ISSN No:-2456-2165

HECAT: Homomorphic Encryption Classification and Taxonomy

Koffka Khan
Department of Computing and Information Technology,
The University of the West Indies, St Augustine, Trinidad and Tobago, W.I.

Abstract:- Homomorphic encryption is a vital cryptographic technique that enables computations on encrypted data while preserving privacy and security. Understanding the diverse landscape of homomorphic encryption schemes is crucial for both researchers and practitioners. To facilitate this understanding, we introduce HECAT, the Homomorphic Encryption Classification and Taxonomy. HECAT is a comprehensive framework designed to classify and categorize homomorphic encryption schemes based on various attributes, including mathematical operations, security levels, use cases, and practical implementations. This taxonomy serves as a valuable resource for anyone seeking to navigate the complex field of homomorphic encryption and make informed decisions about its application in diverse scenarios. By providing a structured overview of the field, HECAT aims to promote the adoption and development of homomorphic encryption for secure and privacy-preserving data processing.

Keywords:- *Homomorphic, Encryption, Classification, Taxonomy.*

I. INTRODUCTION

In the digital age, the need for secure and privacy-preserving data processing has become paramount. Organizations and individuals alike seek innovative solutions to perform computations on sensitive information while keeping it encrypted. Homomorphic encryption [1], [3], [21], a groundbreaking cryptographic technique, has emerged as a powerful tool in this regard. It allows operations to be executed on encrypted data without the need for decryption, effectively bridging the gap between data security and computation. However, the field of homomorphic encryption is vast and complex, comprising a myriad of schemes, each with unique properties and applications. To aid in navigating this intricate landscape, we present "HECAT" – the Homomorphic Encryption Classification and Taxonomy.

HECAT serves as a comprehensive framework for categorizing and classifying homomorphic encryption schemes based on a multitude of attributes. This taxonomy addresses the need for a structured overview of the field, offering a way to discern between various schemes and their suitability for different use cases. Homomorphic encryption can be divided into Partially Homomorphic Encryption (PHE) [32] and Fully Homomorphic Encryption (FHE) [2] based on the types of mathematical operations they support. Understanding the security level of a scheme is paramount, which leads to classifications such as Somewhat Homomorphic Encryption (SHE) [25], Levelled Fully Homomorphic Encryption (L-FHE) [23], and Bootstrappable Fully Homomorphic Encryption (B-FHE) [22]. Moreover, schemes can be characterized by their mathematical foundations, including lattice-based, number-theoretic, and elliptic curve-based homomorphic encryption.

Practical implementations also play a significant role in the taxonomy. As the demand for homomorphic encryption in various domains has grown, open-source libraries and tools have emerged, providing accessible and efficient means to implement these cryptographic techniques. HECAT encompasses all these aspects and more, creating a structured reference framework for researchers, developers, and decision-makers in the realm of secure data processing. In this introduction, we provide an overview of the importance and relevance of homomorphic encryption, setting the stage for the exploration and understanding of this field through the lens of HECAT.

This paper consists of seven sections. The importance of video streaming in modern life is discussed in Section II. The VSST taxonomy is given in Section III with the relationships between taxonomy elements in Section IV. A discussion is given in Section V and taxonomy uses in Section VI. Finally, the conclusion is given in Section VII.

II. HOMOMORPHIC ENCRYPTION

Homomorphic encryption stands as a groundbreaking advancement in the field of cryptography, offering a revolutionary solution to the ever-pressing challenge of balancing data privacy and utility. In essence, it allows computations to be performed on encrypted data without the need to decrypt it, thereby preserving the confidentiality of sensitive information while enabling practical operations. This is a profound departure from traditional encryption, where data must be decrypted before any meaningful computation can take place, inevitably exposing it to potential security risks. Homomorphic encryption, often described as the "holy grail" of cryptography, holds immense promise for a wide range of applications, from secure cloud computing and confidential data analytics to privacy-preserving machine learning and secure data sharing.

The concept of homomorphic encryption was first introduced in the late 1970s, but it remained largely theoretical for several decades due to its complexity and performance limitations. Over time, advances in cryptographic research and computational power have made it increasingly practical. This cryptographic marvel is fundamentally based on the idea that encrypted data, when operated on through a specific set of mathematical operations, can yield results that, when decrypted, are equivalent to the results obtained by performing the same operations on the plaintext data. In other words, homomorphic encryption allows mathematical functions like addition and multiplication to be applied directly to encrypted data, with the outcome being meaningful and accurate results once the data is decrypted.

The applications of homomorphic encryption are vast and continue to expand. It addresses a range of critical data security challenges, from the protection of personal privacy in healthcare and financial sectors to ensuring secure data processing in the cloud, and even enabling confidential machine learning over sensitive datasets. The potential to perform computations on data without ever revealing its underlying contents offers a unique solution to the privacy-versus-utility dilemma in an increasingly data-driven world. However, implementing homomorphic encryption is not without its challenges, as it can be computationally intensive and requires careful management of cryptographic keys. As the field continues to evolve, research and development efforts are focused on making homomorphic encryption more efficient and accessible for a broader range of applications.

Video streaming [12], [11], [13], [14], [15], as a specific application of homomorphic encryption, revolutionizes the way we experience and protect multimedia content. By employing homomorphic encryption techniques, video streaming platforms can secure the entire process of content delivery, from the source to the end-user, while preserving user privacy. This means that video data can remain fully encrypted, even during transmission and processing, providing robust protection against unauthorized access or data breaches. Viewers can watch videos without the need to decrypt the content locally, ensuring that sensitive personal information remains confidential and immune to privacy infringements, all while enjoying a seamless and secure streaming experience. This application of homomorphic encryption is particularly relevant in an era where privacy concerns and data security are paramount in the digital world.

