible and achievable FHE scheme (Gentry 2019). It is based on ideal lattices in math, and it is not only a description of the scheme but also a powerful framework for achieving FHE. However, it is conceptually and practically not a realistic scheme. Especially, the bootstrapping part, which is the intermediate refreshing procedure of a processed ciphertext, is too costly in terms of computation. Therefore, a lot of follow-up improvements and new schemes were proposed in the following years.

Contribution: In this work, we provide a comprehensive survey of all the main FHE schemes as of this writing. We also cover a survey of important PHE and SWHE schemes as they are the first works in accomplishing the FHE dream and are still popular as FHE schemes are very costly. Furthermore, we include the FHE implementations focusing on the improvements with each scheme. FHE attracts the interest of people from very different research areas in terms of theoretical, implementation, and application perspectives. This survey is structured to provide an easy digest of the relatively complex homomorphic encryption topic. For instance, while a mathematician focuses on the improvement from a theoretical perspective, a hardware designer tries to improve the effi- ciency of FHE by implementing on GPU instead of CPU. All such different attempts make it harder to follow recent works. Therefore, it is important to collect and categorize the existing FHE works focusing on recent improvements. In addition, we mention the challenges and future perspectives of HE to motivate the researchers and practitioners to explore and improve the performance of HE schemes and their applications. This survey is intended to give a clear knowledge foundation to researchers and practitioners interested in knowing, applying, and extending state-of-the-art HE systems. Organization: The remainder of the article is organized as follows: In Section 3, descriptions of different HE schemes, PHE, SWHE, and FHE schemes, are presented. Then, in Section 4, different implementations of SWHE and FHE schemes, which were introduced after Gentry’s work, are given and their performances are discussed. Finally, in Section 5, further research directions and lessons learned are given and the article is concluded.

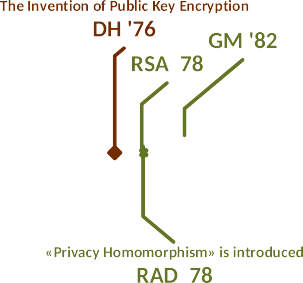
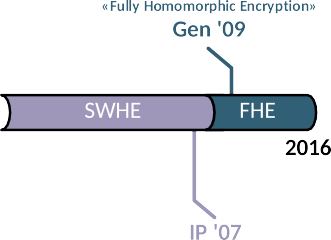
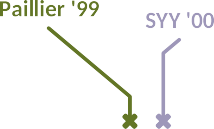
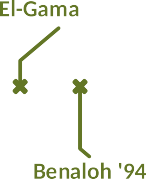
**2.3.1 Related Work**

Like our work in this article, there are similar useful surveys in the literature. In fact, unfortu- nately, some of the surveys only cover the theoretical information of the schemes as in Parmar *et al*. (2014) and Ahila and Shunmuganathan (2014), and some of them are directly for expert readers and mathematicians as in Vaikuntanathan (2021), Silverberg (2023), and Gentry (2014). Compared to these surveys, our survey has a broad reader perspective including researchers and practitioners interested in the advances and implementations in the field of HE, especially FHE. Furthermore, while the survey in Aguilar-Melchor *et al*. (2023) only covers the signal processing applications, that in Hrestak and Picek (2014) covers a few FHEs on only cloud applications. Since our survey is not limited to specific application areas, we do not articulate these specific application areas in detail, but we list the theory and implementation of all existing HE schemes, which can be used in possible futuristic application areas with recent advancements. After Fontaine and Galand (2017), many HE schemes were introduced. Compared to these useful surveys, our survey focuses on the most recent HE schemes, since most of the significant improvements were introduced recently (after 2009). Although (Moore *et al.* 2014b) is one of the most recent surveys, it focuses on the hardware implementation solutions of FHE schemes. This survey is not limited to hardware solutions, as, in addition to hardware solutions, it covers software solutions of implementations as well in the implementation section. After (Sen 2023 and Wu 2015), several new FHE schemes, which improve FHE in a sufficiently great way as to be worthy of attention, were proposed in the literature. Finally, it is worth mentioning that Armknecht *et al*. (2015) pro- vide a systematic explanation of the new terminology related to FHE and Armknecht et al. (223) provide security and a characterization of all existing group homomorphic encryption schemes, where they do not present all the HE schemes and their implementations in detail. Compared to these useful prior works, nonetheless, our survey is intrinsically different from the aforementioned surveys.

**2.3.2 Homomorphic Encryption Schemes**

In this section, we explain the basics of HE theory. Then, we present notable PHE, SWHE, and FHE schemes. For each scheme, we also give a brief description of the scheme.

Fig. 1. Timeline of HE schemes until Gentry’s first FHE scheme.



Definition 1. An encryption scheme is called homomorphic over an operation “x” if it supports the following equation: E (m1) x E (m2) = E (m1 x m2), ∀m1, m2 ∈ M, (1) where E is the encryption algorithm and M is the set of all possible messages. In order to create an encryption scheme allowing the homomorphic evaluation of arbitrary functions, it is sufficient to allow only addition and multiplication operations because addition and multiplication are functionally complete sets over finite sets. Particularly, any Boolean circuit can be represented using only XOR (addition) and AND (multiplication) gates. While an HE scheme can use the same key for both encryption and decryption (symmetric), it can also be designed to use the different keys to encrypt and decrypt (asymmetric). A generic method to transform symmetric and asymmetric HE schemes to each other is demonstrated in Rothblum (2021).

An HE scheme is primarily characterized by four operations: KeyGen, Enc, Dec, and Eval . KeyGen is the operation that generates a secret and public key pair for the asymmetric version of HE or a single key for the symmetric version. Actually, KeyGen, Enc, and Dec are not different from their classical tasks in conventional encryption schemes. However, Eval is an HE-specific op- eration, which takes ciphertexts as input and outputs a ciphertext corresponding to a functioned plaintext. Eval performs the function f () over the ciphertexts (c1, c2) without seeing the messages (m1, m2). Eval takes ciphertexts as input and outputs evaluated ciphertexts. The most crucial point in this homomorphic encryption is that the format of the ciphertexts after an evaluation process must be preserved in order to be decrypted correctly. In addition, the size of the ciphertext should also be constant to support an unlimited number of operations. Otherwise, the increase in the ciphertext size will require more resources and this will limit the number of operations.  
Of all HE schemes in the literature, PHE schemes support the Eval function for only either addition or multiplication, SWHE schemes support for only a limited number of operations or some limited circuits (e.g., branching programs), and FHE schemes support the evaluation of ar- bitrary functions (e.g., searching, sorting, max, min, etc.) for an unlimited number of times over ciphertexts. The well-known PHE, SWHE, and FHE schemes are summarized in the timeline in Figure 2 and are explained in the following sections with greater detail. The interest in the area of HE significantly increased after the work of Gentry (2019) in 2019. Therefore, we articulate the HE schemes, after Gentry’s work in greater detail and we also discuss their implementations and recent techniques to make it faster in Section 4. Here, we start with the PHE schemes, which are the first stepping stones for FHE schemes.

**2.3.2.1 Partially Homomorphic Encryption Schemes**

There are several useful PHE examples (Rivest *et al.* 1978b; Goldwasser & Micali 1982; ElGamal 1985; Benaloh 1994; Naccache & Stern 1998; Okamoto and Uchiyama 1998; Paillier 1999; Damgård & Jurik 2001; Kawachi *et al.* 2007) in the literature. Each has improved the PHE in some way. However, in this section, we primarily focus on major PHE schemes that are the basis for many other PHE schemes.

**2.3.2.2 RSA. RSA is an early example of PHE and introduced by Rivest *et al*. (1978b)** shortly after the invention of public key cryptography by Diffie and Hellman (1976). RSA is the first feasible achievement of the public key cryptosystem. Moreover, the homomorphic property of RSA was shown by Rivest *et al*. (1978a) just after the seminal work of RSA. Indeed, the first attested use of the term “privacy homomorphism” was introduced in Rivest *et al.* (1978a). The security of the RSA cryptosystem is based on the hardness of the factoring problem of the product of two large prime numbers (Montgomery 1994).2 RSA is defined as follows: — KeyGen Algorithm: First, for large primes p and q, n = pq and ϕ = (p  1) (q  1) are com- puted. Then, e is chosen such that дcd (e, ϕ ) and d are calculated by computing the multiplicative inverse of e (i.e., ed 1 mod ϕ). Finally, (e, n) is released as the public key pair while (d, n) is kept as the secret key pair.— Encryption Algorithm: First, the message is converted into a plaintext m such that 0 m < n, and then the RSA encryption algorithm is as follows:  
c = E (m) = me………………………………………………………1

(mod n), ∀m ∈ M, ……………………………………….(2)

where c is the ciphertext.— Decryption. Algorithm: The message m can be recovered from the ciphertext c using the secret key pair (d, n) as follows:

m = D (c ) = cd (mod n). …………………………………...(3)

— Homomorphic Property: For m1, m2 ∈ M, E (m1) ∗ E (m2) = (me (mod n)) ∗ (me (mod n)) = (m1 ∗ m2)e

(mod n) = E (m1 ∗ m2). …………………………………………..(4)

The homomorphic property of RSA shows that E (m1 m2) can be directly evaluated by using E (m1) and E (m2) without decrypting them. In other words, RSA is only homomorphic over mul- tiplication. Hence, it does not allow the homomorphic addition of ciphertexts.

