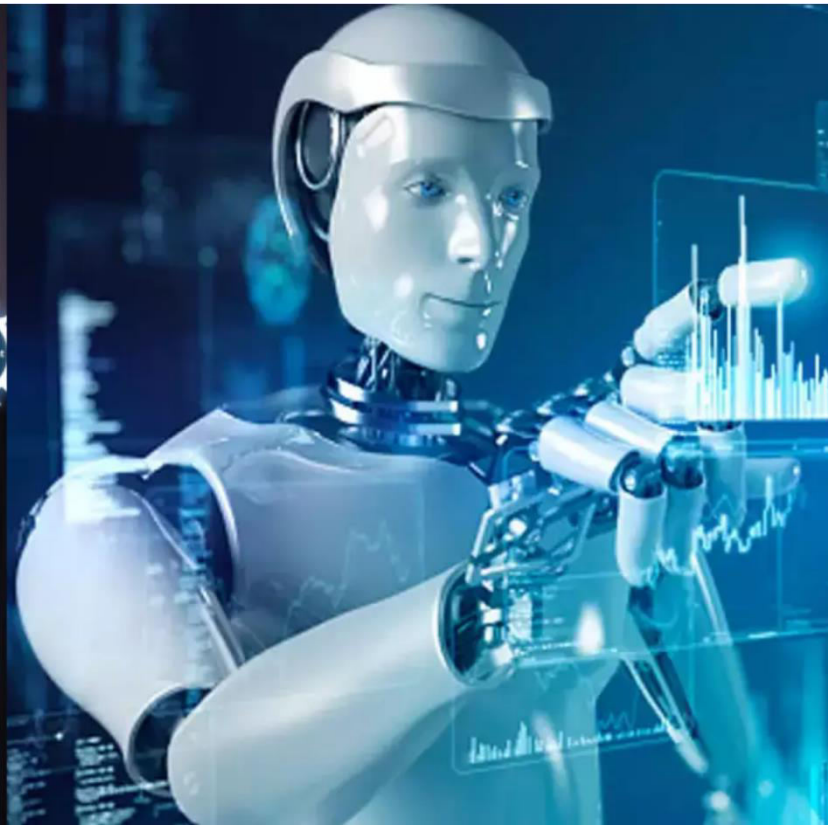
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.699

Volume 14, Issue 5, May 2025

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 5, May 2025

|DOI: 10.15680/IJIRSET.2025.1405337|

Homomorphic Encryption in Cloud Computing

Sujith Kumar R, Tejas Muralidhar Revankar, Dr.Pushpa J

P.G Student, Department of Computer Science and IT, Jain Deemed-to-be University, Bengaluru, India

P.G Student, Department of Computer Science and IT, Jain Deemed-to-be University, Bengaluru, India

Assistant Professor, Department of Computer Science and IT, Jain Deemed-to-be University, Bengaluru, India

ABSTRACT: As the cloud computing increases, data-privacy during computation is of much importance. Conventional encryption involves redecrypting data before processing making sensitive information vulnerable. Homomorphic encryption (HE) solves this limitation where encryption permits computations on ciphertexts without learning the underlying plaintext. In this paper, a survey of HE for secure cloud computing is conducted. we describe HE and its variations (PHE, SHE, FHE), discuss important schemes and libraries (e.g., SEAL, HElib, PALISADE), and present such applications as secure outsourcing, privacy-preserving ML, and encrypted search. We examine the FHE principles (lattice-based RLWE security, noise growth and bootstrapping, approximate schemes like CKKS) and the performance (numerous times slower than plaintext computing). Some of them are the high computational expense, in addition to intricate key and parameter management, and also implementation barriers. We present examples of HE in use (e.g., this is a private search from Apple, encrypted ML from AWS SageMaker), and detail how future work can be done (real-time FHE, integrated with AI, and easier frameworks). Generally, HE delivers strong privacy for cloud workloads but faces immense performance and usability challenges.