III. HECAT TAXONOMY

Homomorphic encryption is a cryptographic technique that allows computations to be performed on encrypted data without decrypting it. It can be categorized into different types and schemes based on various attributes. Here's a taxonomy for homomorphic encryption:

A. Based on Mathematical Operations:

- Partially Homomorphic Encryption (PHE): Partially Homomorphic Encryption (PHE) refers to a class of cryptographic schemes that support only one type of mathematical operation (either addition or multiplication) on encrypted data, while not supporting both operations simultaneously. PHE has been a crucial stepping stone in the development of more advanced homomorphic encryption schemes like Fully Homomorphic Encryption (FHE).

- Additive Homomorphism (PHE-Additive): PHE schemes that are limited to supporting addition operations on encrypted data are known as PHE-Additive. These schemes allow encrypted numbers to be added together, and the result remains encrypted. For example, if you have two ciphertexts representing numbers a and b, you can perform an operation on them to get a ciphertext representing a + b.

- Multiplicative Homomorphism (PHE-Multiplicative): PHE-Multiplicative schemes, on the other hand, support only multiplication operations on encrypted data. This means that you can multiply two ciphertexts, resulting in a ciphertext representing the product of the original numbers.

- The limitation of PHE is that it does not support both addition and multiplication in a single scheme. Consequently, you cannot perform arbitrary computations on encrypted data without using a combination of different schemes and additional protocols. While PHE is not as versatile as Fully Homomorphic Encryption, it remains valuable for specific use cases where only one type of operation is required, and it may offer performance advantages over more complex schemes.

- Fully Homomorphic Encryption (FHE): Fully Homomorphic Encryption (FHE) represents the most advanced and powerful category of homomorphic encryption schemes. Unlike PHE, FHE enables both addition and multiplication operations to be performed on encrypted data, and it goes further by allowing arbitrary computations to be executed on ciphertexts. This remarkable capability makes FHE a groundbreaking technology with a wide range of applications in secure data processing.

- Arbitrary Computations: In FHE, it is possible to perform arbitrary operations on encrypted data, including addition, multiplication, comparison, and logical operations like AND, OR, and NOT. This means that you can execute complex algorithms and computations while keeping the data in encrypted form throughout the entire process.

- Use Cases: Fully Homomorphic Encryption is particularly valuable in scenarios where data privacy and security are paramount. It has applications in secure cloud computing, privacy-preserving data analytics, and confidential machine learning, among others. With FHE, sensitive data can be outsourced to untrusted servers for processing without revealing the actual content of the data.

- Security and Complexity: FHE is conceptually more complex and computationally intensive than PHE, which makes it challenging to implement efficiently. However, ongoing research and development are addressing these challenges, and practical FHE libraries are becoming available, making it increasingly viable for real-world applications.

In summary, Fully Homomorphic Encryption (FHE) is a remarkable advancement that allows for both addition and multiplication operations on encrypted data, enabling arbitrary computations on sensitive information while maintaining data privacy and security. Partially Homomorphic Encryption (PHE), in contrast, supports only one type of operation (either addition or multiplication) on encrypted data, making it less versatile but still valuable for specific use cases.

B. Based on Security Level:

- Somewhat Homomorphic Encryption (SHE):Somewhat Homomorphic Encryption (SHE) represents a class of homomorphic encryption schemes that offer a limited level of security. While they enable some degree of computation on encrypted data, they are typically vulnerable to data leakage after a certain number of homomorphic operations. SHE schemes are often used in situations where a few specific computations need to be performed on encrypted data, and the focus is on simplicity rather than providing strong security guarantees.

- Limited Operations: SHE schemes support either addition or multiplication operations on encrypted data, but not both. This limitation restricts the types of computations that can be performed securely.

- Security Trade-Off: The trade-off with SHE is that while it allows for some degree of computation, it may become insecure if too many operations are performed. If the encryption process is "leaky," it could potentially reveal information about the plaintext data.

- Use Cases: SHE is useful in scenarios where data security is important, but the need for complex computations is limited. It may be used in basic data aggregation or simple calculations where a high level of security is not required.

- Levelled Fully Homomorphic Encryption (L-FHE):

Levelled Fully Homomorphic Encryption (L-FHE) schemes build upon the idea of Fully Homomorphic Encryption (FHE) and offer a more secure version of FHE with certain depth limitations. In L-FHE, there are restrictions on the number of homomorphic operations that can be performed in a sequence, which helps to mitigate security concerns related to FHE.

- Depth Limitations: L-FHE schemes place limitations on the depth of computations that can be performed in a single sequence. This means that only a certain number of operations can be executed before additional security measures are required.

- Enhanced Security: L-FHE provides a higher level of security compared to FHE, as it reduces the potential for data leakage due to overuse of homomorphic operations. This is particularly important for sensitive applications.

- Balancing Security and Computation: L-FHE aims to strike a balance between security and computation depth. It allows for more complex computations than SHE while maintaining a higher level of security than traditional FHE.

- Bootstrappable Fully Homomorphic Encryption (B-FHE):

Bootstrappable Fully Homomorphic Encryption (B-FHE) is a category of homomorphic encryption schemes that addresses the security challenges of performing deep and complex computations. B-FHE schemes enable secure data processing by introducing a process called "bootstrapping," which refreshes the ciphertext to prevent data leakage.