**2.3.2.3 Goldwasser-Micali. GM proposed the first probabilistic public key encryption scheme proposed in Goldwasser and Micali (1982).** The GM cryptosystem is based on the hardness of quadratic residuosity problem (Kaliski 2005). Number a is called quadratic residue modulo n if there exists an integer x such that x 2  a (mod n). The quadratic residuosity problem decides whether a given number q is quadratic modulo n or not. The GM cryptosystem is described as follows:— KeyGen Algorithm: Similar to RSA, n = pq is computed where p and q are distinct large primes, and then x is chosen as one of the quadratic nonresidue modulo n values with ( x ) = 1. Finally, (x, n) is published as the public key while (p, q) is kept as the secret key. 2Here, we do not mean that RSA is secure. We mean the most basic attack on RSA (e.g., key recovering attack) has to solve the problem of the factoring of two large primes. For example, plain RSA is not secure against Chosen Plaintext Attacks (CPAs) as its encryption algorithm is deterministic. We use the same idea for the rest of the article as well. Because of the limited space, we do not discuss the details of the security of each encryption scheme.— Encryption Algorithm: First, the message (m) is converted into a string of bits. Then, for every bit of the message mi , a quadratic nonresidue value yi is produced such that дcd (yi , n) = 1. Then, each bit is encrypted to ci as follows: ci = E (mi ) = y2xmi (mod n),

∀mi = {0, 1}, ………………………………………….(5)

where m = m0m1 ... mr , c = c0c1 ... cr , and r is the block size used for the message space and x is picked from Zn∗ at random for every encryption, where Zn∗ is the multiplicative subgroup of integers modulo n that includes all the numbers smaller than r and relatively prime to r .— Decryption Algorithm: Since x is picked from the set Zn∗ (1 < x   n   1), x is quadratic residue modulo n for only mi = 0. Hence, to decrypt the ciphertext ci , one decides whether ci is a quadratic residue modulo n or not; if so, mi returns 0, or else mi returns 1. — Homomorphic Property: For each bit mi ∈ {0, 1}, E (m1) ∗ E (m2) = (y2xm1 (mod n)) ∗ (y2xm2 (mod n)) ……………………………………………………(6) = (y1 ∗ y2)2xm1+m2 (mod n) = E (m1 + m2). The homomorphic property of the GM cryptosystem shows that encryption of the sum E (m1 m2) can be  directly  calculated  from  the  separately encrypted  bits, E (m1) and E (m2). Since  the message and ciphertext are the elements of the set 0, 1 , the operation is the same with exclusive- OR (XOR).3 Hence, GM is homomorphic over only addition for binary numbers.

**2.3.2.4 El-Gamal. In 1985, Taher Elgamal proposed a new public key encryption scheme (ElGamal 1985),** which is the improved version of the original Diffie-Hellman Key Exchange (Diffie and Hellman 1976) algorithm, which is based on the hardness of certain problems in discrete logarithm (Kevin 1990). It is mostly used in hybrid encryption systems to encrypt the secret key of a symmetric encryption system. The El-Gamal cryptosystem is defined as follows: — KeyGen Algorithm: A cyclic group G with order n using generator д is produced. In a cyclic group, it is possible to generate all the elements of the group using the powers of one of its own element. Then, h = дy is computed for randomly chosen y   Zn∗. Finally, the public key is (G, n, д, h) and x is the secret key of the scheme. — Encryption Algorithm: The message m is encrypted using д and x , where x is randomly chosen from the set 1, 2, ... , n 1 and the output of the encryption algorithm is a ciphertext pair (c = (c1, c2)): c = E (m) = (дx , mhx ) = (дx , mдxy ) = (c1, c2). ………………………………….(7)

— Decryption Algorithm: To decrypt the ciphertext c, first, s = c1y is computed, where y is the secret key. Then, the decryption algorithm works as follows:

c2 · s−1 = mдxy · д−xy = m…………………………. (8)

—Homomorphic Property: E (m1) ∗ E (m2) = (дx1 , m1hx1 ) ∗ (дx2 , m2hx2 ) = (дx1+x2 , m1 ∗ m2hx1+x2 ) = E (m1 ∗ m2). ……………………………..(9)

As seen from this derivation, the El-Gamal cryptosystem is multiplicatively homomorphic. It does not support addition operations over ciphertexts. 3XOR can be thought of as binary addition.  
  
 **2.3.2.5** **Benaloh. Benaloh proposed an extension of the GM cryptosystem by improving it to encrypt the message as a block instead of bit by bit (Benaloh 1994).** Benaloh’s proposal was based on the higher residuosity problem. The higher residuosity problem (xn) (Zheng *et al.* 1988) is the generalization of quadratic residuosity problems (x2) that is used for the GM cryptosystem.— KeyGen Algorithm: Block size r and large primes p and q are chosen such that r divides p 1 and r is relatively prime to (p  1)/r and q  1 (i.e., дcd (r , (p  1)/r ) = 1 and дcd (r , (q  1)) = 1). Then, n = pq and ϕ = (p  1) (q  1) are computed. Lastly, y  Zn∗ is chosen such that yϕ  1 mod n, where Zn∗ is the multiplicative subgroup of integers modulo n that includes all the numbers smaller than r and relatively prime to r . Finally, (y, n) is published as the public key, and (p, q) is kept as the secret key. — Encryption Algorithm: For the message m ∈ Zr , where Zr = {0, 1, ... , r − 1}, choose a ran- dom u such that u ∈ Zn∗. Then, to encrypt the message m:  
c = E (m) = ymur (mod n), (10) where the public key is the modulus n and base y with the block size of r . — Decryption Algorithm: The message m is recovered by an exhaustive search for i Zr such that (y−ic )ϕ/r ≡ 1, …………………………(11)  
where the message m is returned as the value of i, i.e., m = i. — Homomorphic Property:  
E (m1) ∗ E (m2) = (ym1u1r (mod n)) ∗ (ym2u2r (mod n)) = ym1+m2 (u1 ∗ u2)r (mod n) = E (m1 + m2 (mod n)). …………………………………(12)

The homomorphic property of Benaloh shows that any multiplication operation on encrypted data corresponds to the addition on plaintext. As the encryption of the addition of the messages can directly be calculated from encrypted messages E (m1) and E (m2), the Benaloh cryptosystem is additively homomorphic.

**2.3.2.6 Paillier. In 1999, Paillier (1999) introduced another novel probabilistic encryption** scheme based on the composite residuosity problem (Jager 2012). The composite residuosity problem is very similar to quadratic and higher residuosity problems that are used in GM and Benaloh cryptosys- tems. It questions whether there exists an integer x such that xn ≡ a (mod n2) for a given integer a. — KeyGen Algorithm: For large primes p and q such that дcd (pq, (p 1) (q  1)) = 1, com- pute n = pq and λ = lcm(p 1, q 1). Then, select a random integer д Z n2 by checking whether дcd (n, L(дλ mod n2 )) = 1, where the function L is defined as L(u ) = (u 1)/n for every u  from the subgroup Zn∗2   that is a multiplicative subgroup of integers modulo n2 in- stead of n as in the Benaloh cryptosystem. Finally, the public key is (n, д) and the secret key is a (p, q) pair. — Encryption Algorithm For each message m, the number r is randomly chosen and the encryption works as follows:

c = E (m) = дmrn (mod n2). …………………………….(13)  
— Decryption Algorithm: For a proper ciphertext c < n2, the decryption is done by L(cλ (mod n2)) D (c ) = L(дλ (mod n2 )) mod n = m, ………….(14)

where the private key pair is (p, q). — Homomorphic Property: E (m1) ∗ E (m2) = (дm1r1n (mod n2)) ∗ (дm2r2n (mod n2)) = дm1+m2 (r1 ∗ r2)n

(mod n2) = E (m1 + m2)………………………………... (15) This derivation shows that Pailliler’s encryption scheme is homomorphic over addition. In addition to homomorphism over the addition operation, Pailliler’s encryption scheme has some additional homomorphic properties, which allow extra basic operations on plaintexts m1, m2 Z ∗2 by using the encrypted plaintexts E (m1) and E (m2) and public key pair (n, д): E (m1) ∗ E (m2) (mod n2) = E (m1 + m2 (mod n)), …………………………………………..(16)  
E (m1) ∗ дm2 (mod n2) = E (m1 + m2 (mod n)), ……..(17)  
E (m1)m2  (mod n2) = E (m1m2  (mod n)). (18)  
These additional homomorphic properties describe different cross-relations between various operations on the encrypted data and the plaintexts. In other words, Equations (16), (17), and (18) show how the operations computed on encrypted data affect the plaintexts.  
3.1.6 Others. Moreover, Okamoto-Uchiyama (OU) (Okamoto and Uchiyama 1998) proposed a new PHE scheme to improve the computational performance by changing the set, where the en- cryptions of previous HE schemes work. The domain of the scheme is the same as the previous public key encryption schemes, Zn∗ ; however, Okamoto-Uchiyama sets n = p2q  for large primes p and q. Furthermore, Naccache-Stern (NS) (Naccache and Stern 1998) presented another PHE scheme as a generalization of the Benaloh cryptosystem to increase its computational efficiency. The proposed work changed only the decryption algorithm of the scheme. Likewise, Damgard- Jurik (DJ) (Damgård & Jurik 2021) introduced another PHE scheme as a generalization of Paillier. These three cryptosystems preserve the homomorphic property while improving the original homomorphic schemes. Similarly, Kawachi (KTX) *et al.* (2017) suggested an additively homomorphic encryption scheme over a large cyclic group, which is based on the hardness of underlying lattice problems. They named the homomorphic property of their proposed scheme as pseudohomomorphic. Pseudohomo- morphism is an algebraic property and still allows homomorphic operations on ciphertext; how- ever, the decryption of the homomorphically operated ciphertext works with a small decryption error. Finally, Galbraith (2022) introduced a more natural generalization of Paillier’s cryptosystem, applying it on elliptic curves while still preserving the homomorphic property of Paillier’s cryptosystem. Homomorphic properties of well-known PHE schemes are briefly summarized in Table 1.