KEYWORDS: Homomorphic Encryption (HE), Fully Homomorphic Encryption (FHE), Somewhat Homomorphic Encryption (SHE), Partially Homomorphic Encryption (PHE), Bootstrapping, Data Confidentiality, Parameter Management, Smart Grids, Encryption Schemes, Cryptographic Primitives, Secure Outsourced Computation

I. INTRODUCTION

The cloud computing provides tremendous storage and compute capabilities but offloading the data to untrusted servers creates privacy concerns. Traditional schemes involve encryption of data at rest with the subsequent decryption for the processing purposes. however, decyphering exposes raw data to the cloud, bastardizing privacy. Besides, managing and protecting decryption keys in the distributed clouds are complex and have attack diversities. Partially devoted to “data in use” protection, Trusted Execution Environments (TEEs) ask to trust hardware vendors, sometimes do not solve all threat models.

There is a potential cryptographic technique breakthrough, known as homomorphic encryption (HE). overcomes these limitations. An HE scheme enables a cloud to compute arbitrary functions on encrypted data without ever figuring out the plaintext. For instance, a client may encrypt its query (or a model) and send it to the server. The server computes on this ciphertext, produces an encrypted result, and sends the result back, which the client decrypts. The server does not get to see raw data or secret keys, thus, the client’s information is always under cover. Homomorphic schemes therefore allow for secure computation outsourcing and promote disclosure of sensitive data (e.g. in finance or healthcare) without apprehensive feelings of leaks. In this paper, we look at HE’s role in cloud privacy. We cover the basics of HE, its history, survey major schemes and libraries, examine cloud-specific applications (safe outsourcing, privacy-saving ML, and encrypted search), analyze performance and scaling problems, and give representative cases. We also list existing challenges (mostly computation overhead and key management) and indicate future directions to make HE operational in the context of cloud systems.

II. DIFFERENTIATION BETWEEN HOMOMORPHIC ENCRYPTION, HETEROMORPHIC ENCRYPTION, AND CONFIDENTIALITY

Today, protecting confidential information in networks depends greatly on the use of encryption and privacy measures. While homomorphic encryption, heteromorphic encryption and confidentiality are commonly talked about, they all

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 5, May 2025

|DOI: 10.15680/IJIRSET.2025.1405337|

have distinct roles, ways they function and the areas they work in. The purpose of this section is to make it clear which aspects are different about these terms and how they are used.

1. Homomorphic Encryption

With homomorphic encryption, computations may be done on encrypted data without first needing to decode the information. This property is strongly sought in secure multi-party computations, cloud computing and preserving privacy when analyzing data. Homomorphic encryption protects data privacy by allowing calculations to be performed on the encrypted data. At the same time, using FHE can be challenging for both quick and large-scale projects due to the high computational demands.

2. Heteromorphic Encryption

Even though heteromorphic encryption does not appear often in cryptographic writing, the term is sometime used for systems that support many kinds of operations on ciphertexts or those that have several algebraic forms. Sometimes, the label is used in place of multi-algebraic or asymmetric encryption schemes that support computation. Unlike homomorphic encryption which works with the same set of operations everywhere in the ciphertext, heteromorphic approaches allow the possibility to use other operations or combine them for various types of ciphertexts. Some experts use the term heteromorphic to describe multi-key or multi-context encryption systems able to handle different types of data or information. If the term is used in any scholarly or technical writing, authors should explain how they intend to use it. While heteromorphic encryption is not widely studied in cryptography, it is developing and might be useful in decentralized privacy-preserving fields and federated learning.

3. Confidentiality

Confidentiality in information security involves protecting data from being accessed by anyone who should not have access to it. It ensures the privacy of information by only allowing access to authorized users. Homomorphic and heteromorphic encryption are ways to keep data hidden (along with other aims), while confidentiality is simply an overall security objective, not a task. Confidentiality can be secured in various methods such as: Types of encryption include symmetric and asymmetric ones. Access control is supported by systems. Masking and tokenization are two methods of data security. Examples of network security protocols are SSL/TLS. How effective confidentiality becomes depends on how safe the cryptographic tools, deployment of cryptographic keys and established security policies are. Confidentiality is additionally required by law in healthcare, finance and government due to regulations (HIPAA, GDPR), indicating its importance goes past the technical aspect.