- Bootstrapping Process: B-FHE schemes include a bootstrapping procedure that allows ciphertexts to be refreshed. This process effectively "re-encrypts" the ciphertext, ensuring that data remains secure, even after a substantial number of homomorphic operations.

- Deeper Computations: With bootstrapping, B-FHE schemes can support much deeper and more complex computations compared to traditional FHE. This makes B-FHE suitable for a broader range of applications.

- Enhanced Security: The bootstrapping process significantly enhances security by preventing data leakage due to excessive operations. This enables B-FHE to provide robust privacy guarantees in scenarios where deep and complex computations are necessary.

In summary, the security level in homomorphic encryption is crucial for ensuring the confidentiality of data during computations. Somewhat Homomorphic Encryption (SHE) offers limited security but is straightforward to use. Levelled Fully Homomorphic Encryption (L-FHE) provides enhanced security compared to traditional FHE but with depth limitations. Bootstrappable Fully Homomorphic Encryption (B-FHE) addresses these limitations by introducing a bootstrapping process, enabling secure deep and complex computations while maintaining data privacy.

C. Based on Operations Supported:

- Additive Homomorphic Encryption:

Additive Homomorphic Encryption [5], [7] refers to a class of cryptographic schemes that support only addition operations on encrypted data. This means that you can perform operations such as addition or subtraction on ciphertexts without decrypting the data.

- Addition Operations: Additive Homomorphic Encryption allows you to compute the sum or difference of two ciphertexts. For example, if you have encrypted values A and B, you can perform operations to get a ciphertext representing A + B or A - B.

- Limited Computational Capability: The limitation of Additive Homomorphic Encryption is that it only supports addition operations. This means you can't perform multiplication, division, or more complex operations on the encrypted data.
- Use Cases: Additive Homomorphic Encryption is suitable for applications where secure aggregation or basic mathematical calculations are required. It can be applied in scenarios such as privacy-preserving data aggregation or secure statistics computations.

- Multiplicative Homomorphic Encryption:

Multiplicative Homomorphic Encryption [16] is a category of cryptographic schemes that support only multiplication operations on encrypted data. This means you can perform operations like multiplication or division on ciphertexts without revealing the plaintext values.

- Multiplication Operations: Multiplicative Homomorphic Encryption allows you to compute the product or division of two ciphertexts. If you have encrypted values X and Y, you can perform operations to obtain a ciphertext representing X * Y or X / Y.

- Limited Computational Capability: Similar to Additive Homomorphic Encryption, Multiplicative Homomorphic Encryption has limited computational capabilities. It supports multiplication and division but not addition or more complex operations.

- Use Cases: Multiplicative Homomorphic Encryption is valuable in situations where secure multiplicative computations are needed. It can be applied in fields like secure data sharing in healthcare, where computations like multiplying medical data by a constant factor must be performed privately.

- Fully Homomorphic Encryption (FHE):

Fully Homomorphic Encryption (FHE) represents the most advanced category of homomorphic encryption schemes. Unlike Additive and Multiplicative Homomorphic Encryption, FHE supports both addition and multiplication operations on encrypted data. Moreover, FHE allows for arbitrary computations on ciphertexts.

- Arbitrary Computations: FHE enables arbitrary operations on encrypted data, including addition, multiplication, comparison, logical operations (AND, OR, NOT), and much more. This allows for complex algorithms and computations to be executed on encrypted data.

- Versatility: FHE is highly versatile and can be applied in a wide range of scenarios where both addition and multiplication operations are required. It is particularly valuable in privacy-preserving data analytics, secure cloud computing, confidential machine learning, and more.

- Security and Complexity: Implementing FHE can be challenging due to its computational intensity and complexity. However, ongoing research and development are improving its efficiency, making it more practical for real-world applications.

In summary, the classification based on operations supported in homomorphic encryption provides different levels of computational capability. Additive Homomorphic Encryption supports only addition, Multiplicative Homomorphic Encryption supports only multiplication, and Fully Homomorphic Encryption (FHE) supports both addition and multiplication, along with a wide range of other computations, making it the most versatile but also the most complex category. The choice of scheme depends on the specific use case and the operations needed to be performed securely on the encrypted data.

D. Based on Mathematical Framework:

- Lattice-based Homomorphic Encryption:

Lattice-based Homomorphic Encryption [27], [20] schemes are a class of cryptographic systems that are built upon the mathematical foundation of lattice problems. Lattice problems involve the study of lattices, which are geometric structures formed by regular grids of points in n-dimensional space. The security of these schemes relies on the difficulty of certain computational problems related to lattices.

- Post-Quantum Secure: Lattice-based Homomorphic Encryption is known for its strong security properties and is considered "post-quantum secure." This means that these schemes are believed to remain secure even in the face of attacks using quantum computers, which have the potential to break many classical encryption methods.

- Versatile Security Levels: Lattice-based schemes come in various flavors, providing different security levels based on the complexity of the lattice problem. They offer a trade-off between computational efficiency and security.

- Use Cases: These schemes find applications in secure data transmission, secure multiparty computation, and privacy-preserving machine learning. They are particularly valuable in environments where long-term security against quantum attacks is a concern.

- Number-Theoretic Homomorphic Encryption:

Number-Theoretic Homomorphic Encryption [24] schemes are rooted in number theory, a branch of mathematics that deals with properties and relationships of integers. One of the most well-known examples of this category is the RSA-based Paillier cryptosystem.