**2.2.3 Somewhat Homomorphic Encryption Schemes**

There are useful SWHE examples (Yao 1982; Sander *et al.* 1999; Boneh *et al.* 2015; Ishai & Paskin 2017) in the literature before 2009. After the first plausible FHE published in 2009 (Gentry 2009), some SWHE versions of FHE schemes were also proposed because of the performance issues asso- ciated with FHE schemes. We cover these SWHE schemes under the FHE section. In this section, we primarily focus on major SWHE schemes, which were used as a stepping stone to the first plausible FHE scheme.

**3.2.3.1 BGN. Before 2005**, all proposed cryptosystems’ homomorphism properties were re- stricted to only either addition or multiplication operations, i.e., SWHE schemes. One of the most significant steps toward an FHE scheme was introduced by Boneh-Goh-Nissim (BGN) in Boneh

Table1.Homomorphic Properties of Well-Known PHE Schemes Homomorphic Operation Scheme Add Mul RSA (Rivest *et al.* 1978b) GM (Goldwasser & Micali 1982) El-Gamal (ElGamal 1985)4 Benaloh (Benaloh 1994) NS (Naccache and Stern 1998)

OU (Okamoto and Uchiyama 1998) Paillier (Paillier 1999) DJ (Damgård & Jurik 2021)  
KTX (Kawachi *et al.* 2017) Galbraith (Galbraith 2002). BGN evaluates 2-DNF5 formulas on ciphertext and it supports an arbitrary number of additions and one multiplication by keeping the ciphertext size constant. The hardness of the scheme is based on the subgroup decision problem (Gjøsteen 2014). The subgroup decision problem simply decides whether an element is a member of a subgroup Gp of group G of Composite order n = pq, where p and q are distinct primes.— KeyGen Algorithm: The public key is released as (n, G, G1, e, д, h). In the public key, e is a bilinear map such that e : G  G  G1, where G, G1 are groups of order n = q1q2. д and u are the generators of G and set h = uq2 and h is the generator of G with order q1, which is kept hidden as the secret key.— Encryption Algorithm: To encrypt a message m, a random number r from the set {0, 1, ... , n − 1} is picked and encrypted using the precomputed д and h as follows:

c = E (m) = дmhr  mod n. …………………………………………..(19)  
— Decryption Algorithm: To decrypt the ciphertext c, one first computes c J = cq1 = (дmhr )q1 = (дq1 )m (note that hq1 1 mod n) and дJ = дq1 using the secret key q1 and decryption is completed as follows:

m = D (c ) = logдJ c J . ………………………………………….(20)  
In order to decrypt efficiently, the message space should be kept small because of the fact that the discrete logarithm cannot be computed quickly.  
— Homomorphism over Addition: Homomorphic addition of plaintexts m1 and m2 using cipher- texts E (m1) = c1 and E (m2) = c2 are performed as follows:

c = c1c2hr = (дm1hr1 ) (дm2hr2 )hr = дm1+m2hrJ, ……………….(21)  
where r = r1 + r2 + r and it can be seen that m1 + m2 can be easily recovered from the re- sulting ciphertext c. — Homomorphism over Multiplication: To perform homomorphic multiplication, use д1 with order n and h1 with order q1 and set д1 = e (д, д), h1 = e (д, h), and h = дαq2 . Then, the homomorphic multiplication of messages m1 and m2 using the ciphertexts c1 = E (m1) and  
  
4The method to convert El-Gamal into an additively homomorphic encryption scheme is shown in Cramer et al. (1997). However, it is still PHE as it still supports only addition operation, not both at the same time. 5Disjunctive Normal Form with at most two literals in each clause.  
c2 = E (m2) are computed as follows:

c = e (c1, c2)h1r = e (дm1hr1 , дm2hr2 )h1r = д1m1m2 h1m1r2 +r2m1 +αq2r1r2 +r  = д1m1m2 h1r J . ……………………………………………………………………(22)  
It is seen that r J is uniformly distributed like r and so m1m2 can be correctly recovered from resulting ciphertext c. However, c is now in the group G1 instead of G. Therefore, another homo- morphic multiplication operation is not allowed in G1 because there is no pairing from the set G1. However, resulting ciphertext in G1 still allows an unlimited number of homomorphic additions. Moreover, Boneh et al. also showed the evaluation of 2-DNF formulas using the basic 2-DNF pro- tocol. Their protocol gives a quadratic improvement in terms of the protocol complexity over Yao’s well-known garbled circuit protocol in Yao (1982).

|  |  |  |
| --- | --- | --- |
|  | **HomomorphicOperation** | |
| **Scheme** | **Add** | **Mult** |
| RSA (Rivest et al. [1978b](#_bookmark179))  GM (Goldwasser and Micali[1982](#_bookmark112)) El-Gamal(ElGamal [1985](#_bookmark93))4  Benaloh (Benaloh [1994](#_bookmark40))  NS (Naccache and Stern[1998](#_bookmark159))  OU (Okamoto and Uchiyama [1998](#_bookmark164)) Paillier (Paillier[1999](#_bookmark166))  DJ(Damgård and Jurik [2001](#_bookmark80))  KTX (Kawachi et al. [2007](#_bookmark126)) Galbraith (Galbraith [2002](#_bookmark99)) |  | ✔ |
| ✔ |  |
|  | ✔ |
| ✔ |  |
| ✔ |  |
| ✔ |  |
| ✔ |  |
| ✔ |  |
| ✔ |  |
| ✔ |  |

Table1.Homomorphic Properties of Well-Known PHE Schemes

**2.3.2.2 Others. In the literature of HE schemes, one of the first SWHE schemes is the Polly Cracker scheme (Fellows & Koblitz 1994).** It allows both multiplication and addition operations over the ciphertexts. However, the size of the ciphertext grows exponentially with the homomorphic operation, and the multiplication operation is especially extremely expensive. Later more efficient variants (Levy-dit Vehel & Perret 2014; Van Ly 2016) are proposed, but almost all of them are later shown vulnerable to attacks (Steinwandt 2010; Levy-dit Vehel *et al*. 2019). There- fore, they are either insecure or impractical (Le 2013). Recently, Albrecht *et al.* (2011) introduced a Polly Cracker with Noise cryptosystem, where the homomorphic addition operations do not increase the ciphertext size, while the multiplications square it.  
Another idea of evaluating operations on encrypted data is realized over different sets. Sander, Young, and Yung (SYY) described the first SWHE scheme over a semigroup, NC1, 6 (Sander *et al*. 1999), which requires fewer properties than a group. NC1 is a complexity class that includes the circuits with polylogarithmic depth and polynomial size. The proposed scheme supported polyno- mially many ANDing of ciphertexts with one OR/NOT gate. However, the ciphertext size increased by a constant multiplication with each OR/NOT gate evaluation. This increase limits the evalu- ation of circuit depth. Yuval Ishai and Anat Paskin (IP) expanded the set to branching programs (aka Binary Decision Diagrams), which are the directed acyclic graphs where every node has two outgoing edges with labeled binary 0 and 1 (Ishai & Paskin 2017). In other words, they proposed a public key encryption scheme by evaluating the branching programs on the encrypted data. More- over, Melchor *et al.* (2010) proposed a generic construction method to obtain a chained encryp- tion scheme allowing the homomorphic evaluation of constant depth circuits over ciphertext. The chained encryption scheme is obtained from well-known encryption schemes with some homo- morphic properties. For example, they showed how to obtain a combination of BGN (Boneh *et al.* 2015) and Kawachi *et al*. (2007). As mentioned before, BGN allows an arbitrary number of additions and one multiplication, while Kawachi’s scheme is only additively homomorphic. Hence, the resulting combined scheme allows arbitrary additions and two multiplications. They also showed how this procedure is applied to the scheme in Melchor *et al.* (2008) allowing a predefined number of homomorphic additions, to obtain a scheme that allow an arbitrary number of multiplications as well. However, in multiplication, ciphertext size grows exponentially, while it is constant in a homomorphic addition. The summary of some well-known SWHE schemes is given in Table 2. As shown in Table 2, while in Yao, SYY, and IP cryptosystems, the size of the ciphertext grows with each homomorphic operation; in BGN it stays constant. This property of BGN is a significant improvement to obtain an FHE scheme. Accordingly, Gentry, Halevi, and Vaikuntanathan later simplified the BGN cryptosystem (Gentry *et al*. 2010). In their version, the underlying 6NC stands for “Nick’s Class” for the honor of Nick Pippenger security assumption is changed to hardness of the LWE problem. The BGN cryptosystem chooses input from a small set to decrypt correctly. In contrast, a recent scheme introduced in Gentry *et al.* (2010) have much larger message space. Moreover, some of the attempts to obtain an FHE scheme based on SWHE schemes are reported as broken. For instance, vulnerabilities for Mullen and Shiue (1994), i Ferrer (1996), Grigoriev and Ponomarenko (2016), and Domingo-Ferrer (2002) were reported in Steinwandt and Geiselmann (2012), Choi *et al*. (2017), Wagner (2013), and Cheon *et al.* (2016), respectively.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Evaluation Size** | **Evaluation Circuit** | **Ciphertext Size** |
| Yao ([1982](#_bookmark211)) | arbitrary | Garbled circuit | Grows atleast l in early |
| SYY(Sander et al.[1999](#_bookmark184)) | Poly many AND  & one OR/NOT | *NC*1circuit | Grows exponentially |
| BGN (Boneh et al. [2015](#_bookmark41)) | Unlimited add &  1mult | 2-DNFformulas | constant |
| IP (Ishai and Paskin [2017](#_bookmark123)) | arbitrary | Branching programs | doesn’t depend on the  size of function |