III. BACKGROUND AND RELATED WORK ON HOMOMORPHIC ENCRYPTION

Homomorphic encryption (HE) refers to encryption schemes supporting computations on ciphertexts that correspond to operations on the underlying plaintexts. Formalized by Rivest et al. in 1978, an HE scheme can support addition or multiplication (or both) on encrypted data. Schemes are classified into three categories:

Partially Homomorphic Encryption (PHE): Supports only one type of operation. For example, the Paillier cryptosystem supports unlimited additive homomorphism, whereas RSA or ElGamal support only multiplicative homomorphism. PHE schemes are efficient but limited to special use-cases (e.g. voting tallies or simple aggregations).

Somewhat Homomorphic Encryption (SHE): Allows both addition and multiplication, but only up to a fixed circuit depth before noise accumulation corrupts the result. Early lattice-based SHE schemes could evaluate a small number of multiplications before “decryption noise” becomes prohibitive.

Fully Homomorphic Encryption (FHE): Supports arbitrary computation on ciphertexts, enabling **unbounded** addition and multiplication. The first FHE scheme was proposed by Gentry in 2009, using ideal lattices and a key technique called **bootstrapping**. This breakthrough showed that any computation on encrypted data is possible in principle.

Modern FHE schemes typically rely on lattice-based cryptography (e.g. Ring Learning With Errors, RLWE) for security. RLWE provides strong (post-quantum) hardness assumptions, ensuring encrypted data resists quantum

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 5, May 2025

|DOI: 10.15680/IJIRSET.2025.1405337|

attackers. All state-of-the-art HE libraries (SEAL, HElib, PALISADE, OpenFHE, TFHE, etc.) implement schemes built on RLWE or related lattices.

Key research developments include leveled FHE (which fixes a maximum operation depth without bootstrapping) and more practical bootstrapping algorithms. Gentry's original construction introduced bootstrapping, which homomorphically evaluates the decryption circuit itself to "refresh" ciphertext noise. Later works (Brakerski–Gentry–Vaikuntanathan (BGV), Brakerski/Fan–Vercauteren (BFV), Cheon–Kim–Kim–Song (CKKS), and TFHE) improved efficiency, noise growth, and supporting real-number operations. For instance, the CKKS scheme allows approximate arithmetic on real/complex numbers (useful for ML) by tolerating small precision errors.

A mature ecosystem of software libraries has emerged: Microsoft SEAL (BFV/CKKS), IBM HElib (BGV/BFV), PALISADE/OpenFHE (BGV/BFV/CKKS), TFHE (bitwise FHE), Concrete (Zama's library for FHE), and others. These open-source toolkits provide APIs for encryption, evaluation, and key management. They are increasingly integrated into cloud platforms (e.g., Azure's Confidential Computing offers SEAL-based services) and research prototypes. Notably, academic studies like CryptoNets (Naehrig et al., 2016) demonstrate neural-network inference on encrypted medical data, highlighting FHE's potential for private AI.

IV. APPLICATIONS IN CLOUD COMPUTING

Homomorphic encryption enables several important cloud services while preserving data privacy:
Secure Outsourced Computation: Clients can offload data-intensive tasks to the cloud without revealing inputs. For example, a user encrypts their data locally and sends it to a cloud server, which performs the computation homomorphically. The server returns an encrypted result, which only the client can decrypt. This model is analogous to outsourcing a function f(x) to a server: homomorphic encryption ensures f can be evaluated on encrypted x without the server ever seeing the raw data. In effect, HE provides a form of confidential computing: even the cloud provider cannot learn anything about the data it is processing.