- Number Theory Foundations: These schemes rely on number-theoretic properties, such as the difficulty of factoring large composite numbers or solving the discrete logarithm problem. The security of these schemes is closely tied to the mathematical properties of certain number-theoretic functions.
- Homomorphism through Number Theory: The homomorphic properties of these schemes are achieved by carefully designing mathematical operations in a way that

allows the ciphertexts to be manipulated to perform addition and multiplication operations without decryption.

- Use Cases: Number-Theoretic Homomorphic Encryption is widely used in applications where security and performance are both important. It is commonly applied in secure voting systems, privacy-preserving financial transactions, and data sharing with trust constraints.

- Elliptic Curve Homomorphic Encryption:

Elliptic Curve Homomorphic Encryption [6] schemes are based on elliptic curve cryptography (ECC). ECC is a branch of mathematics and cryptography that involves using elliptic curves to provide security and efficiency in encryption.

- Elliptic Curve Cryptography: These schemes use the mathematical properties of elliptic curves to create encryption systems with strong security and computational efficiency. ECC is known for its smaller key sizes compared to traditional cryptosystems like RSA.

- Homomorphic Properties: Elliptic Curve Homomorphic Encryption schemes are designed to maintain homomorphic properties while operating on encrypted data. This means they can support addition and multiplication operations on ciphertexts.

- Use Cases: ECC-based homomorphic encryption is particularly valuable in resource-constrained environments, such as IoT devices, where efficient encryption and secure computation are essential. It is also used in scenarios where key size and computational speed are crucial, such as secure messaging applications.

In summary, the classification based on the mathematical framework in homomorphic encryption reflects the underlying mathematical structures that provide security and computational efficiency. Lattice-based Homomorphic Encryption is known for its post-quantum security, Number-Theoretic Homomorphic Encryption relies on number theory for security, and Elliptic Curve Homomorphic Encryption leverages the properties of elliptic curves for efficient and secure encryption. The choice of scheme depends on the specific use case, security requirements, and computational constraints.

E. Based on Use Cases:

- Text-based Homomorphic Encryption:

Text-based Homomorphic Encryption [9] is a category of homomorphic encryption schemes specifically designed for processing and searching encrypted text data. It addresses the need to perform operations on sensitive textual information without revealing the content of the data.

- Textual Data Processing: These schemes are tailored for operations on encrypted text, including searching, sorting, and basic computations, while keeping the data confidential.

- Use Cases: Text-based Homomorphic Encryption is relevant in scenarios like secure cloud-based document searching, encrypted email processing, and confidential text data analytics. It is widely used in situations where privacy-preserving text data processing is essential.

- Privacy-Preserving Search: It enables secure searches on encrypted documents or databases, allowing users to retrieve relevant information without exposing the content of the documents.

- Image-based Homomorphic Encryption:

Image-based Homomorphic Encryption [28], [18] focuses on processing and analyzing encrypted image data while maintaining data privacy and security.

- Image Data Processing: These schemes are optimized for operations on encrypted image data, such as image filtering, feature extraction, or encrypted image classification.

- Use Cases: Image-based Homomorphic Encryption has applications in secure image processing, medical image analysis, and privacy-preserving image recognition. It's valuable in situations where sensitive visual data needs to be processed without decryption.

- Secure Medical Imaging: In the healthcare sector, it's used for secure processing and sharing of medical images, allowing for diagnosis while preserving patient privacy.

- Database Homomorphic Encryption:

Database Homomorphic Encryption [30] is tailored to enable secure computations on encrypted databases, ensuring data privacy while still permitting various operations on the data.

- Database Operations: These schemes are designed to support database query operations on encrypted data, including search, aggregation, and analysis.

- Use Cases: Database Homomorphic Encryption is crucial for secure data outsourcing and sharing, as it allows authorized parties to interact with the data without exposing the raw data itself. It finds applications in secure cloud-based databases, multi-party computation, and privacy-preserving analytics.

- Multi-party Computation: It facilitates secure collaboration and computation over data owned by multiple parties, making it suitable for scenarios involving multiple data providers.

- Machine Learning Homomorphic Encryption:

Machine Learning Homomorphic Encryption [31], [8], [17] schemes are designed to perform secure computations on encrypted machine learning models and data, facilitating privacy-preserving machine learning and model deployment.

- Machine Learning Operations: These schemes enable operations on encrypted machine learning models, including model evaluation, predictions, and training, all while preserving the confidentiality of the model and data.

- Use Cases: Machine Learning Homomorphic Encryption is valuable in applications where privacy and confidentiality are critical, such as secure predictive maintenance, healthcare diagnostics, and confidential machine learning in financial and industrial sectors.

- Model Deployment: It allows machine learning models to be securely deployed in untrusted environments, enabling predictive services on encrypted data without revealing the underlying model or sensitive training data.

In summary, the classification based on use cases in homomorphic encryption reflects the diverse range of applications and scenarios where secure data processing is paramount. Text-based, Image-based, Database, and Machine Learning Homomorphic Encryption cater to specific data types and processing requirements while upholding data privacy and security. The choice of scheme depends on the specific use case, the type of data involved, and the operations required for secure computation.

F. Based on Key Management:

- Fully Key-Homomorphic Encryption:

Fully Key-Homomorphic Encryption [29] refers to a category of homomorphic encryption schemes that allow operations to be performed with different keys for data and operations. This approach provides enhanced security, control, and flexibility in various use cases.

