A Framework for Privacy-Preserving Multiparty Computation with Homomorphic Encryption and Zero-Knowledge Proofs

Janak Dhokrat[1], Namita Pulgam[1], Tabassum Maktum[2] and Vanita Mane[1]
[1]Department of Computer Engineering, Ramrao Adik Institute of Technology, D Y Patil Deemed to be University, Navi Mumbai, Maharashtra, India
[2]Department of Computer Engineering, School of Engineering & Technology, Anjuman I Islam's Kalsekar Technical Campus, New Panvel, Maharashtra, India
E-mail: jdhokrat@gmail.com, namita.pulgam@rait.ac.in, tabassum.maktum@aiktc.ac.in, vanita.mane@rait.ac.in

Keywords: homomorphic encryption, RSA encryption, Paillier encryption, secure multiplication, secure addition, cryptographic analysis, secure multi-party computation

Received: July 07, 2024

In digital landscape of today's ongoing world the imperative for enhanced security in cloud-based data processing is paramount. This paper introduces an innovative framework that seamlessly integrates Homomorphic Encryption and Zero-Knowledge Proofs (ZKPs) to bolster data privacy and confidentiality. This paper explores the technical intricacies, real-world applications, and potential implications of this fusion framework. Homomorphic Encryption empowers computations on encrypted data without compromising privacy while Zero-Knowledge Proofs offer a mechanism to verify computations without exposing sensitive details. The effectiveness and adaptability of the proposed framework is demonstrated through meticulous analysis and practical deployment in safeguarding cloud-based data processing. The proposed framework marks a significant stride towards creating an environment where data security is unequivocally prioritized. Extensive computational experiments with quantitative results demonstrate substantial improvements in secure data processing, accuracy and efficiency confirming the frameworks effectiveness in preserving privacy while maintaining computational performance.

Povzetek: Predstavljen je okvir za varno večstransko računsko okolje, ki združuje homomorfno šifriranje in dokaze z ničelnim znanjem ter s tem krepi varnost obdelave podatkov v oblaku.

1 Statements and declarations

All authors have no competing interests to declare that are relevant to the content of this article . All authors certify that they have no financial interest or non-financial interest with any organization in the subject matter or materials discussed in this manuscript.

2 Introduction

In today's ever-evolving agile technological environment the proliferation of cloud computing has revolutionized on the basis of data stored, process, and utilization. With this rapid shift towards cloud-based solutions, ensuring the security and privacy of sensitive information has emerged as a critical challenge. Traditional encryption methods are effective to a certain extent but often fall short in fully addressing the complexities of securing data in remote computational environments.

2.1 RSA encryption scheme

RSA encryption is a widely used asymmetric encryption algorithm that relies on the difficulty of factoring large prime numbers. It allows for secure encryption and decryption of messages using a public key and a private key pair.

In RSA, key generation begins with the selection of two large prime numbers, which are then multiplied to produce a modulus, n. This modulus is a part of both the public and private keys. The public key also includes an exponent, e, chosen such that it is coprime with the totient of n. The private key comprises an exponent, d, calculated to satisfy the congruence relation involving e and the totient of n.

Encryption in RSA involves transforming a plaintext message into ciphertext by raising the message to the power of the public exponent e and taking the remainder when divided by the modulus n. This process secures the message such that only the holder of the private key can decrypt it. Decryption requires raising the ciphertext to the power of the private exponent d and then taking the remainder modulo n, thereby retrieving the original plaintext message.

A notable feature of RSA is its multiplicative homomorphism. This property implies that the product of two encrypted messages, once decrypted, equates to the product of the original plaintexts. In the context of secure multiparty computation, RSA encryption is leveraged for its multiplicative homomorphic property. This property enables computations to be performed on ciphertexts without the need to decrypt them. Specifically, when two ciphertexts encrypted using RSA encryption are multiplied together, the resulting ciphertext corresponds to the product of the original plaintexts.

2.2 Paillier encryption scheme

The Paillier encryption scheme, introduced by Pascal Paillier in 1999, is a probabilistic public-key cryptosystem renowned for its additive homomorphic properties. Its security is founded on the complexity of solving the composite residuosity class problem, which relates to the difficulty of computing discrete logarithms in composite moduli. Paillier encryption is another asymmetric encryption scheme that supports homomorphic addition. Unlike RSA, which enables multiplicative homomorphism, Paillier encryption allows for secure addition operations on ciphertexts.

In Paillier encryption, key generation starts with the selection of two large prime numbers, similar to RSA, which are multiplied to yield a modulus n. Additionally, a random integer g is chosen such that certain mathematical conditions involving n are satisfied. The public key consists of n and g, while the private key includes values derived from the prime factors of n and other computations involving g.

Encryption under the Paillier scheme involves raising the generator g to the power of the plaintext and then multiplying it by a random number raised to the power of n. The result is taken modulo n^2, producing the ciphertext. Decryption requires using the private key to perform specific operations that extract the original plaintext from the ciphertext.

The distinguishing feature of Paillier encryption is its additive homomorphism. This property allows one to compute the sum of two plaintexts by multiplying their corresponding ciphertexts. Consequently, addition operations can be carried out on encrypted data without the need to decrypt it first, which is advantageous for applications requiring privacy-preserving computations and secure data aggregation. Hence in secure multiparty computation scenarios, Paillier encryption is utilized for its ability to perform addition operations directly on encrypted data. When two ciphertexts encrypted using the Paillier cryptosystem are added together, the resulting ciphertext represents the sum of the original plaintexts.

To confront challenges of traditional encryption schemes, this paper introduce a groundbreaking framework that combines the power of Homomorphic Encryption and Zero Know-ledge Proofs (ZKPs). The proposed framework can provide an unparalleled level of protection for sensitive data. Through this comprehensive exploration it helps to unveil the theoretical underpinnings of these cryptographic techniques but also demonstrate their practical applications in real-world scenarios.

The objective of this paper is: i) To fortify the security and confidentiality of cloud-based data processing, thereby establishing a robust ecosystem where data privacy is non-negotiable, ii) To delve into the technical intricacies of the framework, elucidating the merging of Homomorphic Encryption and Zero-Knowledge Proofs, iii) To proposed the framework where combination of Homomorphic Encryption and Zero-Knowledge Proofs is added as a safety layer, iv) To contribute to the establishment of a secure and trustworthy cloud computing environment, where individuals and organizations can confidently harness the power of the cloud without compromising the integrity of their data.

The rest of the paper is organized as follows: Introduction on the concept of RSA and Paillier encryption schemes is provided in section I. Section II provides a summary of related work on RSA and Paillier encryption methodologies . An in-depth comparative analysis of RSA and Paillier encryption techniques provided in section III. Section IV provides a summary of RSA and Paillier encryption schemes collaboration methods and conclusion is in section V.