Privacy-Preserving Machine Learning: HE allows training and inference of ML models on encrypted data. For instance, a medical institution could encrypt patient records and have a cloud AI service compute predictions or aggregate statistics without accessing raw patient data. CryptoNets and follow-up works show that neural networks can be adapted to FHE parameters (often using only addition and multiplication) and run on encrypted images or genomic data, yielding accurate results. Privacy-preserving ML via FHE is an active area: cloud ML platforms (like Amazon SageMaker) are now adding FHE to enable encrypted model inference. Zama's Concrete ML and other frameworks simplify building FHE-friendly AI workflows.

Encrypted Search and Retrieval: Homomorphic techniques support private information retrieval (PIR) and searchable encryption. A client can encrypt a query (e.g. a keyword or ID) and send it to a database server. The server uses HE to match or compute on encrypted entries and returns an encrypted answer. For example, Apple's recent work uses HE for **private query routing**: an iPhone encrypts a request (such as a URL to check), the server homomorphically tests the request against a protected database, and returns an encrypted Boolean or data label. The client decrypts it and sees the result (e.g. "safe" or "blocked"), while the server never learns the actual URL. This enables features like encrypted database lookups (e.g. spam filtering, caller ID, or web filters) without revealing the query content to the cloud provider.

Secure Data Aggregation and Analytics: HE can also aggregate sensitive data from multiple users or sources. For example, multiple hospitals could encrypt their patient data and allow a central server to compute global statistics (means, sums, or trained models) on the combined encrypted dataset. Similarly, financial institutions might collaboratively analyze encrypted transaction data to detect fraud or risk, without exposing individual records. While some of these tasks could be done via multi-party computation (MPC), HE provides a non-interactive alternative that is easier to integrate with existing cloud pipelines.

Healthcare and Genomics: Homomorphic encryption is particularly promising in healthcare. For instance, a patient's genome or health profile could be encrypted and sent to a cloud service that assesses genetic risk or runs medical AI models, all without ever decrypting the genetic data. In fact, chain.link's educational blog highlights a use-case: you

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 5, May 2025

|DOI: 10.15680/IJIRSET.2025.1405337|

could use a genomics service without revealing your DNA sequence, thanks to HE. Similarly, cloud-based medical image analysis could operate on encrypted X-rays or MRIs, ensuring patient privacy even during AI inference. Companies and research labs are actively exploring these scenarios, since privacy is a major concern in medical data sharing.

Each of these applications leverages the core HE capability: **compute on ciphertext**. Table 1 (below) summarizes key use-cases:

- Secure Outsourcing: Encrypt data locally; cloud computes on ciphertext.
- Privacy-Preserving ML: Train or apply models on encrypted datasets.
- Encrypted Search/PIR: Homomorphic query to a database without revealing keyword.
- Data Aggregation: Sum or analyze combined encrypted contributions from multiple parties.
- Healthcare Analytics: Run clinical/genomic analysis on encrypted patient data.

V. TECHNICAL ANALYSIS

The power of FHE comes from **lattice-based cryptography** and clever noise management. Modern schemes encrypt plaintexts into ring elements with an added "noise" term that hides the message. Homomorphic operations (additions, multiplications) act on these ciphertexts and also add or multiply the noise. Security relies on the noise being large enough to hide plaintext (RLWE hardness), but each operation increases noise magnitude.

Noise and Bootstrapping: In Somewhat Homomorphic Encryption, the noise grows with each operation and eventually exceeds a threshold, making decryption fail. To achieve full homomorphism, FHE uses bootstrapping: a homomorphic self-evaluation of the decryption circuit. As explained by Gentry's scheme, bootstrapping takes a noisy ciphertext and "refreshes" it by homomorphically decrypting (using an encrypted secret key) back to a fresh ciphertext with low noise. Conceptually, this is like "pulling yourself up by your bootstraps". Bootstrapping enables unlimited depth of computation at the cost of heavy overhead. It is by far the most expensive part of FHE: the latency of an FHE bootstrapping can be orders of magnitude slower than even the most complex plaintext operations. Researchers continue to optimize bootstrapping: TFHE specializes in very fast bit-wise bootstrapping, and schemes like CKKS and BFV have optimized bootstrappers for numeric data.