- Separate Keys for Data and Operations: In Fully Key-Homomorphic Encryption, there are distinct keys for encrypting data and for performing operations on the data. These keys are managed separately, offering more fine-grained control.

- Enhanced Security and Access Control: The separation of keys provides enhanced security and access control. It enables data owners to restrict who can perform computations on their data without having access to the data itself.

- Use Cases: Fully Key-Homomorphic Encryption is valuable in scenarios where data owners want to maintain tight control over who can process their data. It's applicable in secure multi-party computation, confidential data sharing, and situations where data providers and data processors are separate entities.

- Complex Key Management: Managing multiple keys can be more complex than using a single key, but it provides a higher level of security and control over the data.

- Single Key Homomorphic Encryption:

Single Key Homomorphic Encryption [26] is a category of homomorphic encryption schemes that use a single key for both data encryption and operations, simplifying the key management process. However, it may have some limitations in certain scenarios.

- Unified Key for Data and Operations: In Single Key Homomorphic Encryption, a single key is used for both encrypting the data and performing operations on the encrypted data. This simplifies key management but may have trade-offs in terms of security and control.

- Simplicity and Efficiency: The use of a single key simplifies the encryption and decryption processes, making the scheme easier to implement and use. It may also offer computational efficiency advantages.

- Use Cases: Single Key Homomorphic Encryption is suitable for scenarios where the primary goal is ease of use and where the data owner is also the entity performing computations. It is commonly used in applications like secure cloud-based processing, confidential data analytics, and personal data protection.

- Limitations: The main limitation of Single Key Homomorphic Encryption is that it may not provide the same level of security and access control as Fully Key-Homomorphic Encryption. Data processors with access to the key may have more control over the data, which could pose privacy concerns in certain situations.

In summary, the classification based on key management in homomorphic encryption reflects the choices made in managing cryptographic keys for data protection and secure computations. Fully Key-Homomorphic Encryption provides enhanced security and control by using separate keys for data and operations, while Single Key Homomorphic Encryption simplifies key management but may have limitations in terms of access control. The choice of scheme depends on the specific use case, the desired level of control, and the security requirements of the data owners and processors.

G. Based on Practical Implementations:

- Common Homomorphic Encryption Schemes:

Common Homomorphic Encryption Schemes include widely recognized and established encryption techniques that have been applied and adapted to support homomorphic encryption. These schemes form the foundation of practical implementations of homomorphic encryption.

- Paillier Cryptosystem: The Paillier cryptosystem [10] is one of the earliest and most well-known homomorphic encryption schemes. It is primarily used for additive homomorphic operations, allowing secure addition of encrypted data. The Paillier cryptosystem is applied in various privacy-preserving applications, such as secure voting, secure auctioning, and encrypted data aggregation.

- RSA-based Cryptosystem: The RSA cryptosystem, known for its use in public key encryption, can be adapted for homomorphic encryption. It is often used for partially homomorphic encryption, especially in scenarios where data security and encryption interoperability are essential.

- BGV Scheme: The BGV scheme [19] is a fully homomorphic encryption scheme that is considered to be one of the more advanced and efficient options for practical implementations. It supports both addition and multiplication operations, making it suitable for complex computations on encrypted data. The BGV scheme is applied in secure cloud computing, confidential data analytics, and secure machine learning.

- Use Cases: Common Homomorphic Encryption Schemes are employed in a wide range of applications, including privacy-preserving data processing, secure data sharing, confidential analytics, and cryptographic protocols for secure computations. They provide the cryptographic foundation for many real-world implementations.

- Open-Source Libraries and Tools:

Open-Source Libraries and Tools play a crucial role in making homomorphic encryption more accessible and practical for developers and organizations. These libraries and tools offer pre-built software solutions, APIs, and resources for implementing homomorphic encryption in real-world applications.

- Microsoft's Simple Encrypted Arithmetic Library (SEAL): SEAL [4] is an open-source library developed by Microsoft that provides a user-friendly, high-performance implementation of homomorphic encryption. It supports the BGV scheme and is designed to be easy to use for developers. SEAL is applied in various secure computing scenarios, including secure machine learning, privacy-preserving analytics, and confidential data processing.

- TenSEAL: TenSEAL [33] is another open-source library that extends Microsoft SEAL with support for tensor operations, making it particularly useful for machine learning and deep learning applications. It is designed to simplify the process of building machine learning models with homomorphic encryption.

- Other Open-Source Implementations: Besides Microsoft's offerings, there are several other open-source libraries and tools available, such as PALISADE, HElib, and PySEAL [33]. These libraries provide a wide range of options for developers and researchers to experiment with and apply homomorphic encryption in different use cases.

- Community Contributions: Open-source libraries and tools benefit from community contributions, which enhance their functionality, security, and usability. This collaborative effort promotes the adoption of homomorphic encryption by lowering barriers to entry and fostering innovation.

In summary, the classification based on practical implementations in homomorphic encryption encompasses the well-established Common Homomorphic Encryption Schemes that serve as the cryptographic basis for secure computations, as well as the user-friendly and accessible Open-Source Libraries and Tools that make it easier for developers to incorporate homomorphic encryption into their applications. These tools and libraries play a pivotal role in advancing the practical adoption of homomorphic encryption across various industries and use cases.