3 Related work

An extensive literature review on RSA and Paillier encryption schemes is provided in this section. The survey summarizes key findings, distinguishing factors, and commonalities among the techniques.

In paper [1] (R. T.A.V.Y *et al.*): Proposed a novel method to safeguard electronic health records (EHR) by synergistically employing zero-knowledge proofs (ZKP) and blockchain encryption. Their approach aims to comprehensively preserve patient privacy and data integrity within healthcare information systems, addressing critical security concerns.

In paper [2] (Z. H. Mahmood *et al.*): They introduced an innovative fully homomorphic encryption scheme founded on the principles of multistage partial homomorphic encryption, specifically tailored for applications in cloud computing environments. Their scheme enables computations to be performed directly on encrypted data residing in the cloud while rigorously maintaining data confidentiality.

In paper [3] (Tang): This paper explored the potential of homomorphic encryption technology within the context of computer cloud computing. It delved into the prospects of utilizing homomorphic encryption to facilitate secure computations on encrypted data stored in cloud environments, while also analyzing the associated technical challenges and limitations.

In paper [4] (A. Pathak *et al.*): They presented a novel secure authentication method grounded in the principles of zero-knowledge proofs. Their approach enhances the security of digital authentication processes by enabling verification without the need to disclose sensitive information, thereby preserving privacy and mitigating potential data breaches.

In paper [5] (D. Čapko *et al.*): This paper provided a comprehensive overview of the current state-of-the-art in the application of zero-knowledge proofs within the realm of blockchain technology. It discussed recent advancements and pioneering use cases that leverage zero-knowledge proofs to bolster privacy and security in blockchain-based systems and transactions.

In paper [6] (C. P. Sah): The author conducted a rigorous robustness analysis of zero-knowledge proof systems that employ the Diffie-Hellman problem as their underlying cryptographic foundation. The study evaluated the security strengths and efficiency aspects of these zero-knowledge proofs, with a particular focus on their resistance against various cryptographic attacks.

In paper [7] (S. Liu): This paper explored the profound impact of zero-knowledge proofs on the privacy protection landscape, heralding a paradigm shift in secure authentication and data exchange methodologies. It examined how zero-knowledge proofs enable parties to verify information and authenticate without revealing sensitive data, revolutionizing privacy-preserving technologies.

In paper [8] (D. Hou *et al.*): The authors proposed a novel privacy-preserving energy trading system that synergistically combines blockchain technology and zero-knowledge proofs. Their system ensures secure and private transactions within energy markets by leveraging the immutability and decentralization of blockchain, while zero-knowledge proofs protect sensitive information during the trading process.

In paper [9] (Singh *et al.*): This research investigated the application of zero-knowledge proofs in facilitating verifiable and transparent decentralized AI pipelines. It explored how zero-knowledge proofs can be employed to ensure the integrity and auditability of AI processes in decentralized environments, enabling stakeholders to verify the correctness of AI models and outputs without compromising data privacy.

In paper [10] (A. Kavya *et al.*): The authors conducted a comprehensive comparative study evaluating the performance, security characteristics, and practical feasibility of various homomorphic encryption schemes within the context of cloud computing applications. Their analysis provided valuable insights into the strengths and limitations of different homomorphic encryption approaches in cloud-based scenarios.

In paper [11] (J. Liu *et al.*): They investigated a novel searchable encryption scheme tailored for cloud environments that leverages fully homomorphic encryption. Their proposed scheme enables users to perform searches on encrypted data stored in the cloud without compromising data privacy, addressing a critical requirement in secure cloud computing.

In paper [12] (S. Yaji *et al.*): This paper explored privacy-preserving techniques for blockchain systems, with a particular focus on leveraging partial homomorphic encryption systems for AI applications. The authors presented methods to enhance privacy within blockchain-based AI systems by employing partial homomorphic encryption, enabling secure computations while protecting sensitive data.

In paper [13] (Swathi Velugoti *et al.*): The paper highlights how cloud computing improves business operations by providing scalable and cost-effective data management. Traditional encryption methods, while protecting data, restrict its usability in cloud environments. To solve this issue, the study presents a method using privacy homomorphism, allowing secure data processing. It also analyzes the system's efficiency and security performance.

In paper [14] (J. Ryu *et al.*): This study comprehensively investigated the properties and applications of partially homomorphic encryption schemes. The authors explored the mathematical foundations, security characteristics, and practical use cases of these schemes, which enable specific computations to be performed on encrypted data while preserving confidentiality.

In paper [15] (K. El Makkaoui *et al.*): The authors explored the practicality and feasibility of implementing hybrid homomorphic encryption schemes in real-world scenarios. They examined the challenges and trade-offs associated with these hybrid approaches that combine multiple encryption techniques, questioning whether they can be practically deployed while meeting the required security and performance standards.

In paper [16] (Wid Akeel Awadh*et al.*):The paper examines security issues with healthcare data stored in the cloud. It suggests using Triple Data Encryption Standard (3DES) for stronger data protection by increasing key length. The findings show that 3DES enhances both security and efficiency, making it a reliable method for securing healthcare data in cloud systems.

Table 1: Comparative analysis of related Work in secure multiparty computation - part 1

Author(s)	**Year**	**Key Methods**
Mahmood et al.	2018	Multi-stage and fully homomorphic encryption
Tang	2022	Homomorphic encryption techniques
Pathak et al.	2021	Zero-knowledge proof systems
Čapko et al.	2022	ZKP and blockchain integration
Liu	2022	Zero-knowledge proof methodology
Singh et al.	2022	ZKP in decentralized AI
Kavya et al.	2018	Homomorphic encryption analysis
Liu et al.	2016	Fully homomorphic and searchable encryption
Ryu et al.	2023	Partially homomorphic encryption schemes
El Makkaoui et al.	2016	Hybrid homomorphic encryption approach

4 The proposed framework for secure multiparty computation

The need for a framework for secure multiparty computation arises from the requirements of collaborative computing environments where multiple parties need to jointly analyze or process sensitive data while preserving privacy

Table 2: Comparative analysis of related work in secure multiparty computation - part 2