Approximate Encryption (CKKS): A recent trend is approximate homomorphic encryption, exemplified by the CKKS scheme. CKKS encrypts real or complex numbers with limited precision: operations yield a close approximation to the true result. This is acceptable in many ML and signal-processing tasks. CKKS includes a "rescaling" step that manages precision and noise, trading off some accuracy for efficiency. By allowing slight imprecision, CKKS can perform non-trivial arithmetic on encrypted floating-point data, enabling encrypted AI workloads.

Performance Considerations: HE computations are extremely heavy compared to plaintext. As noted by recent studies, a computation that takes 1 second on plaintext might take days with current FHE libraries. For example, using popular libraries (HElib, PALISADE, etc.) to evaluate a simple addition-multiplication circuit could slow down by factors of 106 or more. Even with moderate improvements (custom hardware or GPUs), homomorphic operations typically run tens to thousands of times slower than unencrypted operations. Key factors include large ciphertext sizes (many kilobytes or more), expensive polynomial arithmetic in large rings, and costly bootstrapping.

To mitigate these issues, researchers explore hardware acceleration and algorithmic tweaks. For instance, some projects use GPUs or FPGAs to speed up the Number Theoretic Transforms (NTTs) and large integer arithmetic underpinning HE. The BU/Red Hat project mentioned earlier is designing FHE accelerators on FPGAs specifically for CKKS image classification. Also, leveled FHE (which omits bootstrapping) is used when the computation depth is known ahead of time, trading off maximal flexibility for much faster evaluation. Hybrid approaches may combine HE with other PETs: for example, using a TEE to handle small or critical parts of computation, while using FHE for bulk data processing.

In summary, while FHE's theoretical underpinnings are well-understood (RLWE lattices, bootstrapping, approximate homomorphism), making FHE practical involves careful parameter tuning, efficient implementation, and, often, accepting trade-offs in precision or interaction. The field has rapidly evolved since Gentry's proof-of-concept; new

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 5, May 2025

|DOI: 10.15680/IJIRSET.2025.1405337|

schemes and software improve throughput and ease of use, but fundamental limits (noise accumulation and ciphertext size) remain active areas of research.

Challenges

Despite its promise, homomorphic encryption faces significant challenges in cloud deployment:

Computational Overhead: The biggest hurdle is performance. FHE is typically many orders of magnitude slower than plaintext computation. Even simple queries or ML inferences can take seconds or minutes, hindering real-time applications. Bootstrapping adds extra latency and computational cost. High-level optimizations (parallelism, hardware acceleration) can help, but do not eliminate the gap. For many use-cases (especially those requiring low latency or high throughput), the HE slowdown is currently prohibitive.

Key and Parameter Management: HE involves large keys and complex parameters (lattice dimensions, noise budgets, plaintext moduli). Generating secure parameters that balance performance is tricky. Moreover, while HE allows the server to compute without a secret key, clients must manage public/private keys securely. In multi-user scenarios, one must handle key distribution, re-encryption, or multi-key HE protocols. These complicate integration into cloud services. For example, sharing results among users may require proxy re-encryption or threshold HE, adding layers of complexity beyond basic HE. Key management is further complicated by ensuring forward secrecy, key rotation, and compatibility with existing cloud KMS (Key Management Systems).

Implementation Complexity: Implementing HE correctly is non-trivial. Developers must avoid parameter pitfalls (e.g. insufficient noise budget) that can silently break security or correctness. Many libraries offer different APIs and parameter choices, leading to a steep learning curve. Ensuring interoperability between libraries or languages (C++, Python, JavaScript) can be challenging. Also, supporting new data types (e.g. floating-point, complex numbers, or large databases) often requires protocol design on top of raw HE, such as encoding schemes or batching (packing multiple values into one ciphertext). These engineering challenges slow adoption.

Security Considerations: Although lattice-based HE is believed to be post-quantum secure, it is relatively new. Careful review is needed to guard against subtle attacks (such as side-channel leaks or implementation bugs). Open-source libraries mitigate some concerns (through transparency), but publicly available parameters and large attack surfaces raise issues. Moreover, HE typically protects data confidentiality, but not necessarily other aspects like access patterns or result privacy. Designers must often combine HE with other techniques (e.g. differential privacy, oblivious RAM) for full privacy.