IV. RELATIONSHIPS AMONG TAXOMONY ELEMENTS

The taxonomy elements provided categorize homomorphic encryption schemes based on different characteristics. Let's explore how these elements are related to each other:

- Based on Mathematical Operations: This taxonomy element classifies homomorphic encryption schemes based on the types of mathematical operations they support. The categorization into Partially Homomorphic Encryption (PHE), Fully Homomorphic Encryption (FHE), and other variations is crucial because it determines the range of computations that can be performed on encrypted data. These categories directly influence the practical use cases of homomorphic encryption.

- Based on Security Level: This element categorizes homomorphic encryption schemes based on their security properties. The security level of a scheme is a critical factor in determining its suitability for various applications. More secure schemes, like Bootstrappable Fully Homomorphic Encryption (B-FHE), are preferred for scenarios where strong data privacy is paramount. Security level is closely related to the level of mathematical operations a scheme can support, as stronger security often comes with trade-offs in terms of computational complexity.

- Based on Operations Supported: The mathematical operations a scheme can perform directly impact its categorization. For instance, Additive Homomorphic Encryption and Multiplicative Homomorphic Encryption support specific operations, influencing how they can be used. Fully Homomorphic Encryption (FHE) is the most versatile category as it supports both addition and multiplication operations.

- Based on Mathematical Framework: The mathematical framework used in a homomorphic encryption scheme is a fundamental characteristic that influences its security and efficiency. Lattice-based, Number-Theoretic, and Elliptic Curve Homomorphic Encryption schemes all leverage different mathematical foundations. The choice of framework can impact the level of security and computational efficiency.

- Based on Use Cases: The practical applications of homomorphic encryption are diverse and may require specific characteristics. Text-based, Image-based, Database, and Machine Learning Homomorphic Encryption categories correspond to the types of data and computations they are designed for. The use cases of homomorphic encryption are closely tied to the capabilities of the schemes, such as whether they support text or image data and specific operations required for machine learning.

- Based on Key Management: How cryptographic keys are managed is a crucial aspect of homomorphic encryption. Fully Key-Homomorphic Encryption and Single Key Homomorphic Encryption differ in how keys are used for data encryption and operations. Key management directly impacts the control, security, and practicality of the encryption scheme.

- Based on Practical Implementations: Practical implementations of homomorphic encryption encompass both common schemes like Paillier and RSA-based encryption, and open-source libraries and tools such as Microsoft's SEAL and TenSEAL. The choice of scheme or tool is influenced by the specific requirements of a practical application. Common schemes provide the cryptographic foundation, while open-source libraries and tools make it more accessible for developers to integrate homomorphic encryption into their applications.

In summary, the taxonomy elements are interrelated and provide a structured framework for understanding the diverse landscape of homomorphic encryption. The characteristics of a scheme in each category influence and determine its practical applications, security properties, mathematical framework, key management, and level of support for different operations. Together, these elements help researchers, developers, and decision-makers navigate the field of homomorphic encryption and select the most suitable schemes for their specific needs.

V. DISCUSSION: PURPOSE OF TAXONOMY

The purpose of a taxonomy in any field, including homomorphic encryption, is to provide a structured framework for classifying and organizing complex concepts, objects, or systems based on their inherent characteristics. In the context of homomorphic encryption, a taxonomy serves several important purposes:

- Clarity and Organization: Homomorphic encryption is a highly intricate and multifaceted field. A taxonomy helps bring order to this complexity by categorizing and organizing the various encryption schemes and concepts into distinct and comprehensible groups. This makes it easier for researchers, developers, and users to navigate and understand the field.
- Comparative Analysis: By categorizing homomorphic encryption schemes based on specific attributes, a taxonomy facilitates comparative analysis. Users can easily compare and contrast different schemes within the same category or between categories. This is crucial for evaluating the strengths, weaknesses, and suitability of schemes for specific use cases.

- Use Case Identification: The taxonomy provides a clear link between the attributes of homomorphic encryption schemes and their practical applications. This helps users identify which schemes are most suitable for their particular use cases. For example, if someone is interested in secure data processing for machine learning, they can look for schemes categorized under "Machine Learning Homomorphic Encryption."

- Security and Risk Assessment: Categorizing schemes based on security levels and key management provides insights into the trade-offs between security and performance. Users can assess the risks and benefits associated with each scheme and select the one that aligns with their security requirements.

- Research and Development: Researchers in the field of homomorphic encryption can use the taxonomy to identify gaps in existing schemes and focus on areas that require further investigation. The taxonomy also aids in the development of new encryption techniques that fit specific categories or address shortcomings in current schemes.

- Practical Implementation Guidance: The taxonomy helps developers and engineers in implementing homomorphic encryption by pointing them toward specific open-source libraries and tools that align with their needs. It provides a bridge between theoretical concepts and real-world applications.

- Communication and Standardization: A taxonomy provides a common language and framework for discussing homomorphic encryption within the research and development community. This standardization improves communication and knowledge sharing.

- Future Advancements: As the field of homomorphic encryption evolves, the taxonomy can adapt and expand to accommodate new schemes, emerging use cases, and advances in cryptographic research. This ensures that the taxonomy remains a relevant and valuable resource over time.

In summary, the purpose of a taxonomy in homomorphic encryption is to simplify the understanding and application of this complex field. It aids in classifying schemes, matching them with use cases, assessing security properties, and promoting effective communication and standardization. Ultimately, a well-constructed taxonomy fosters innovation, collaboration, and the responsible adoption of homomorphic encryption in a wide range of applications, from secure data processing to privacy-preserving machine learning.