Table2.Comparison of Some Well-Known SWHE Schemes before Gentry’s Work

**2.4 Fully Homomorphic Encryption Schemes**

An encryption scheme is called an FHE scheme if it allows an unlimited number of evaluation operations on the encrypted data and resulting output is within the ciphertext space. After almost 30 years from the introduction of privacy homomorphism concept (Rivest *et al*. 1978a), Gentry presented the first feasible proposal in his seminal PhD thesis to a long-term open problem, which is obtaining an FHE scheme (Gentry 2009). Gentry’s proposed scheme gives not only an FHE scheme but also a general framework to obtain an FHE scheme. Hence, a lot of researchers have attempted to design a secure and practical FHE scheme after Gentry’s work.  
Although Gentry’s proposed ideal lattice-based FHE scheme (Gentry 2009) is very promising.

**2.5 Reviewed Literature Matrix**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Reference** | **Title** | **Authors** | **Year** | **Journal/conference** | **Findings/ contributions** |
| 1 | Fully Homomorphic Encryption Over the Integers | Craig Gentry | 2019 | STOCIntroduced the first fully homomorphic encryption scheme based on lattice-based cryptography | Introduced the first fully homomorphic encryption scheme based on lattice-based cryptography |
| 2 | A Survey of Homomorphic Encryption for Secure Computation | Pascal Paillier | 2010 | IEEE | Security & PrivacyProvides an overview of homomorphic encryption schemes and their applications in secure computation |
| 3 | Practical Homomorphic Encryption Schemes | Marten van Dijk | 2010 | CRYPTO | Proposes practical homomorphic encryption schemes based on ring-LWE and polynomial approximation techniques. |
| 4 | Cryptographic Cloud Computing: Challenges and Opportunities. | Seny Kamara | 2010 | IEEE | Internet ComputingDiscusses the challenges and opportunities of using homomorphic encryption in cloud computing environments |
| 5 | Homomorphic Encryption: A Survey | Zvika Brakerski | 2014 | IEEE | Security & PrivacyPresents a comprehensive survey of homomorphic encryption schemes, including FHE, PHE, and LHE |
| 6 | A Comprehensive Review on Homomorphic Encryption Techniques | Muhammad Asim | 2017 | Computers & Security | Reviews various homomorphic encryption techniques and their applications in secure computation |
| 7 | Homomorphic Encryption in Healthcare: A Systematic Review | Rachel Cummings | 2018 | Journal of Medical Internet Research | Investigates the use of homomorphic encryption in healthcare applications and its impact on data privacy. |
| 8 | Secure Outsourced Computation via Fully Homomorphic Encryption | Shai Halevi | 2018 | ACM | Computing SurveysExamines the feasibility and practicality of outsourcing computations using fully homomorphic encryption. |
| 9 | Homomorphic Encryption for Machine Learning: A Review | Clara Schneidewind | 2020 | Machine Learning | Explores the application of homomorphic encryption in machine learning tasks and its potential for privacy-preserving data analysis |
| 10 | Advances and Challenges in Homomorphic Encryption for Privacy-Preserving Machine Learning | Cheng Chen | 2021 | IEEE | Transactions on Services ComputingDiscusses recent advances and challenges in using homomorphic encryption for privacy-preserving machine learning |

Table 3: shows the Reviewed literature matrix

Source: Research survey, 2023

**CHAPTER THREE: METHODOLOGY**

* 1. **Introduction**

This chapter describes the research design and approach, research setting, and validity and reliability.

* 1. **Research Approach:**

**3.2.1 Exploratory Approach:** this study will employ an exploratory approach which will give the evolving nature of homomorphic encryption in identify challenges, and explore potential solutions (Smart, 2019).

* 1. **Research Design:**

**Mixed-Methods Design:** in this study we are going to use both quantitative and qualitative methods to allow for a comprehensive investigation.

The quantitative methods will be used to measure performance metrics, while qualitative methods will be use to capture user perceptions and usability aspects (Halevi & Shoup. 2023a)

* 1. **Data Collection Methods:**

**3.4.2 Interviews and Surveys:** in this study data will the collected by engage with experts in cryptography, data security, and related fields to gather expert opinions, insights, and feedback on homomorphic encryption approaches and their practical implications (Smart, 2019).

**3.4.3 Experimental Data:** experimental will be collected through simulations, prototyping, or real world experiments to evaluate the performance, security, and usability of homomorphic encryption schemes (Smart, 2019).

**3.5 Data Analysis Techniques:**

**3.5.1 Quantitative Analysis:** data will be analyzed quantitatively using statistical techniques to measure performance metrics (encryption/decryption speed, computational overhead) and security properties (resistance to cryptographic attacks) (Morris,2023).

**3.5.2 Qualitative Analysis:** qualitative data will be analyze from interviews, surveys and user feedback to identify themes, patterns, and insights related to user perceptions, usability issues, and practical challenges (Morris,2023).

**3.6 Prototype Development:**

**3.6.1 Software Prototyping:** proof-of-concept will be design for implementations of homomorphic encryption frameworks using programming languages and cryptographic libraries.

**3.6.2 Simulation Studies:** simulation studies will be conducted to evaluate the performance and scalability of homomorphic encryption algorithms under various scenarios and parameters (Halevi & Shoup. 2023a)

**3.8 Validation and Verification:**

**3.8.1 Formal Verification:** a formal methods and verification techniques will be use to validate the correctness and robustness of homomorphic encryption algorithms or frameworks (Yuanmi & Phong. 2022).

**3.8.2 Experimental Validation:** the performance, security, and usability of homomorphic encryption schemes will be validated through experimental studies, comparing against existing approaches or benchmarks (Yuanmi & Phong. 2022).

**References**

Bogos, S., Gaspoz, J., &Vaudenay, S. (2017). Cryptanalysis of a Homomorphic Encryption Scheme. In *ArcticCrypt 2016*. Retrieved from <https://infoscience.epfl.ch/record/220692>

Cramer, R., Damgård, I., Nielsen, J.B (2021) Multiparty computation from threshold homomorphic encryption. In: International Conference on the Theory and Applications of Cryptographic Techniques, pp. 280–300

Chen, H., *et al* (2018): Logistic regression over encrypted data from fully homomorphic encryption. BMC Med. Genomics 11(4), 3–12.

Delfs, H., & Knebl, H. (2017). *Introduction to Cryptography: Principles and Applications*. Springer Science & Business Media.

dos Santos, L. C., Bilar, G. R., & Pereira, F. D. (2019). Implementation of the fully homomorphic encryption scheme over integers with shorter keys. In *New Technologies, Mobility and Security (NTMS), 2015 7th International Conference on* (pp. 1–5). IEEE. Retrieved from <http://ieeexplore.ieee.org/abstract/document/7266495/>

Fontaine, C., & Galand, F. (2017). A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, *2007*(1), 1–10.

Fan, J., Vercauteren, F. (2022): Somewhat practical fully homomorphic encryption. IACR Cryptol. ePrint Arch. 2012, 144.

Greenberg, A.: Hacker lexicon (2017). what is homomorphic encryption.

Gentry, C (2019).: A fully homomorphic encryption scheme, vol. 20(9). Stanford University Stanford.

Gai, K., Qiu, M. (2017): Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers. IEEE Trans. Ind. Inform. 14(8), 3590–3598

Gentry, C., &Halevi, S. (2021). Implementing Gentry’s fully-homomorphic encryption scheme. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 129–148). Springer. Retrieved from <http://link.springer.com/chapter/10.1007/978-3-642-20465-4_9>

Halevi, S., &Shoup, V. (2017). Algorithms in helib. In *International Cryptology Conference* (pp. 554–571). Springer. Retrieved from http://link.springer.com/chapter/10.1007/978-3-662-44371-2\_31

Liu, D. (2019). Practical Fully Homomorphic Encryption without Noise Reduction. *IACR Cryptology ePrint Archive*, *2015*, 468.

Morris, L. (2023): Analysis of partially and fully homomorphic encryption. Rochester Inst. Technol. 1–5.