Overall, while homomorphic encryption is theoretically strong, its practical adoption in cloud computing is limited by speed, complexity, and maturity of toolchains. The “overhead” of HE remains a stubborn barrier, and many enterprise applications wait for more efficient FHE or hybrid solutions before wide-scale deployment.

VI. CASE STUDIES AND EXAMPLES

Several real-world projects and products illustrate homomorphic encryption in cloud contexts:

Apple’s Private Queries (2023–2024): Apple has integrated HE into its ecosystem for features like enhanced visual search and content filtering. In these systems, a user’s device encrypts a query and sends it to Apple’s servers. The servers compute on the encrypted query against private databases and return encrypted results. The device then decrypts to obtain the answer (e.g. identified object or block decision). Crucially, Apple’s server never sees the user’s actual query. In published blog posts, Apple engineers describe using a BFV-based HE scheme and private information retrieval (PIR) to enable encrypted database lookups for caller ID, Mail logo lookup, and web content filtering. This demonstrates HE in production: it handles millions of encrypted lookups daily while keeping user data private.

Amazon SageMaker Encrypted ML (2023): Amazon Web Services (AWS) announced support for fully homomorphic encryption in SageMaker for secure model inference. In this setup, data owners encrypt their input data before sending it to an ML model endpoint. SageMaker evaluates the model on ciphertexts and returns encrypted predictions. AWS notes that FHE “allows computations and analytical functions to be run on encrypted data” so that data can “be represented with numbers” through encoding. This enables near real-time inferencing under zero-trust

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 5, May 2025

|DOI: 10.15680/IJIRSET.2025.1405337|

policies. AWS's partnership with Leidos has showcased neural-network inference (e.g. with encrypted images) in fields like medical imaging, emphasizing that encrypted data must often be numerically encoded, and performance tuning is needed for latency requirements.

Healthcare Analytics (Research Prototypes): In academia and industry research, HE is applied to medical data analysis. For example, the BU/Red Hat project mentioned earlier is building FPGA accelerators for the CKKS scheme aimed at encrypted healthcare imaging tasks. They plan to run end-to-end experiments on encrypted X-ray and MRI classification, highlighting how hardware acceleration can enable HE workloads. Similarly, firms like Duality Technologies (now part of Duality) and startups like Zama have developed HE platforms for secure data collaboration in industries including healthcare and finance, often partnering with hospitals and research labs to test encrypted data sharing.

Finance and Genomics (Conceptual Use-Cases): While proprietary, there are reports of HE pilots in finance (e.g. credit scoring on encrypted data, collaborative risk analysis across banks) and genomics (e.g. 23andMe using HE to let users privately determine genetic relative matches). In the genomics domain, Chainlink's example illustrates the appeal: a patient could leverage cloud-based genomics analysis without revealing raw DNA. Such scenarios emphasize HE's value proposition even if public details are limited.

These examples show that HE is transitioning from theory to practice. Major tech companies are embedding HE into cloud services and consumer products, while research groups explore hardware and novel applications. Nonetheless, all these deployments must carefully balance latency and accuracy against privacy needs.

VII. FUTURE WORK

Homomorphic encryption remains an active research frontier. Key directions include:
Performance Optimization: Continued algorithmic improvements (faster bootstrapping, better encoding schemes) and hardware acceleration will help make HE more practical. Research into GPU/FPGAs for FHE is promising, as is exploiting parallelism across cloud clusters. A milestone goal is to support real-time encrypted computation (e.g. live video analytics on encrypted streams).

Integration with AI: Combining HE with advanced AI is crucial. Future frameworks may allow seamless encrypted training of deep models or encrypted federated learning. Libraries that automatically handle FHE-friendly ML model conversion (like Concrete ML) are emerging. There is also interest in hybrid models that use HE for linear parts of inference and alternative methods (like secure enclaves) for non-linear parts to boost efficiency.