VI. TAXONOMY USE CASES

The taxonomy of homomorphic encryption can be beneficial in various real-world use cases. Here are some examples of how different stakeholders can use the taxonomy:

- *Researchers and Academics:*

Researchers can use the taxonomy to categorize and compare homomorphic encryption schemes in their studies and publications.

It helps in identifying gaps in the field and areas where further research is needed.

➢ *Developers and Engineers:*
Developers can utilize the taxonomy to choose the right homomorphic encryption scheme for their specific application or system.

It guides them in understanding the security and performance trade-offs and how to implement the chosen scheme effectively.

➢ *Data Scientists and Analysts:*
Data scientists can use the taxonomy to identify homomorphic encryption schemes suitable for their data analytics and machine learning tasks.

It helps in selecting the appropriate scheme for preserving data privacy while performing computations.

➢ *Data Owners and Privacy Advocates:*
Data owners can leverage the taxonomy to understand the security and privacy implications of different homomorphic encryption schemes.

It aids in making informed decisions regarding how to protect sensitive data while allowing it to be used for various purposes.

➢ *Businesses and Organizations:*
Organizations can use the taxonomy to assess the feasibility of adopting homomorphic encryption for securing sensitive data.

It provides a framework for understanding the potential benefits, challenges, and practical implementation options.

➢ *Government and Regulatory Bodies:*
Regulatory bodies can reference the taxonomy to understand the different categories of homomorphic encryption and their security levels.

It helps in crafting policies and regulations related to data privacy and security.

➢ *Consultants and Security Experts:*
Consultants and security experts can use the taxonomy to advise clients on implementing homomorphic encryption for specific security and privacy needs.

It assists in guiding organizations through the selection and deployment process.

➢ *Open-Source Community:*
The open-source community can contribute to the taxonomy by developing and maintaining open-source libraries and tools that align with its categories.

It encourages collaboration and the development of user-friendly homomorphic encryption solutions.

In summary, the taxonomy of homomorphic encryption serves as a valuable reference and decision-making tool for a wide range of stakeholders. Whether it's for research, development, data analysis, regulatory compliance, or consulting, the taxonomy helps individuals and organizations navigate the complex landscape of homomorphic encryption and make informed choices about how to leverage it for various use cases while balancing security, privacy, and computational requirements.

VII. CONCLUSION

In conclusion, the taxonomy of homomorphic encryption provides a structured framework for categorizing and understanding this complex field of cryptography. It offers a clear delineation of different encryption schemes based on various attributes such as mathematical operations, security level, operations supported, mathematical framework, use cases, key management, and practical implementations. This taxonomy is a vital resource for researchers, developers, data scientists, and organizations seeking to harness the power of homomorphic encryption for secure data processing, privacy preservation, and innovative applications. It aids in selecting the most suitable encryption scheme for specific needs, comparing security and performance trade-offs, and promoting the responsible adoption of homomorphic encryption in an increasingly data-centric and privacy-conscious world. As the field continues to evolve, the taxonomy will adapt and expand, ensuring its ongoing relevance and value to the community of stakeholders invested in the security and privacy of data.

REFERENCES

[1]. Acar, A., Aksu, H., Uluagac, A.S. and Conti, M., 2018. A survey on homomorphic encryption schemes: Theory and implementation. ACM Computing Surveys (Csur), 51(4), pp.1-35.
[2]. Al Badawi, A., Bates, J., Bergamaschi, F., Cousins, D.B., Erabelli, S., Genise, N., Halevi, S., Hunt, H., Kim, A., Lee, Y. and Liu, Z., 2022, November. Openfhe: Open-source fully homomorphic encryption library. In Proceedings of the 10th Workshop on Encrypted Computing & Applied Homomorphic Cryptography (pp. 53-63).
[3]. Alloghani, M., Alani, M.M., Al-Jumeily, D., Baker, T., Mustafina, J., Hussain, A. and Aljaaf, A.J., 2019. A systematic review on the status and progress of homomorphic encryption technologies. Journal of Information Security and Applications, 48, p.102362.
[4]. Aydin, F. and Aysu, A., 2022, November. Exposing side-channel leakage of seal homomorphic encryption library. In Proceedings of the 2022 Workshop on Attacks and Solutions in Hardware Security (pp. 95-100).
[5]. Chait, K., Laouid, A., Laouamer, L. and Kara, M., 2021, November. A multi-key based lightweight additive homomorphic encryption scheme. In 2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP) (pp. 1-6). IEEE.