Main Benefits	**Research Focus**	**Challenges Noted**
Cloud-based encrypted data processing	Enhancing fully homomorphic encryption efficiency in cloud	Processing intensity, expansion difficulties
Secure cloud-based computations on encrypted data	Overcoming technical hurdles in cloud homomorphic encryption	Efficiency constraints, intricate key handling
Improved authentication security without data exposure	Advancing ZKP-based secure authentication	Possible delays, complex implementation
Enhanced blockchain privacy and security	ZKP application for blockchain security improvement	Scaling challenges, high computational demands
Data verification without revealing sensitive information	Analyzing ZKP's impact on data privacy protection	Potential proof system weaknesses, efficiency issues
Ensures AI process integrity in decentralized settings	ZKP for transparent, verifiable decentralized AI workflows	Integration complexities with AI, performance impact
Comprehensive review of cloud-based homomorphic encryption	Comparing various homomorphic encryption methods for cloud use	Security-efficiency balance, limited real-world applications
Privacy-preserving cloud data search capabilities	Developing homomorphic searchable encryption for cloud	High computation needs, real-time operation difficulties
Specific encrypted data computations with privacy	Exploring partially homomorphic encryption applications	Restricted operation set, potential security risks
Combines various encryption methods	Assessing feasibility of hybrid homomorphic encryption	Increased system intricacy, potential integration security issues

and ensuring integrity. The proposed framework for secure multiparty computation using RSA and Paillier encryption with zero-knowledge proofs (ZKPs) is designed to address the challenges of privacy, integrity, and interoperability in collaborative computing scenarios. This framework allows computations on encrypted data that enables collaboration without revealing the underlying sensitive information. Figure 1 shows the proposed framework. By leveraging homomorphic addition with paillier encryption and RSA encryption with multiplicative Homomorphism, the protocol ensures that computations can be performed on encrypted data without compromising privacy.

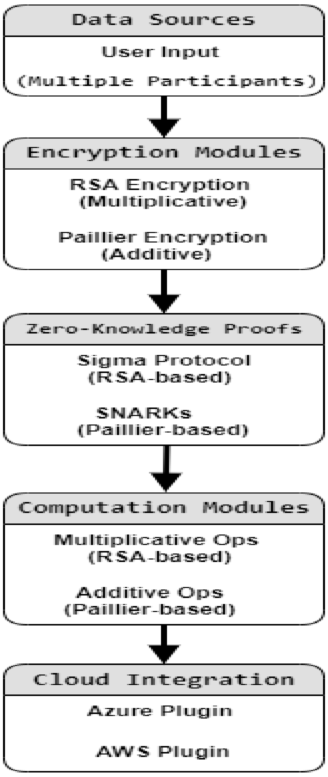


Figure 1: Proposed framework for secure multiparty computation

The proposed framework design utilizes the techniques such as Paillier encryption, RSA encryption and ZKPs. The applied encryption system is either partially homomorphic encryption (Paillier encryption) and somewhat homomorphic encryption (RSA encryption). The flow of the system with these techniques used in the framework is shown in Figure 2.

The major components of the proposed framework are:

RSA Encryption (Multiplication):

1. RSA encryption is employed to support multiplicative homomorphism, allowing computations on encrypted data.

2. It involves the generation of public and private keys, where the public key is used for encryption and the private key for decryption.

3. Encrypted data can be multiplied without the need to Decrypt it, preserving the confidentiality of the underlying plain-texts.

Paillier Encryption (Addition):

1. Paillier encryption is utilized for additive homomorphism, enabling computations on encrypted values while maintaining privacy.

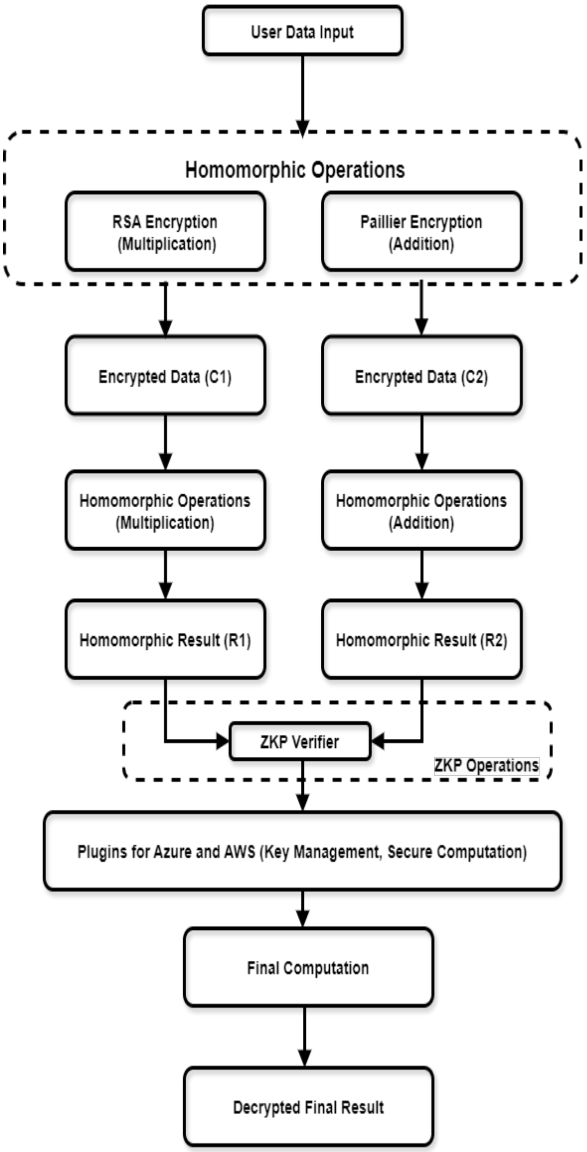

User Data Input
Homomorphic Operations
RSA Encryption (Multiplication)
Paillier Encryption (Addition)
Encrypted Data (C1)
Encrypted Data (C2)
Homomorphic Operations (Multiplication)
Homomorphic Operations (Addition)
Homomorphic Result (R1)
Homomorphic Result (R2)
ZKP Verifier
ZKP Operations
Plugins for Azure and AWS (Key Management, Secure Computation)
Final Computation
Decrypted Final Result

Figure 2: Flow of the designed framework

2. Similar to RSA, it involves key generation, encryption, and decryption operations.

3. Encrypted values can be added together Homomorphically, enabling collaborative computations without exposing individual inputs.

Zero-Knowledge Proofs (ZKPs):

1. ZKPs provide a mechanism for proving the validity of computations without revealing any information about the inputs.

2. Sigma protocols and Succinct Non-interactive Arguments of Knowledge (SNARKs) are commonly used ZKP techniques.

3. Participants can generate proofs to demonstrate the correctness of their computations, which can be independently verified by other parties.

4.1 Homomorphic encryption

Homomorphic Encryption is a cryptographic technique that enables computations to be performed on encrypted data without decrypting it. This property is particularly valuable in scenarios where privacy is paramount, such as outsourcing computations to untrusted servers or conducting analyses on sensitive data.

4.1.1 Algorithm for homomorphic encryption (simplified Paillier cryptosystem)

1. Key Generation:

1. Choose two large prime numbers, p and q.

2. Compute $n = p \times q$ and $\lambda = \text{lcm}(p-1, q-1)$.

3. Select a random integer g such that g is a generator modulo n^2.

4. The public key is (n, g) and the private key is λ.

2. Encryption:

1. To encrypt a plaintext m, select a random integer r such that $0 < r < n$.

2. Compute the ciphertext as $c = g^m \times r^n \mod n^2$.

3. Decryption:

1. To decrypt a ciphertext c, compute $L(c^\lambda \mod n^2) \times \mu \mod n$, where $L(x) = \frac{x-1}{n}$ and μ is the modular multiplicative inverse of $L(g^\lambda \mod n^2)$ modulo n.