Interoperable Frameworks: To broaden adoption, HE tools must be easier to use. Ongoing work aims to integrate FHE into mainstream programming environments, database systems, and big data platforms. Standardized interfaces and parameter-selection tools will reduce developer burden. Also, combining HE with complementary PETs (e.g. TEE, MPC, differential privacy) will create robust, multi-layered solutions for cloud privacy.

Policy and Standardization: As HE matures, industry standards and legal frameworks will evolve. Cloud providers may offer certified FHE services, and regulatory guidelines might encourage encrypted computation for sensitive data. Encouragingly, international bodies are beginning to explore FHE benchmarks and interoperability standards.

In summary, the path to widespread HE adoption involves co-design of cryptography, systems, and application architectures. Progress in HE will not only rely on math breakthroughs, but also on practical engineering, ecosystem development, and use-case innovation.

VIII. CONCLUSION

Homomorphic encryption offers a **paradigm shift** for secure cloud computing: it enables **privacy-preserving computation** by design, allowing untrusted servers to process data without seeing it. This capability is unique among encryption schemes and could unlock many cloud services for sensitive domains like healthcare and finance. Indeed,

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 5, May 2025

|DOI: 10.15680/IJIRSET.2025.1405337|

FHE encourages data sharing under "zero trust" policies and supports powerful applications from encrypted AI inference to confidential database queries.

However, FHE also faces steep practical hurdles. Current fully homomorphic computations are far slower and more complex than plaintext workloads. Key management, implementation complexity, and large ciphertext sizes pose challenges for integration with existing cloud platforms. As of now, FHE is best suited to scenarios where privacy is paramount and some performance trade-off is acceptable.

Looking ahead, continued advances in algorithms (bootstrapping, approximate arithmetic) and hardware will narrow the performance gap. Already, leading companies (Apple, AWS, Microsoft) and research teams (universities, startups) are proving out HE in real use-cases. When combined with other privacy technologies, HE could ultimately enable cloud computing on encrypted data at scale. The promise is compelling: full data confidentiality even in the cloud, which could transform how sensitive data is handled. Achieving that vision will require sustained effort, but the potential rewards – secure, privacy-preserving cloud services – justify the endeavor.

Sources: Peer-reviewed articles, industry research blogs, and technical documents on homomorphic encryption and cloud security. Each section above cites relevant literature to provide a thorough graduate-level overview of the topic.

REFERENCES

1. What Is Homomorphic Encryption? - IEEE Digital Privacy https://digitalprivacy.ieee.org/publications/topics/what-is-homomorphic-encryption
2. What Is Homomorphic Encryption? - Chainlink
3. https://chain.link/education-hub/homomorphic-encryption
4. Bootstrapping in Fully Homomorphic Encryption (FHE)
5. https://dualitytech.com/blog/bootstrapping-in-fully-homomorphic-encryption-fhe/
6. Preserving privacy in the cloud: speeding up homomorphic encryption with custom hardware – Red Hat Research
7. https://research.redhat.com/blog/article/privacy-in-the-cloud-speeding-up-homomorphic-encryption-with-fpgas/
8. Privacy-Preserving Cloud Computing using Homomorphic Encryption | Center for Information & Systems Engineering https://www.bu.edu/cise/research/privacy-preserving-cloud-computing-using-homomorphic-encryption/
9. Combining Machine Learning and Homomorphic Encryption in the Apple Ecosystem - Apple Machine Learning Research https://machinelearning.apple.com/research/homomorphic-encryption
10. Enable fully homomorphic encryption with Amazon SageMaker endpoints for secure, real-time inferencing | AWS Machine Learning Blog https://aws.amazon.com/blogs/machine-
11. learning/enable-fully-homomorphic-encryption-with-amazon-sagemaker-endpoints-for secure-real-time-inferencing/
12. Using Homomorphic Encrypted Data in the Real World
13. https://dualitytech.com/blog/how-to-use-homomorphic-encryption-in-the-real-world/

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details