Mazonka, O., Tsoutsos, N. G., & Maniatakos, M. (2016). Cryptoleq: A heterogeneous abstract machine for encrypted and unencrypted computation. *IEEE Transactions on Information Forensics and Security*, *11*(9), 2123–2138.

Moore, C., O’Neill, M., O’Sullivan, E., Doroz, Y., &Sunar, B. (2018). Practical homomorphic encryption: A survey. In *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on* (pp. 2792–2795). IEEE. Retrieved from <http://ieeexplore.ieee.org/abstract/document/6865753/>

Smart, N. (2019). *Cryptography Made Simple*. Springer.

Tilborg, H. C. A. van, & Jajodia, S. (2019). *Encyclopedia of Cryptography and Security*. Springer Science & Business Media.

Yousuf, H., Lahzi, M., Salloum, S.A., Shaalan, K.(2021). Systematic review on fully homomorphic encryption scheme and its application. In: Al-Emran, M., Shaalan, K., Hassanien, A. (eds.) Recent Advances in Intelligent Systems and Smart. Studies in Systems, Decision and Control, vol. 295, pp. 537–551. Springer, Cham.

Zhou, J., & Yung, M. (2010). *Applied Cryptography and Network Security: 8th International Conference, ACNS 2010, Beijing, China, June 22-25, 2010, Proceedings*. Springer Science & Business Media. <https://www.ivoryresearch.com/writers/olive> bentham/https://www.ivoryresearch.com.

Nitesh Aggarwal, Cp Gupta, and Iti Sharma. (2014). Fully homomorphic symmetric scheme without bootstrapping. In 2014 International Conference on Cloud Computing and Internet of Things (CCIOT’14). IEEE, 14–17.

Carlos Aguilar-Melchor, Simon Fau, Caroline Fontaine, Guy Gogniat, and Renaud Sirdey. (2023). Recent advances in homo- morphic encryption: A possible future for signal processing in the encrypted domain. IEEE Signal Processing Magazine 30, 2 (2023), 108–117.

S. Sobitha Ahila and K. L. Shunmuganathan. (2014). State Of art in homomorphic encryption schemes. International Journal of Engineering Research and Applications 4, 2 (2014), 37–43.

Miklós Ajtai. (1996). Generating hard instances of lattice problems. In Proceedings of the 28th Annual ACM Symposium on Theory of Computing. ACM, 99–108.

Martin Albrecht, Shi Bai, and Léo Ducas. (2016). A subfield lattice attack on overstretched NTRU assumptions. In Annual Cryptology Conference. Springer, 153–178.  
Martin Albrecht, Pooya Farshim, Jean-Charles Faugere, and Ludovic Perret. 2011. Polly cracker, revisited. Advances in Cryptology (ASIACRYPT’11), 179–196.

Martin R. Albrecht. 2017. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 103–129.

Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. 2015. Ciphers for MPC and FHE. In Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 430– 454.

Jacob Alperin-Sheriff and Chris Peikert. 2023. Practical bootstrapping in quasilinear time. In Advances in Cryptology (CRYPTO’13). Springer, 1–20.

Jacob Alperin-Sheriff and Chris Peikert. 2014. Faster bootstrapping with polynomial error. In Advances in Cryptology (CRYPTO’14). Springer, 297–314.

Frederik Armknecht, Colin Boyd, Christopher Carr, Kristian Gjøsteen, Angela Jäschke, Christian A. Reuter, and Martin Strand. 2015. A guide to fully homomorphic encryption. IACRCryptology ePrint Archive 2015 (2015), 1192.

rederik Armknecht, Stefan Katzenbeisser, and Andreas Peter. 2023. Group homomorphic encryption: Characterizations, impossibility results, and applications. Designs, Codes and Cryptography 67, 2 (2013), 209–232.

Josh Benaloh. 1994. Dense probabilistic encryption. In Proceedings of the Workshop on Selected Areas of Cryptography. 120– 128. Josh Daniel Cohen Benaloh. 1987. Verifiable Secret-Ballot Elections. Yale University, Department of Computer Science. Dan Boneh, Eu-Jin Goh, and

Kobbi Nissim. 2015. Evaluating 2-DNF formulas on ciphertexts. In Theory of Cryptography.  
Springer, 325–341.

Joppe W. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. 2023. Improved security for a ring-based fully homomorphic encryption scheme. In Cryptography and Coding. Springer, 45–64.

Zvika Brakerski. 2022. Fully homomorphic encryption without modulus switching from classical GapSVP. In Advances in Cryptology (CRYPTO’12). Springer, 868–886.

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. 2011. Fully homomorphic encryption without bootstrapping.  
Cryptology ePrint Archive, Report 2011/277. Retrieved from <http://eprint.iacr.org/>.

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. 2014. (Leveled) fully homomorphic encryption without boot- strapping. ACM Transactions on Computation Theory 6, 3, Article 13 (July 2014), 36 pages. DOI:http://dx.doi.org/10.1145/2633600

Zvika Brakerski and Vinod Vaikuntanathan. 2021. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Advances in Cryptology (CRYPTO’11). Springer, 505–524.

Zvika Brakerski and Vinod Vaikuntanathan. 2014a. Efficient fully homomorphic encryption from (standard) LWE. SIAM Journal on Computing 43, 2 (2014), 831–871.

Zvika Brakerski and Vinod Vaikuntanathan. 2014b. Lattice-based FHE as secure as PKE. In Proceedings of the 5th Conference on Innovations in Theoretical Computer Science. ACM, 1–12.

Xiaolin Cao, Ciara Moore, Máire O’Neill, Elizabeth O’Sullivan, and Neil Hanley. 2023. Accelerating fully homomorphic encryption over the integers with super-size hardware multiplier and modular reduction. IACR Cryptology ePrint Archive 2013 (2023), 616.

Xiaolin Cao, Ciara Moore, Máire O’Neill, Neil Hanley, and Elizabeth OSullivan. 2014. High-speed fully homomorphic en- cryption over the integers. In Financial Cryptography and Data Security. Springer, 169–180.

Henry Carter, Benjamin Mood, Patrick Traynor, and Kevin Butler. 2023. Secure outsourced garbled circuit evaluation for mobile devices. Journal of Computer Security 24, 2 (2013), 137–180.

Henry Carter, Benjamin Mood, Patrick Traynor, and Kevin Butler. 2015. Outsourcing secure two-party computation as a black box. Security and Communication Networks 9, 14 (2015), 2261–2275.

Donald Donglong Chen, Nele Mentens, Frederik Vercauteren, Sujoy Sinha Roy, Ray C. C. Cheung, Derek Pao, and Ingrid Verbauwhede. 2015. High-speed polynomial multiplication architecture for ring-LWE and SHE cryptosystems. IEEE Transactions on Circuits and Systems I: Regular Papers, 62, 1 (2015), 157–166.

Hao Chen, Kim Laine, and Rachel Player. 2017. Simple Encrypted Arithmetic Library. Retrieved from https://www. microsoft.com/en-us/research/wp content/uploads/2017/06/sealmanual\_v2.2.pdf (accessed September 2017).

Liquan Chen, Hongmei Ben, and Jie Huang. 2014. An encryption depth optimization scheme for fully homomorphic en- cryption. In 2014 International Conference on Identification, Information and Knowledge in the Internet of Things (IIKI’14). IEEE, 137–141.

Yuanmi Chen and Phong Q. Nguyen. 2022. Faster algorithms for approximate common divisors: Breaking fully- homomorphic-encryption challenges over the integers. In Advances in Cryptology (EUROCRYPT’12). Springer, 502–519.

Zhigang Chen, Jian Wang, ZengNian Zhang, and Song Xinxia. 2014. A fully homomorphic encryption scheme with better key size. Communications, China 11, 9 (2014), 82–92.

Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrède Lepoint, Mehdi Tibouchi, and Aaram Yun. 2013. Batch fully homomorphic encryption over the integers. In Advances in Cryptology (EUROCRYPT’13). Springer, 315–335.

Jung Hee Cheon, Hyunsook Hong, Moon Sung Lee, and Hansol Ryu. 2016. The polynomial approximate common divisor problem and its application to the fully homomorphic encryption. Information Sciences 326 (2016), 41–58.

Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. 2016. Homomorphic encryption for arithmetic of approxi- mate numbers (HEANN). Retrieved from https://github.com/kimandrik/HEAAN (accessed September 2017).

Jung Hee Cheon, Woo-Hwan Kim, and Hyun Soo Nam. 2006. Known-plaintext cryptanalysis of the domingo-ferrer alge- braic privacy homomorphism scheme. Information Processing Letters 97, 3 (2016), 118–123.

Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. 2016. Faster fully homomorphic encryption: Boot- strapping in less than 0.1 seconds. In Advances in Cryptology (ASIACRYPT’16): 22nd International Conference on the Theory and Application of Cryptology and Information Security, Proceedings, Part I 22. Springer, 3–33.

Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. 2017. TFHE: Fast fully homomorphic encryption library over the Torus. Retrieved from https://github.com/tfhe/tfhe (accessed September 2017).

Su-Jeong Choi, Simon R. Blackburn, and Peter R. Wild. 2007. Cryptanalysis of a homomorphic public-key cryptosystem over a finite group. Journal of Mathematical Cryptology 1, 4 (2017), 351.