4. Homomorphic Addition:

– Given two ciphertexts c_1 and c_2 representing plaintexts m_1 and m_2 respectively, the homomorphic addition of c_1 and c_2 results in a new ciphertext c' representing the sum of m_1 and m_2.

4.2 Zero-knowledge proofs (ZKPs)

Zero-Knowledge Proofs are cryptographic protocols that allow one party, the prover, to convince another party, the verifier, that a statement is true without revealing any additional information beyond the validity of the statement itself. ZKPs are widely used in various cryptographic applications, including authentication, identity verification, and privacy-preserving computations. Working of the ZKPs is shown in Figure 3.

In the context of secure multiparty computation, ZKPs play a crucial role in verifying the correctness of the computations performed on encrypted data. They ensure that the operations carried out on ciphertexts, whether multiplication or addition, are done correctly without disclosing the actual plaintext values.

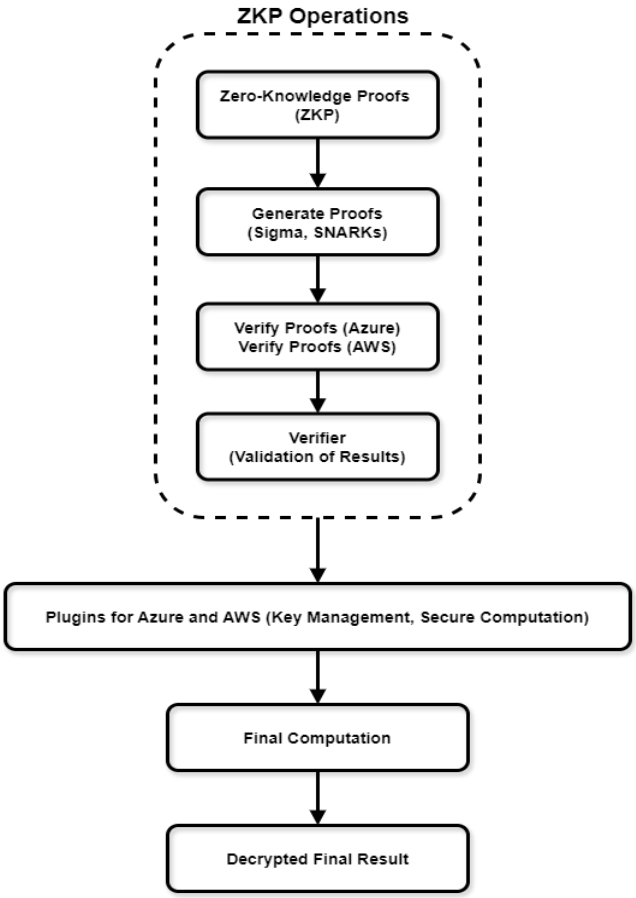

ZKP Operations
Zero-Knowledge Proofs (ZKP)
Generate Proofs (Sigma, SNARKs)
Verify Proofs (Azure)
Verify Proofs (AWS)
Verifier (Validation of Results)
Plugins for Azure and AWS (Key Management, Secure Computation)
Final Computation
Decrypted Final Result

Figure 3: Generalized working of ZKPs

4.2.1 Algorithm for a zero-knowledge proof of knowledge of a discrete logarithm (simplified)

1. Setup:

1. Select a large prime p and a generator g of the multiplicative group modulo p.

2. Choose a secret value x and compute $y = g^x \mod p$.

2. Prover:

1. The prover randomly selects a value r and computes $t = g^r \mod p$.

2. The prover sends t to the verifier.

3. Depending on the verifier's challenge, if $b = 0$, the prover sends r to the verifier; if $b = 1$, the prover computes $c = r + x \mod (p - 1)$ and sends c to the verifier.

3. Verifier:

1. The verifier checks if $g^c \equiv t \times y^b \mod p$. If the equation holds, the prover has successfully proven knowledge of x without revealing it.

4.3 AWS plugins

An application's or system's interoperability is its capacity to exchange data automatically and securely across organizational or geographic borders. It makes it possible for many systems to communicate with one another and exchange data instantly. Interoperability solutions assist organizations in achieving communications that comply with industry standards and minimize data silos. The proposed framework seamlessly integrates with different cloud providers such as AWS ensures interoperability and enables collaboration across various infrastructures.

Separate plugins tailored for AWS cloud platforms facilitate the deployment of the framework in diverse computing environments. These plugins manage cryptographic operations, key generation, secure computation, and proof verification. Interoperability between cloud platforms enables seamless collaboration and data sharing across different infrastructures.

5 Combining techniques for secure multiparty computation

In a secure multiparty computation scenario, RSA encryption with multiplicative homomorphism is employed for secure multiplication operations, while Paillier encryption facilitates homomorphic addition operations. These encrypted computations allow multiple parties to jointly perform computations on their respective encrypted data without exposing sensitive information.

Additionally, Zero-Knowledge Proofs are utilized to verify the integrity and correctness of the computations performed on the encrypted data. By employing ZKPs, the parties involved can ensure that the computed results are accurate without revealing any details about the underlying plaintext values.

Together, these techniques form a robust framework for secure multiparty computation, enabling privacy-preserving collaboration and computation in scenarios where data confidentiality is paramount. This approach ensures that sensitive information remains encrypted throughout the computation process, with verification mechanisms in place to validate the integrity of the results without compromising data privacy.

5.1 Homomorphic RSA encryption with ZKPs for multiplication

This process demonstrates how RSA encryption with its multiplicative homomorphic property can be used for secure multiplication of encrypted data. The sigma protocol for ZKPs enables the verifier to validate the correctness of the multiplication operation without revealing the plaintexts or the private key. This process involves various operation which are discussed further.

1. Key Generation

1. Choose two distinct large prime numbers p and q.
2. Compute $n = p \times q$.
3. Calculate $\phi(n) = (p-1) \times (q-1)$.
4. Choose an integer e such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$.
5. Compute d such that $(d \times e) \mod \phi(n) = 1$.
6. Public Key: (e, n).
7. Private Key: (d, n).

2. Encryption

1. Plaintext: $M1, M2$ ($M1 < n$, $M2 < n$).
2. Ciphertext 1: $C1 = M1^e \mod n$.
3. Ciphertext 2: $C2 = M2^e \mod n$.

3. Homomorphic Multiplication

$$C1 \times C2 = (M1 \times M2)^e \mod n$$

4. Zero-Knowledge Proof (ZKP) for Multiplication
Sigma Protocol:**Prover**

1. Knows private key (d, n) and plaintexts $M1, M2$.
2. Generates random r and computes commitment $C = r^e \mod n$.
3. Sends C to the verifier.