[6]. Chaudhary, P., Gupta, R., Singh, A. and Majumder, P., 2019, September. Analysis and comparison of various fully homomorphic encryption techniques. In 2019 International Conference on Computing, Power and Communication Technologies (GUCON) (pp. 58-62). IEEE.
[7]. Cominetti, E.L. and Simplicio, M.A., 2020. Fast additive partially homomorphic encryption from the approximate common divisor problem. IEEE Transactions on Information Forensics and Security, 15, pp.2988-2998.
[8]. Fang, H. and Qian, Q., 2021. Privacy preserving machine learning with homomorphic encryption and federated learning. Future Internet, 13(4), p.94.
[9]. Gan, W., Chen, X., Wang, W., Chen, L., Wu, J., Wang, X., He, X. and Wu, F., 2022, July. Multi-device Continuous Authentication Mechanism Based on Homomorphic Encryption and SVM Algorithm. In International Conference on Artificial Intelligence and Security (pp. 625-638). Cham: Springer International Publishing.
[10]. Jiang, C. and Pang, Y., 2020. Encrypted images-based reversible data hiding in Paillier cryptosystem. Multimedia Tools and Applications, 79, pp.693-711.
[11]. Khan, K. and Goodridge, W., 2017. SAND and Cloud-based Strategies for Adaptive Video Streaming. International Journal of Advanced Networking and Applications, 9(3), pp.3400-3410.
[12]. Khan, K. and Goodridge, W., 2018. Future DASH applications: A survey. International Journal of Advanced Networking and Applications, 10(2), pp.3758-3764.
[13]. Khan, K. and Goodridge, W., 2019. Stochastic Dynamic Programming in DASH. International Journal of Advanced Networking and Applications, 11(3), pp.4263-4269.
[14]. Khan, K. and Goodridge, W., 2020. Reinforcement Learning in DASH. International Journal of Advanced Networking and Applications, 11(5), pp.4386-4392.
[15]. Khan, K. and Goodridge, W., 2021. QoE Evaluation of Legacy TCP Variants over DASH. International Journal of Advanced Networking and Applications, 12(5), pp.4656-4667.
[16]. Kogiso, K., 2020. Encrypted control using multiplicative homomorphic encryption. Privacy in Dynamical Systems, pp.267-286.
[17]. Lee, J.W., Kang, H., Lee, Y., Choi, W., Eom, J., Deryabin, M., Lee, E., Lee, J., Yoo, D., Kim, Y.S. and No, J.S., 2022. Privacy-preserving machine learning with fully homomorphic encryption for deep neural network. IEEE Access, 10, pp.30039-30054.
[18]. Liu, J., Zhao, K. and Zhang, R., 2020. A fully reversible data hiding scheme in encrypted images based on homomorphic encryption and pixel prediction. Circuits, Systems, and Signal Processing, 39(7), pp.3532-3552.
[19]. Mahato, G.K. and Chakraborty, S.K., 2023. A comparative review on homomorphic encryption for cloud security. IETE Journal of Research, 69(8), pp.5124-5133.
[20]. Marandi, A., Alves, P.G.M., Aranha, D.F. and Jacobsen, R.H., 2023. Lattice-Based Homomorphic Encryption For Privacy-Preserving Smart Meter Data Analytics. The Computer Journal, p.bxad093.
[21]. Marcolla, C., Sucasas, V., Manzano, M., Bassoli, R., Fitzek, F.H. and Aaraj, N., 2022. Survey on fully homomorphic encryption, theory, and applications. Proceedings of the IEEE, 110(10), pp.1572-1609.
[22]. Marcolla, C., Sucasas, V., Manzano, M., Bassoli, R., Fitzek, F.H. and Aaraj, N., 2022. Survey on fully homomorphic encryption, theory, and applications. Proceedings of the IEEE, 110(10), pp.1572-1609.
[23]. Onoufriou, G., Hanheide, M. and Leontidis, G., 2022. EDLaaS: Fully Homomorphic Encryption Over Neural Network Graphs for Vision and Private Strawberry Yield Forecasting. Sensors, 22(21), p.8124.
[24]. Özerk, Ö., Elgezen, C., Mert, A.C., Öztürk, E. and Savaş, E., 2022. Efficient number theoretic transform implementation on GPU for homomorphic encryption. The Journal of Supercomputing, 78(2), pp.2840-2872.
[25]. Park, J. and Tibouchi, M., 2020, September. SHECS-PIR: somewhat homomorphic encryption-based compact and scalable private information retrieval. In European Symposium on Research in Computer Security (pp. 86-106). Cham: Springer International Publishing.
[26]. Park, J., 2021. Homomorphic encryption for multiple users with less communications. Ieee Access, 9, pp.135915-135926.
[27]. Paul, B., Yadav, T.K., Singh, B., Krishnaswamy, S. and Trivedi, G., 2022. A resource efficient software-hardware co-design of lattice-based homomorphic encryption scheme on the FPGA. IEEE Transactions on Computers, 72(5), pp.1247-1260.
[28]. Sultan, A., Tahir, S., Tahir, H., Anwer, T., Khan, F., Rajarajan, M. and Rana, O., 2022. A novel image-based homomorphic approach for preserving the privacy of autonomous vehicles connected to the cloud. IEEE Transactions on Intelligent Transportation Systems, 24(2), pp.1936-1948.
[29]. Susilo, Willy, Dung Hoang Duong, Huy Quoc Le, and Josef Pieprzyk. "Puncturable encryption: a generic construction from delegatable fully key-homomorphic encryption." In Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part II 25, pp. 107-127. Springer International Publishing, 2020.
[30]. Tan, B.H.M., Lee, H.T., Wang, H., Ren, S. and Aung, K.M.M., 2020. Efficient private comparison queries over encrypted databases using fully homomorphic encryption with finite fields. IEEE Transactions on Dependable and Secure Computing, 18(6), pp.2861-2874.
[31]. Wood, A., Najarian, K. and Kahrobaei, D., 2020. Homomorphic encryption for machine learning in medicine and bioinformatics. ACM Computing Surveys (CSUR), 53(4), pp.1-35.

[32]. Wu, T., Zhao, C. and Zhang, Y.J.A., 2021. Privacy-preserving distributed optimal power flow with partially homomorphic encryption. IEEE Transactions on Smart Grid, 12(5), pp.4506-4521.

[33]. Yang, W., Wang, S., Cui, H., Tang, Z. and Li, Y., 2023. A Review of Homomorphic Encryption for Privacy-Preserving Biometrics. Sensors, 23(7), p.3566.