Ashish Choudhury, Jake Loftus, Emmanuela Orsini, Arpita Patra, and Nigel P. Smart. 2023. Between a rock and a hard place: Interpolating between MPC and FHE. In Advances in Cryptology (ASIACRYPT’13). Springer, 221–240.

Michael Clear and Ciarán McGoldrick. 2014. Bootstrappable identity-based fully homomorphic encryption. In Cryptology and Network Security. Springer, 1–19.

Michael Clear and Ciarán McGoldrick. 2015. Multi-identity and multi-key leveled FHE from learning with errors. In Annual Cryptology Conference. Springer, 630–656.

Michael Clear and Ciarán McGoldrick. 2016. Attribute-based fully homomorphic encryption with a bounded number of inputs. In International Conference on Cryptology in Africa. Springer, 307–324.

Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. 2013. Batch fully homomorphic encryption over the integers. Cryptology ePrint Archive, Report 2013/036. Retrieved from <http://eprint.iacr.org/>.

Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. 2014. Scale-invariant fully homomorphic encryption over the integers. In Public-Key Cryptography (PKC’14). Springer, 311–328.

Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi. 2021. Fully homomorphic encryption over the integers with shorter public keys. In Advances in Cryptology (CRYPTO’11). Springer, 487–504.

Jean-Sébastien Coron, David Naccache, and Mehdi Tibouchi. 2022. Public key compression and modulus switching for fully homomorphic encryption over the integers. In Advances in Cryptology (EUROCRYPT’12). Springer, 446–464.

David Bruce Cousins, John Golusky, Kurt Rohloff, and Daniel Sumorok. 2014. An FPGA co-processor implementation of Homomorphic Encryption. In 2014 IEEE High Performance Extreme Computing Conference (HPEC’14). IEEE, 1–6.

David Bruce Cousins, Kathrin Rohloff, Chris Peikert, and Richard Schantz. 2022. An update on SIPHER (scalable imple- mentation of primitives for homomorphic encryption) FPGA implementation using Simulink. In 2022 IEEE Conference on High Performance Extreme Computing (HPEC’12). IEEE, 1–5.

Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. 1997. A secure and optimally efficient multi-authority election scheme. Transactions on Emerging Telecommunications Technologies 8, 5 (1997), 481–490.

Wei Dai, Yarkın Doröz, and Berk Sunar. 2014. Accelerating NTRU based homomorphic encryption using GPUs. In 2014 IEEE High Performance Extreme Computing Conference (HPEC’14). IEEE, 1–6.

Wei Dai, Yarkın Doröz, and Berk Sunar. 2015. Accelerating SWHE based PIRs using GPUs. In International Conference on Financial Cryptography and Data Security. Springer, 160–171.

Wei Dai, Yarkın Doröz, and Berk Sunar. 2017. cuHE: Homomorphic and fast. Retrieved from https://github.com/vernamlab/ cuHE (accessed September 2017).

Ivan Damgård and Mads Jurik. 2021. A generalisation, a simplification and some applications of Paillier’s probabilistic public-key system. In Public Key Cryptography. Springer, 119–136.

Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias. 2012. Multiparty computation from somewhat homomor- phic encryption. In Advances in Cryptology (CRYPTO’12). Springer, 643–662.

Ivan Damgård, Antigoni Polychroniadou, and Vanishree Rao. 2016. Adaptively secure multi-party computation from LWE (via equivocal FHE). In Public-Key Cryptography (PKC’16). Springer, 208–233.

Whitfield Diffie and Martin E. Hellman. 1976. New directions in cryptography. IEEE Transactions on Information Theory 22, 6 (1976), 644–654.

Josep Domingo-Ferrer. 2002. A provably secure additive and multiplicative privacy homomorphism. In Information Security. Springer, 471–483.

Yarkın Doröz, Yin Hu, and Berk Sunar. 2014. Homomorphic AES evaluation using NTRU. IACR Cryptology ePrint Archiven 2014 (2014), 39.

Yarkın Doröz, Erdinç Öztürk, Erkay Savaş, and Berk Sunar. 2015b. Accelerating LTV based homomorphic encryption in reconfigurable hardware. In Cryptographic Hardware and Embedded Systems (CHES’15). Springer, 185–204.

Yarkın Doröz, Erdinç Öztürk, and Berk Sunar. 2023. Evaluating the hardware performance of a million-bit multiplier. In 2023 Euromicro Conference on Digital System Design (DSD’13). IEEE, 955–962.

Yarkın Doröz, Erdinç Öztürk, and Berk Sunar. 2015a. Accelerating fully homomorphic encryption in hardware. IEEE Trans- actions on on Computers 64, 6 (2015), 1509–1521.

Yarkın Doröz, Aria Shahverdi, Thomas Eisenbarth, and Berk Sunar. 2014. Toward practical homomorphic evaluation of block ciphers using prince. In Financial Cryptography and Data Security. Springer, 208–220.

Yarkin Doröz and Berk Sunar. 2016. Flattening NTRU for evaluation key free homomorphic encryption. IACR Cryptology ePrint Archive 2016 (2016), 315.

Léo Ducas and Daniele Micciancio. 2014. A fully homomorphic encryption library. Retrieved from https://github.com/ lducas/FHEW (accessed December 2015).

Léo Ducas and Daniele Micciancio. 2015. FHEW: Bootstrapping homomorphic encryption in less than a second. In Advances in Cryptology (EUROCRYPT’15). Springer, 617–640.

Taher ElGamal. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. In Advances in Cryptology. Springer, 10–18.

Junfeng Fan and Frederik Vercauteren. 2022a. Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive 2022 (2022), 144.

Junfeng Fan and Frederik Vercauteren. 2012b. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144. (2022). Retrieved from <http://eprint.iacr.org/2012/144>.

Michael Fellows and Neal Koblitz. 1994. Combinatorial cryptosystems galore! Contemporary Mathematics 168 (1994), 51–51. Caroline Fontaine and Fabien Galand. 2017. A survey of homomorphic encryption for nonspecialists. EURASIP Journal on  
Information Security 2007 (2007), 15.

Matteo Frigo and Steven G. Johnson. 2015. The design and implementation of FFTW3. Proceedings of the IEEE 93, 2 (2015), 216–231. Special issue on “Program Generation, Optimization, and Platform Adaptation.”

Steven D. Galbraith. 2022. Elliptic curve paillier schemes. Journal of Cryptology 15, 2 (2022), 129–138.

Steven D. Galbraith, Shishay W. Gebregiyorgis, and Sean Murphy. 2016. Algorithms for the approximate common divisor problem. LMS Journal of Computation and Mathematics 19, A (2016), 58–72.

Craig Gentry. 2019. A Fully Homomorphic Encryption Scheme. Ph.D. Dissertation. Stanford University.

Craig Gentry. 2010. Toward basing fully homomorphic encryption on worst-case hardness. In Advances in Cryptology (CRYPTO’10). Springer, 116–137.

Craig Gentry. 2022. Personal communication.

Craig Gentry. 2014. Computing on the edge of chaos: Structure and randomness in encrypted computation. In Electronic Colloquium on Computational Complexity (ECCC’14), Vol. 21. 106.

Craig Gentry and Shai Halevi. 2021. Implementing gentrys fully-homomorphic encryption scheme. In Advances in Cryp- tology (EUROCRYPT’11). Springer, 129–148.

Craig Gentry, Shai Halevi, Chris Peikert, and Nigel P. Smart. 2022. Ring switching in BGV-style homomorphic encryption. In Security and Cryptography for Networks. Springer, 19–37.

Craig Gentry, Shai Halevi, and Nigel P. Smart. 2012. Homomorphic evaluation of the AES circuit. In Advances in Cryptology (CRYPTO’12). Springer, 850–867.

Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. 2010. A simple BGN-type cryptosystem from LWE. In Advances in Cryptology (EUROCRYPT’10). Springer, 506–522.

Craig Gentry, Amit Sahai, and Brent Waters. 2023. Homomorphic encryption from learning with errors: Conceptually- simpler, asymptotically-faster, attribute-based. In Advances in Cryptology (CRYPTO’13). Springer, 75–92.

Kristian Gjøsteen. 2014. Subgroup membership problems and public key cryptosystems. PhD thesis, Norwegian University of Science and Technology.

Oded Goldreich, Shafi Goldwasser, and Shai Halevi. 1997. Public-key cryptosystems from lattice reduction problems. In Advances in Cryptology (CRYPTO’97). Springer, 112–131.

Shafi Goldwasser and Silvio Micali. 1982. Probabilistic encryption & how to play mental poker keeping secret all partial information. In Proceedings of the 14th Annual ACM Symposium on Theory of Computing. ACM, 365–377.

Dima Grigoriev and Ilia Ponomarenko. 2016. Homomorphic public-key cryptosystems and encrypting boolean circuits.

Applicable Algebra in Engineering, Communication and Computing 17, 3–4 (2016), 239–255.

Shai Halevi and Nalini K. Ratha. 2021. Public challenges for fully-homomorphic encryption. Retrieved from http:// researcher.watson.ibm.com/researcher/view\_group.php?id=1548 (accessed March 2016).

Shai Halevi and Victor Shoup. 2023a. Design and implementation of a homomorphic-encryption library. IBM Research (Manuscript).

Shai Halevi and Victor Shoup. 2013b. An implementation of homomorphic encryption. Retrieved from https://github.com/ shaih/HElib (accessed December 2015).