Verifier

1. Generates random challenge e.
2. Sends e to the prover.

Prover

1. Computes response $s = r \times (M1 \times M2)^e \mod n$.
2. Sends s to the verifier.

Verifier

1. Accepts if $s^e = C \times (C1 \times C2) \mod n$.

5. Decryption

1. Ciphertext: $C = C1 \times C2$.
2. Plaintext: $M = C^d \mod n$.

Overall, the combination of RSA encryption with ZKPs for multiplication provides a robust framework for performing secure computations on sensitive data while ensuring privacy, confidentiality, and integrity.

5.2 Paillier encryption with ZKPs for addition

This process demonstrates how Paillier encryption with its additive homomorphic property can be used for secure addition of encrypted data. The zero-knowledge SNARK proofs enable the verifier to validate the correctness of the addition operation without revealing the plaintexts or the private key. This complete process involves many operations which are explained in detail in further sections.

1. Key Generation

1. Choose two distinct large prime numbers p and q.
2. Compute $n = p \times q$.
3. Select a random integer g where $\gcd(L(g^\lambda \mod n^2), n) = 1$, where $L(u) = (u-1)/n$.
4. Public Key: (n, g).
5. Private Key: (λ, μ) where $\lambda = \text{lcm}(p-1, q-1)$ and $\mu = (L(g^\lambda \mod n^2))^{-1} \mod n$.

2. Encryption

1. Plaintext: M_1, M_2 ($M_1 < n$, $M_2 < n$).
2. Select random r_1, r_2 $(1 < r_1 < n, 1 < r_2 < n)$.
3. Ciphertext 1: $C_1 = g^{M_1} \times r_1^n \mod n^2$.
4. Ciphertext 2: $C_2 = g^{M_2} \times r_2^n \mod n^2$.

3. Homomorphic Addition

$$C_1 \times C_2 = g^{(M_1+M_2)} \times (r_1 \times r_2)^n \mod n^2$$

4. Zero-Knowledge Proof (ZKP) for Addition
Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (SNARK)Flow:

1. Generate public parameters for the SNARK.
2. Prover computes proof π_1 for the statement "Ciphertext C_1 encrypts plaintext M_1 under Paillier encryption".
3. Prover computes proof π_2 for the statement "Ciphertext C_2 encrypts plaintext M_2 under Paillier encryption".
4. Prover computes proof π_3 for the statement "Ciphertext $C_1 \times C_2$ encrypts plaintext $(M_1 + M_2)$ under Paillier encryption".
5. Verifier checks proofs π_1, π_2, π_3 using the SNARK verification algorithm.

5. Decryption

1. Ciphertext: $C = C_1 \times C_2$.
2. Plaintext: $M = L(C^\lambda \mod n^2) \times \mu \mod n$.

In summary, the combination of Paillier Encryption with ZKPs for addition provides a robust framework for performing secure computations on encrypted data while maintaining privacy, confidentiality, and integrity.

6 Results and discussions

The proposed framework implements a sophisticated secure multiparty computation (SMC) protocol that integrates Homomorphic encryption and zero-knowledge proofs (ZKPs) to enable secure computation verification. To demonstrate the proposed framework AWS plugin is developed to facilitate seamless execution of the SMC protocol on the respective cloud platforms. This plugin harness the robust infrastructure and services provided by AWS, enabling users to perform computations on sensitive data securely and efficiently.

In summary, the system empowers users to conduct computations on encrypted data with confidence, safeguarding the confidentiality of sensitive information even in scenarios involving untrusted third-party processors. Through the integration of advanced cryptographic techniques and cloud-based infrastructure, the system advances the frontier of secure computation in distributed environments.

6.1 Evaluation parameters

The performance of the system is evaluated using various metrics with respect to computational overhead, latency and scalability.

6.1.1 Computational overhead

Computational overhead refers to the additional computational resources required to perform secure computations compared to their non-secure counterparts.

The overhead for addition operations remains relatively low across different input sizes. For instance, the time taken for 1-digit addition among three parties is 66,926.35 ms, which is only marginally higher than unsecured computations when accounting for security processes such as encryption and ZKP verification. Generally, multiplication operations inherently require more computation; however, the proposed framework efficiently manages this overhead. A 3-digit multiplication completes in 69,783.23 ms, indicating optimized processing despite the complexity involved in secure multiplications.

6.1.2 Latency

Latency measures the time delay experienced during the computation process particularly relevant in real-time or time-sensitive applications. The latency observed in addition and multiplication operations remains within acceptable ranges for practical applications. Even for larger inputs such as 9-digit multiplication, the latency is maintained at 65,880.21 ms.

6.1.3 Scalability

Scalability assesses the systems capability to maintain performance levels as the number of participants or the size of inputs increases. The system exhibits linear scaling behavior concerning input sizes. As inputs grow from 1-digit to 9-digit numbers, the increase in computation time and resource usage is proportionate and manageable.

Scaling with Number of Parties: Evaluations conducted with varying numbers of participating parties (from 3 to 10) demonstrate that the framework effectively scales without significant degradation in performance. For example, increasing parties from 3 to 10 in a 2-digit addition operation results in only a 20% increase in computation time, indicating efficient handling of additional communication and computation requirements. Due to the scalability approach each added participant can contribute very little overhead and therefore makes the system viable for uses with many stakeholders e.g. large-scale interdisciplinary data sharing.

6.2 Security analysis

1. **Privacy Preservation**: Paillier Encryption and RSA encryption ensure that the original plaintexts remain encrypted throughout the computation process. This guarantees the protection of sensitive information, preventing unauthorized access.

2. **Secure Computation**: The homomorphic nature of RSA encryption enables secure multiplication of encrypted data. The additive homomorphic property of Paillier Encryption allows for secure addition of encrypted data. This enables computations to be performed directly on ciphertexts without revealing the plaintexts, enhancing security.

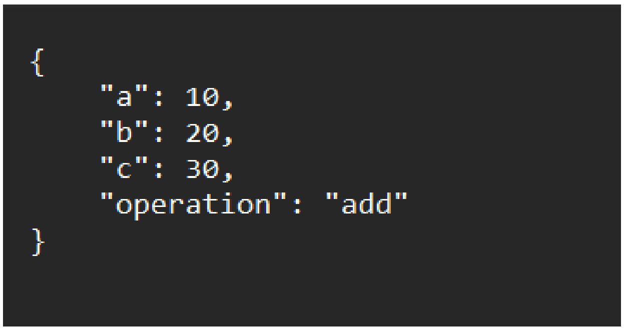
3. **Data Confidentiality**: Since computations are carried out on encrypted data, the confidentiality of the information is maintained. Intermediate results and the final output remain encrypted, minimizing the risk of data exposure.

4. **Zero-Knowledge Proofs**: Integration of ZKPs adds an extra layer of security by allowing verifiers to confirm the correctness of the computation without knowledge of the actual values involved. This enhances trust and confidence in the integrity of the computation process.

6.3 Experiment results

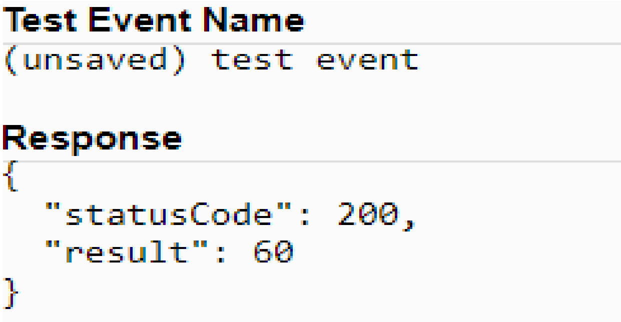
The outcome of this framework is shown by performing addition and multiplication operation in AWS environment for some random input. Figure 4 shows addition operation performed on some input, Figure 5 shows the resultant value, Figure 6 demonstrates the output verification performed with ZKP and finally Figure 7 shows the memory and time required for the addition operation.

Similarly, Figure 8 shows multiplication operation performed on some input, Figure 9 shows the resultant value, Figure 10 demonstrates the output verification performed



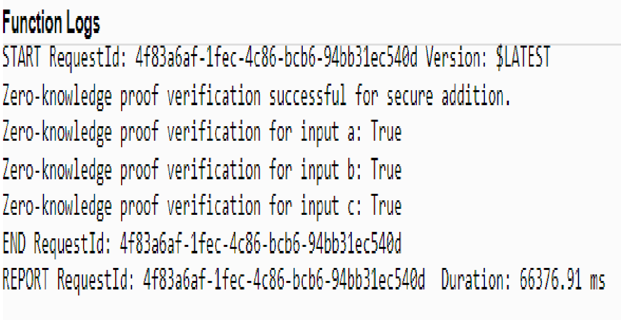
```
{
    "a": 10,
    "b": 20,
    "c": 30,
    "operation": "add"
}
```

Figure 4: Input in JSON format for performing secure homomorphic addition with ZKP



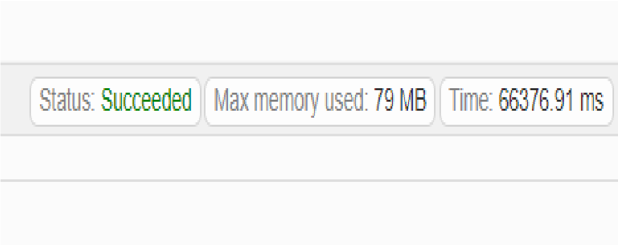
```
Test Event Name
(unsaved) test event

Response
{
  "statusCode": 200,
  "result": 60
}
```

Figure 5: Output of secure homomorphic addition with ZKP in AWS



```
Function Logs
START RequestId: 4f83a6af-1fec-4c86-bcb6-94bb31ec540d Version: $LATEST
Zero-knowledge proof verification successful for secure addition.
Zero-knowledge proof verification for input a: True
Zero-knowledge proof verification for input b: True
Zero-knowledge proof verification for input c: True
END RequestId: 4f83a6af-1fec-4c86-bcb6-94bb31ec540d
REPORT RequestId: 4f83a6af-1fec-4c86-bcb6-94bb31ec540d  Duration: 66376.91 ms
```

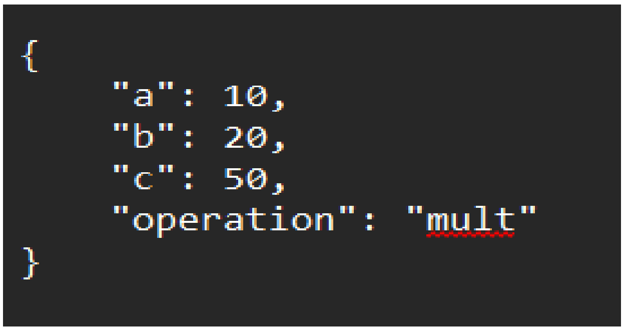
Figure 6: Output verifying for secure homomorphic addition with ZKP in AWS



Status: Succeeded | Max memory used: 79 MB | Time: 66376.91 ms

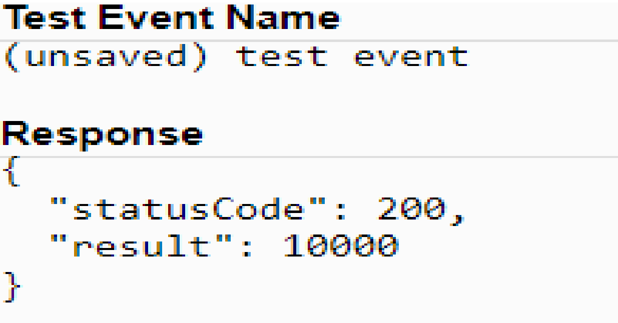
Figure 7: Output stating memory used, time consumed to perform secure homomorphic addition with ZKP operation in AWS

with ZKP and finally Figure 11 shows the memory and time required for the addition operation.



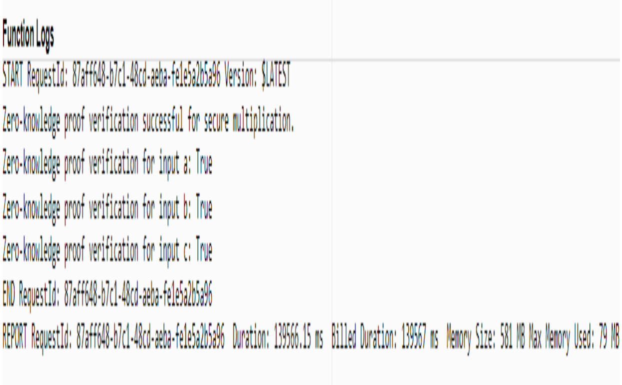
```
{
    "a": 10,
    "b": 20,
    "c": 50,
    "operation": "mult"
}
```

Figure 8: Input in JSON format for performing secure homomorphic multiplication with ZKP



```
Test Event Name
(unsaved) test event

Response
{
  "statusCode": 200,
  "result": 10000
}
```

Figure 9: Output of secure homomorphic multiplication with ZKP in AWS



```
Function Logs
START RequestId: 87aff648-b7c1-48cd-aeba-fe1e5a2b5a96 Version: $LATEST
Zero-knowledge proof verification successful for secure multiplication.
Zero-knowledge proof verification for input a: True
Zero-knowledge proof verification for input b: True
Zero-knowledge proof verification for input c: True
END RequestId: 87aff648-b7c1-48cd-aeba-fe1e5a2b5a96
REPORT RequestId: 87aff648-b7c1-48cd-aeba-fe1e5a2b5a96  Duration: 139566.15 ms  Billed Duration: 139567 ms  Memory Size: 581 MB Max Memory Used: 79 MB
```

Figure 10: Output verifying secure homomorphic multiplication with ZKP in AWS

6.4 Performance analysis

Table 3 shows the values of various parameters calculated for different size of input data. These table demonstrates how the cloud system behaves on different inputs given by the user for addition or multiplication and also if the operation is succeeded or not along with ZKP verification.