Shai Halevi and Victor Shoup. 2014. Algorithms in helib. In Advances in Cryptology (CRYPTO’14). Springer, 554–571.

Shai Halevi and Victor Shoup. 2015. Bootstrapping for helib. In Advances in Cryptology (EUROCRYPT’15). Springer, 641–670. Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. 1998. NTRU: A ring-based public key cryptosystem. In Algorithmic  
Number Theory. Springer, 267–288.

Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman, and Joseph H. Silverman. 2018. An Introduction to Mathematical Cryp- tography. Vol. 1. Springer.

Darko Hrestak and Stjepan Picek. 2014. Homomorphic encryption in the cloud. In 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO’14). IEEE, 1400–1404.

Josep Domingo i Ferrer. 1996. A new privacy homomorphism and applications. Information Processing Letters 60, 5 (1996), 277–282. Yuval Ishai and Anat Paskin. 2007. Evaluating branching programs on encrypted data. In Theory of Cryptography. Springer, 575–594.

Tibor Jager. 2022. The Generic Composite Residuosity Problem. Vieweg+Teubner Verlag, Wiesbaden, 49–56. DOI:http://dx. doi.org/10.1007/978-3-8348-1990-1\_5

Burt Kaliski. 2015. Quadratic Residuosity Problem. Springer US, Boston, MA, 493–493. DOI:http://dx.doi.org/10.1007/ 0-387-23483-7\_336

Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. 2017. Multi-bit cryptosystems based on lattice problems. In Public Key Cryptography (PKC’07). Springer, 315–329.  
Kevin S. McC. 1990. The discrete logarithm problem. Cryptology and Computational Number Theory 42 (1990), 49.

Jinsu Kim, Moon Sung Lee, Aaram Yun, and Jung Hee Cheon. 2023. CRT-based fully homomorphic encryption over the integers. IACR Cryptology ePrint Archive 2013 (2013), 57.

Eyal Kushilevitz and Rafail Ostrovsky. 1997. Replication is not needed: Single database, computationally-private informa- tion retrieval. In Proceedings of the 38th Annual Symposium on Foundations of Computer Science. IEEE, 364.

Kim Laine, Hao Chen, and Rachel Player. 2017. Simple encrypted arithmetic library. Retrieved from https://sealcrypto. codeplex.com/ (accessed September 2017).

Van-Ly Le. 2013. Polly Two-a Public Key Cryptosystem Based on Polly Cracker. Ph.D. Dissertation. Ruhr University Bochum, Germany.

Moon Sung Lee. 2021. On the sparse subset sum problem from Gentry-Halevi’s implementation of fully homomorphic encryption. IACR Cryptology ePrint Archive 2021 (2021), 567.

Moon Sung Lee, Yongje Lee, Jung Hee Cheon, and Yunheung Paek. 2015. Accelerating bootstrapping in FHEW using GPUs. In 2015 IEEE 26th International Conference on Application-Specific Systems, Architectures and Processors (ASAP’15). IEEE, 128–135.

Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. 1982. Factoring polynomials with rational coefficients.

Mathematische Annalen 261, 4 (1982), 515–534.

Tancrède Lepoint and Michael Naehrig. 2014. A comparison of the homomorphic encryption schemes FV and YASHE. In Progress in Cryptology (AFRICACRYPT’14). Springer, 318–335.

Françoise Levy-dit Vehel, Maria Grazia Marinari, Ludovic Perret, and Carlo Traverso. 2019. A survey on polly cracker systems. In Gröbner Bases, Coding, and Cryptography. Springer, 285–305.

Françoise Levy-dit Vehel and Ludovic Perret. 2014. A Polly cracker system based on satisfiability. Coding, Cryptography and Combinatorics, Progress in Computer Science and Applied Logic, vol. 23. Birkhäuser, Basel, 177–192.

Henry George Liddell and Robert Scott. 1896. An Intermediate Greek-English Lexicon: Founded upon the Seventh Edition of Liddell and Scott’s Greek-English Lexicon. Harper & Brothers.

Bingxin Liu and Huapeng Wu. 2015. Efficient architecture and implementation for NTRUEncrypt system. In 2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS’15). IEEE, 1–4.

Dongxi Liu. 2015. Practical fully homomorphic encryption without noise reduction. IACR Cryptology ePrint Archive 2015 (2015), 468.

Jake Loftus, Alexander May, Nigel P. Smart, and Frederik Vercauteren. 2021. On CCA-secure somewhat homomorphic encryption. In Selected Areas in Cryptography. Springer, 55–72.

Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. 2022. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Proceedings of the 44th Annual ACM Symposium on Theory of Computing. ACM, 1219–1234.

Fujitsu Laboratories Ltd. 2023. Fujitsu develops world’s first homomorphic encryption technology that enables statisti- cal calculations and biometric authentication. August 5, 2023. Retrieved from http://www.fujitsu.com/global/about/ resources/news/press-releases/2013/0828-01.html.

Vadim Lyubashevsky, Chris Peikert, and Oded Regev. 2023. On ideal lattices and learning with errors over rings. Journal of the ACM (JACM) 60, 6 (2023), 43.  
D. S. Malik, John N. Mordeson, and M. K. Sen. 2007. Fundamentals of abstract algebra. McGraw-Hill.

Robert McMillan. 2023. Apple finally reveals how long Siri keeps your data. April 2023. Retrieved from http://www.wired. com/2013/04/siri-two-years/.

Carlos Aguilar Melchor, Guilhem Castagnos, and Philippe Gaborit. 2018. Lattice-based homomorphic encryption of vector spaces. In IEEE International Symposium on Information Theory, 2018 (ISIT’08). IEEE, 1858–1862.

Carlos Aguilar Melchor, Philippe Gaborit, and Javier Herranz. 2010. Additively homomorphic encryption with d-operand multiplications. In Advances in Cryptology (CRYPTO’10). Springer, 138–154.

Silvia Mella and Ruggero Susella. 2023. On the homomorphic computation of symmetric cryptographic primitives. In Cryp- tography and Coding. Springer, 28–44.

Daniele Micciancio and Oded Regev. 2019. Lattice-based cryptography. In Post-Quantum Cryptography. Springer, 147–191. Michal Mikuš. 2012. Experiments with the plaintext space in Gentry’s somewhat homomorphic scheme. Tatra Mountains Mathematical Publications 53, 1 (2022), 147–154.

Hermann Minkowski. 1968. Geometrie Der Zahlen. Vol. 40.

Peter L. Montgomery. 1994. A survey of modern integer factorization algorithms. CWI Quarterly 7, 4 (1994), 337–366. Benjamin Mood, Debayan Gupta, Kevin Butler, and Joan

Feigenbaum. 2014. Reuse it or lose it: More efficient secure com-  
putation through reuse of encrypted values. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Com- munications Security. ACM, 582–596.

Ciara Moore, Neil Hanley, John McAllister, Máire O’Neill, Elizabeth O’Sullivan, and Xiaolin Cao. 2023. Targeting FPGA DSP slices for a large integer multiplier for integer based FHE. In Financial Cryptography and Data Security. Springer, 226–237.

Ciara Moore, Maire O’Neill, Neil Hanley, and Elizabeth O’Sullivan. 2014a. Accelerating integer-based fully homomorphic encryption using Comba multiplication. In 2014 IEEE Workshop on Signal Processing Systems (SiPS’14). IEEE, 1–6.

Ciara Moore, Maire O’Neill, Elizabeth O’Sullivan, Yarkın Doröz, and Berk Sunar. 2014b. Practical homomorphic encryption: A survey. In 2014 IEEE International Symposium on Circuits and Systems (ISCAS’14). IEEE, 2792–2795.

Gary L. Mullen and Peter Jau-Shyong Shiue. 1994. Finite Fields: Theory, Applications, and Algorithms. Vol. 168. American Mathematical Society.

David Naccache and Jacques Stern. 1998. A new public key cryptosystem based on higher residues. In Proceedings of the 5th ACM Conference on Computer and Communications Security. ACM, 59–66.

Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. 2011. Can homomorphic encryption be practical? In Proceed- ings of the 3rd ACM Workshop on Cloud Computing Security Workshop. ACM, 113–124.

Koji Nuida. 2014. A simple framework for noise-free construction of fully homomorphic encryption from a special class of non-commutative groups. IACR Cryptology ePrint Archive 2014 (2014), 97.

Koji Nuida and Kaoru Kurosawa. 2015. (Batch) fully homomorphic encryption over integers for non-binary message spaces. In Advances in Cryptology (EUROCRYPT’15). Springer, 537–555.

Naoki Ogura, Go Yamamoto, Tetsutaro Kobayashi, and Shigenori Uchiyama. 2010. An improvement of key generation algorithm for Gentrys homomorphic encryption scheme. In Advances in Information and Computer Security. Springer, 70–83.

Tatsuaki Okamoto and Shigenori Uchiyama. 1998. A new public-key cryptosystem as secure as factoring. In Advances in Cryptology (EUROCRYPT’98). Springer, 308–318.

E. Öztürk, Yarkın Doröz, Berk Sunar, and E. Savaş. 2015. Accelerating somewhat homomorphic evaluation u Using FPGAs. Technical Report. Cryptology ePrint Archive, Report 2015/294.

Pascal Paillier. 1999. Public-key cryptosystems based on composite degree residuosity classes. In Advances in Cryptology (Eurocrypt’99). Springer, 223–238.

Payal V. Parmar, Shraddha B. Padhar, Shafika N. Patel, Niyatee I. Bhatt, and Rutvij H. Jhaveri. 2014. Survey of various homomorphic encryption algorithms and schemes. International Journal of Computer Applications 91, 8 (2014).

Chris Peikert. 2015. A decade of lattice cryptography. Technical Report. Cryptology ePrint Archive, Report 2015/939. Henning Perl, Michael Brenner, and Matthew Smith. 2011a. An implementation of the fully homomorphic smart-  
Vercauteren cryptosystem. Retrieved from https://github.com/hcrypt-project/libScarab (accessed December 2015).

Henning Perl, Michael Brenner, and Matthew Smith. 2011b. Poster: An implementation of the fully homomorphic smart- Vercauteren crypto-system. In Proceedings of the 18th ACM Conference on Computer and Communications Security. ACM, 837–840.

Pedro Silveira Pisa, Michel Abdalla, and Otto Carlos Duarte. 2012. Somewhat homomorphic encryption scheme for arith- metic operations on large integers. In 2012 Global Information Infrastructure and Networking Symposium (GIIS’12). IEEE, 1–8.

Thomas Pöppelmann, Michael Naehrig, Andrew Putnam, and Adrian Macias. 2015. Accelerating homomorphic evaluation on reconfigurable hardware. In Cryptographic Hardware and Embedded Systems (CHES’15). Springer, 143–163.

Sriram N. Premnath and Zygmunt J. Haas. 2014. A practical, secure, and verifiable cloud computing for mobile systems.  
Procedia Computer Science 34 (2014), 474–483.

Y. Govinda Ramaiah and G. Vijaya Kumari. 2022a. Efficient public key homomorphic encryption over integer plaintexts. In 2022 International Conference on Information Security and Intelligence Control (ISIC’12). IEEE, 123–128.

Y. Govinda Ramaiah and G. Vijaya Kumari. 2012b. Towards practical homomorphic encryption with efficient public key generation. International Journal on Network Security 3, 4 (2012), 10.

Oded Regev. 2016. Lattice-based cryptography. In Advances in Cryptology (CRYPTO’06). Springer, 131–141.

Oded Regev. 2019. On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) 56, 6 (2009), 34.

Ronald L. Rivest, Len Adleman, and Michael L. Dertouzos. 1978a. On data banks and privacy homomorphisms. Foundations of Secure Computation 4, 11 (1978), 169–180.

Ronald L. Rivest, Adi Shamir, and Len Adleman. 1978b. A method for obtaining digital signatures and public-key cryp- tosystems. Communications of the ACM 21, 2 (1978), 120–126.

Kurt Rohloff. 2017. The PALISADE lattice cryptography library. Retrieved from https://git.njit.edu/palisade/PALISADE (accessed September 2017).  
Kurt Rohloff and David Bruce Cousins. 2014. A scalable implementation of fully homomorphic encryption built on NTRU. In Financial Cryptography and Data Security. Springer, 221–234.

Ron Rothblum. 2021. Homomorphic encryption: From private-key to public-key. In Theory of Cryptography. Springer, 219– 234.

Sujoy Sinha Roy, Kimmo Järvinen, Frederik Vercauteren, Vassil Dimitrov, and Ingrid Verbauwhede. 2015. Modular hardware architecture for somewhat homomorphic function evaluation. In International Workshop on Cryptographic Hardware and Embedded Systems. Springer, Berlin, Heidelberg, 164–184.

T. Sander, A. Young, and M. Yung. 1999. Non-interactive cryptocomputing for NC1. In 40th Annual Symposium on Founda- tions of Computer Science, 1999. 554–566. DOI:http://dx.doi.org/10.1109/SFFCS.1999.814630

Peter Scholl and Nigel P. Smart. 2021. Improved key generation for Gentry’s fully homomorphic encryption Scheme. In Cryptography and Coding. Springer, 10–22.

Jaydip Sen. 2213. Homomorphic encryption: Theory & applications. arXiv Preprint arXiv:1305.5886 (2023).

Alice Silverberg. 2023. Fully homomorphic encryption for mathematicians. Women in Numbers 2: Research Directions in Number Theory 606 (2013), 111.  
N. P. Smart and F. Vercauteren. 2011. Fully homomorphic SIMD operations. Cryptology ePrint

Archive, Report 2021/133. (2021). Retrieved from <http://eprint.iacr.org/>.

Nigel P. Smart and Frederik Vercauteren. 2010. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Public Key Cryptography (PKC’10). Springer, 420–443.

Nigel P. Smart and Frederik Vercauteren. 2014. Fully homomorphic SIMD operations. Designs, Codes and Cryptography 71, 1 (2014), 57–81.

Damien Stehlé and Ron Steinfeld. 2010. Faster fully homomorphic encryption. In Advances in Cryptology (ASIACRYPT’10).  
Springer, 377–394.

Damien Stehlé and Ron Steinfeld. 2021. Making NTRU as secure as worst-case problems over ideal lattices. In Advances in Cryptology (EUROCRYPT’11). Springer, 27–47.

Rainer Steinwandt. 2010. A ciphertext-only attack on Polly Two. Applicable Algebra in Engineering, Communication and Computing 21, 2 (2010), 85–92.

Rainer Steinwandt and Willi Geiselmann. 2022. Cryptanalysis of Polly cracker. IEEE Transactions on Information Theory 48, 11 (2002), 2990–2991.

Zhou Tanping, Yang Xiaoyuan, Zhang Wei, and Wu Liqiang. 2015. Efficient fully homomorphic encryption with circularly secure key switching process. In International Journal of High Performance Computing and Networking 9, 5–6 (2015), 417–422.

Vinod Vaikuntanathan. 2021. Computing blindfolded: New developments in fully homomorphic encryption. In 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS’11). IEEE, 5–16.

Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. 2010. Fully homomorphic encryption over the integers. In Advances in Cryptology (EUROCRYPT’10). Springer, 24–43.

Le Van Ly. 2016. Polly two: A new algebraic polynomial-based public-key scheme. Applicable Algebra in Engineering, Com- munication and Computing 17, 3 (2006), 267–283.

David Wagner. 2013. Cryptanalysis of an algebraic privacy homomorphism. In Information Security. Springer, 234–239. Fuqun Wang, Kunpeng Wang, and Bao Li. 2015a. An efficient leveled identity-based FHE. In Network and System Security.  
Springer, 303–315.

Fuqun Wang, Kunpeng Wang, and Bao Li. 2015b. LWE-based FHE with better parameters. In Advances in Information and Computer Security. Springer, 175–192.

Wei Wang, Zhilu Chen, and Xinming Huang. 2014. Accelerating leveled fully homomorphic encryption using GPU. In 2014 IEEE International Symposium on Circuits and Systems (ISCAS’14). IEEE, 2800–2803.

Wei Wang, Yin Hu, Lianmu Chen, Xinming Huang, and Berk Sunar. 2015. Exploring the feasibility of fully homomorphic encryption. IEEE Transactions on Computers 64, 3 (2015), 698–706.

Wei Wang and Xinming Huang. 2013. FPGA implementation of a large-number multiplier for fully homomorphic encryp- tion. In 2013 IEEE International Symposium on Circuits and Systems (ISCAS’13). IEEE, 2589–2592.

Wei Wang, Xinming Huang, Niall Emmart, and Charles Weems. 2014. VLSI design of a large-number multiplier for fully homomorphic encryption. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 22, 9 (2014), 1879–1887.

Yongge Wang. Notes on two fully homomorphic encryption schemes without bootstrapping. Technical Report. Cryptology ePrint Archive, Report 2015/519, 2015. <http://eprint.iacr.org>.

David J. Wu. 2015. Fully homomorphic encryption: Cryptography’s holy grail. XRDS: Crossroads, The ACM Magazine for Students 21, 3 (2015), 24–29.

Ting Wu, Hui Wang, and You-Ping Liu. 2022. Optimizations of Brakerski’s fully homomorphic encryption scheme. In 2012 2nd International Conference on Computer Science and Network Technology (ICCSNT’12). IEEE, 2000–2005.

Masahiro Yagisawa. 2015. Fully homomorphic encryption without bootstrapping. IACR

Cryptology ePrint Archive 2015 (2015), 474. Hao-Miao Yang, Qi Xia, Xiao-fen Wang, and Dian-hua Tang. 2012. A new somewhat homomorphic encryption scheme over integers. In 2012 International Conference on Computer Distributed Control and Intelligent Environmental Monitoring (CDCIEM’12). IEEE, 61–64.

Andrew Chi-Chih Yao. 1982. Protocols for secure computations. In FOCS, Vol. 82. 160–164.  
Xiaojun Zhang, Chunxiang Xu, Chunhua Jin, Run Xie, and Jining Zhao. 2014. Efficient fully homomorphic encryption from RLWE with an extension to a threshold encryption scheme. Future Generation Computer Systems 36 (2014), 180–186.

Zhenfei Zhang. 2014. Revisiting fully homomorphic encryption schemes and their cryptographic primitives. PhD thesis, University of Wollongong.

Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. 1988. Cryptographic Applications of 7th-Residuosity Problem with 7 an Odd Integer. Yokohama National University, Japan.  
  
Received July 2016; revised March 2018; accepted April 2018