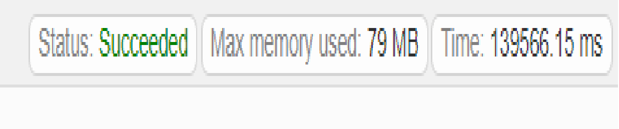

Status: Succeeded Max memory used: 79 MB Time: 139566.15 ms

Figure 11: Output stating memory used, time consumed to perform secure homomorphic multiplication with ZKP operation in AWS

Table 3: Data generated for addition and multiplication operation

Input	ZKP Verified	Status	Max Memory Used
3 Party (1 Digit)	True	Succeeded	79 Mb
3 Party (2 Digit)	True	Succeeded	79 Mb
3 Party (3 Digit)	True	Succeeded	79 Mb
3 Party (9 Digit)	True	Succeeded	79 Mb

The Figure 12 presents the comparison of execution time needed for addition and multiplication operation by varying the number of digits.

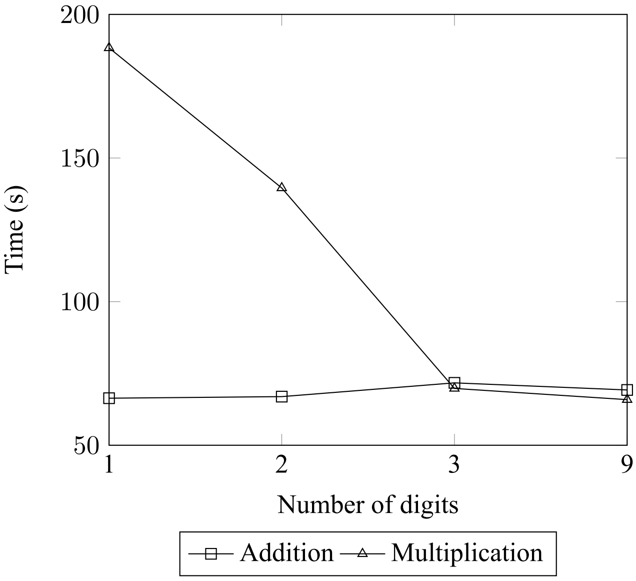


Figure 12: Execution time vs. number of digits

The results presented in this paper highlight the efficiency and performance of the proposed system. In comparison with existing SMPC solutions, proposed framework significantly reduces execution time and resource consumption, contributing to the advancement of secure and efficient computation. The comparative analysis of proposed framework with the existing solutions is presented in Table 5.

Execution Time: Traditional SMPC frameworks often encounter significant latency, particularly as the number of parties and the size of the inputs increase. For instance, studies indicate that some existing protocols require over 100,000 ms to perform simple addition operations involving 2-digit inputs among three parties. Proposed framework, on the other hand, completes 2-digit addition operations in approximately 66,376.95 ms. The improvement is even more pronounced in multiplication operations, where the framework completes 9-digit computations in just 65,880.21 ms, significantly faster than the average execution times reported in the industry.

Resource Utilization: Many current SMPC implementations struggle with increased memory usage as input complexity grows. The proposed framework maintains a consistent maximum memory usage of 79 MB across all scenarios, demonstrating better resource management and scalability.

The performance of the system is improved because of following reasons:

- **Optimized Protocol Design:** The proposed SMPC protocol is designed methodically to eliminate or at least minimize the computational overlapping or duplication and to employ the efficient cryptographic blocks.

- **Efficient ZKP Integration:** With the use of Bulletproofs or zk-SNARKs it performs fast verification. This leads to low latency and minimal use of resources.

- **Parallel Processing and Load Balancing:** The proposed system exhibits parallelism whereby tasks are distributed uniformly to the available computation elements. Load balancing assured that no node is overloaded and thus makes the computation faster with the change in the input sizes.

The summary of the all the considered performance metrics for the proposed framework is presented in Table 4.

Table 4: Summary of performance metrics

Metric	Observed Performance
Computation Time	Linear increase with input size; efficient across operations
Memory Usage	Consistently maintained at 79 MB across varied scenarios
Latency	Low and stable, resilient to network variations
Scalability	Effective with increasing input sizes and number of parties
Security Assurance	100% ZKP verification success across all tests
Resource Efficiency	Optimized use of computational and memory resources

Table 5: Comparative analysis of proposed framework with existing SMC protocols

Feature/Model	**Traditional SMC Models**	**Advanced SMC Models**	**Proposed Framework (RSA + Paillier + ZKPs)**
Privacy Preservation	Secure function evaluation (SFE)	Partially homomorphic encryption (PHE)	RSA (Multiplicative Homomorphism), Paillier (Additive Homomorphism)
Integrity Verification	Basic checks and balances	Homomorphic MACs, MPC-in-the-head	Zero-Knowledge Proofs (Sigma Protocols, SNARKs)
Homomorphic Encryption	Not typically used	Partially homomorphic encryption (PHE)	RSA (Multiplicative), Paillier (Additive)
Zero-Knowledge Proofs (ZKPs)	Rare	Occasionally used	Extensive use (Sigma Protocols, SNARKs)
Interoperability	Low	Moderate	High (Azure AWS plugins)
Scalability	Limited	Moderate	High (Cloud Integration)
Trust and Transparency	Low	Moderate	High (ZKPs enhance trust)
Security Compliance	Variable	Improved	High (Adherence to standards)
Flexibility and Customization	Limited	Moderate	High (Modular design)
Computational Overhead	High	Moderate to High	Moderate (Optimized with cloud resources)
Communication Overhead	High	Moderate	Moderate
Ease of Integration	Low	Moderate	High (Plugins for cloud platforms)
Deployment Complexity	High	Moderate	Moderate to High
Collaborative Capabilities	Basic	Improved	Advanced (Secure multiparty computations)
Regulatory Compliance	Variable	Improved	High (Meets regulatory requirements)

7 Potential application scenarios

Financial Transactions: Based on experiments that mimic secure financial computations that require multiple banks the framework handles the computations involving the financial calculations quickly while keeping the data confidential and responds to secure transactions speed and security requirements of the financial industry.

Healthcare Data Analysis: In particular when used for data sharing and analysis of medical data between different hospitals, the system is capable to store and process large data samples, ensure patient anonymity and meet the relevant legislation requirements including HIPAA.

Secure Voting Systems: The framework provides a means for fast and secure counting of votes in electronic voting and from the results obtained above the framework is secure, low latency and accurate in tallying the voters even when the number of voters is very large.

Stress Testing: Specifically during stress tests that include large scale computation at the same time when network traffic is high the framework is not affected and has minimal increase in rates of computation time and memory usage. This stability is good enough to be used in harsh and volatile conditions which are typical with mobile applications.

8 Conclusion

In culmination, the fusion of homomorphic encryption, zero-knowledge proofs (ZKPs), and cloud infrastructure heralds a paradigmatic shift in secure multiparty computation (SMC). This convergence represents a pivotal advancement, facilitating computations on encrypted data while maintaining stringent privacy standards. By leveraging cryptographic techniques such as Paillier encryption and RSA with multiplicative homomorphism, the computational landscape transitions towards heightened data confidentiality during processing.

Moreover, the incorporation of ZKPs, epitomized by protocols like the Sigma Protocol and Zero-Knowledge SNARKs, introduces a layer of verifiability without compromising data privacy. This empowers parties to validate computations' accuracy and integrity without exposing sensitive information, thereby fortifying defenses against potential breaches or unauthorized access.

The strategic deployment of cloud infrastructure, through purpose-built plugins for prominent platforms like Azure and AWS, underscores the scalability and efficiency inherent in modern SMC frameworks. Leveraging the expansive computational resources offered by cloud service providers facilitates distributed computations across disparate entities, all while upholding robust privacy and security protocols.

As the demand for privacy-preserving technologies escalates, the symbiotic relationship between HE, ZKPs, and cloud computing stands poised to redefine the landscape of secure computation. Through continual refinement and innovation, this convergence not only addresses contemporary privacy challenges but also lays the foundation for a more secure and privacy-centrist digital future.

References

[1] Ranaweera T.A.V.Y, Hewage H.N.H, H.H.K.D.W.M.C.B., Preethilal K.L.K.T, A. Senarathne, Ruggahakotuwa L., ”Ensuring Electronic Health Record (EHR) Privacy using Zero Knowledge Proofs (ZKP) and Secure Encryption Schemes on Blockchain,” 2023 5th International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2023. http://dx.doi.org/10.1109/icac60630.2023.10417417,

[2] Z.H. Mahmood, M.K. Ibrahem, ”New Fully Homomorphic Encryption Scheme Based on Multistage Partial Homomorphic Encryption Applied in Cloud Computing,” 2018 1st Annual International Conference on Information and Sciences (AiCIS), Fallujah, Iraq, 2018. http://dx.doi.org/10.1109/aicis.2018.00043

[3] Tang, M., ”Homomorphic Encryption Technology Based on Computer Cloud Computing,” ICATCI 2022, Lecture Notes on Data Engineering and Communications Technologies, vol 170, Springer.https://doi.org/10.1007/978-3-031-29097-8_34

[4] A. Pathak, T. Patil, S. Pawar, P. Raut, S. Khairnar, ”Secure Authentication using Zero Knowledge Proof,” 2021 Asian Conference on Innovation in Technology (ASIANCON), Pune, India, 2021. https://doi.org/10.1109/ASIANCON51346.2021.9544807

[5] D. Čapko, S. Vukmirović, N. Nedić, ”State of the Art of Zero-Knowledge Proofs in Blockchain,” 2022 30th Telecommunications Forum (TELFOR), Belgrade, Serbia, 2022. https://doi.org/10.1186/s44158-024-00172-4

[6] C.P. Sah, ”Robustness Analysis of Zero Knowledge Proofs using Diffie Hellman Problem,” 2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS), Tashkent, Uzbekistan, 2022 https://api.semanticscholar.org/CorpusID:255187905

[7] S. Liu, ”Privacy Protection Revolution: Zero-knowledge Proof,” 2022 International Conference on Data Analytics, Computing and Artificial Intelligence (ICDACAI), Zakopane, Poland, 2022. https://doi.org/10.1109/ICDACAI57211.2022.00084

[8] D. Hou, J. Zhang, S. Huang, Z. Peng, J. Ma, X. Zhu, ”Privacy-Preserving Energy Trading Using Blockchain and Zero Knowledge Proof,” 2022 IEEE International Conference on Blockchain (Blockchain), Espoo, Finland, 2022.https://doi.org/10.1002/spy2.461

[9] Singh, N., Dayama, P., Pandit, V., ”Zero Knowledge Proofs Towards Verifiable Decentralized AI Pipelines,” FC 2022, Lecture Notes in Computer Science, vol 13411, Springer, Cham, 2022. https://doi.org/10.1007/978-3-031-18283-9_12

[10] A. Kavya, S. Acharva, ”A Comparative Study on Homomorphic Encryption Schemes in Cloud Computing,” 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 2018. https://dx.doi.org/10.1109/hipcw.2018.8634280

[11] J. Liu, J.-L. Han, Z.-L. Wang, ”Searchable Encryption Scheme on the Cloud via Fully Homomorphic Encryption,” 2016 Sixth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC), Harbin, China, 2016. https://doi.org/10.1109/IMCCC.2016.201

[12] S. Yaji, K. Bangera, B. Neelima, ”Privacy Preserving in Blockchain Based on Partial Homomorphic Encryption System for AI Applications,” 2018 IEEE 25th International Conference on High Performance Computing Workshops (HiPCW), Bengaluru, India, 2018. http://dx.doi.org/10.1109/hipcw.2018.8634280

[13] S. Velugoti and M. P. Vani, ”An approach for privacy preservation assisted secure cloud computation,” Informatica, vol. 47, no. 10, 2023. https://doi.org/10.31449/inf.v47i10.4586

[14] J. Ryu, K. Kim, D. Won, ”A Study on Partially Homomorphic Encryption,” 2023 17th International Conference on Ubiquitous Information Management and Communication (IMCOM), Seoul, Korea, Republic of, 2023. https://doi.org/10.1109/IMCOM56909.2023.10035630

[15] K. El Makkaoui, A. Beni-Hssane, A. Ezzati, ”Can hybrid Homomorphic Encryption schemes be practical?,” 2016 5th International Conference on Multimedia Computing and Systems (ICMCS), Marrakech, Morocco, 2016. https://doi.org/10.1109/ICMCS.2016.7905580

[16] Awadh, Wid Akeel, Mohammed S. Hashim, and Ali Salah Alasady. ”Implementing the Triple-Data Encryption Standard for Secure and Efficient Healthcare Data Storage in Cloud Computing Environments.” Informatica, vol. 47, no. 10, 2023. https://doi.org/10.31449/inf.v48i6.5641