Improved Delegation Of Computation Using Somewhat Homomorphic Encryption To Reduce Storage Space

Dhivya.S (PG Scholar)
M.E Computer Science and Engineering
Institute of Road and Transport Technology
Erode, TamilNadu
dhivyashanthi10@gmail.com

Venkatachalam.G (Associate Professor)
M.E Computer Science and Engineering
Institute of Road and Transport Technology
Erode, TamilNadu

Abstract-- Fully Homomorphic Encryption scheme is far from being practical because of its large computational cost and large cipher texts. Since then, considerable efforts have been made to devise more efficient schemes. However, most Fully Homomorphic Encryption schemes still have very large cipher texts. This presents a considerable bottleneck in practical deployments. It becomes more important to protect the data from misuse by insiders or hacking by outsiders. To reduce the risk, the data may be encrypted prior to storage. Under this scenario, we give an efficient storage solution using a hybrid scheme. Our hybrid scheme can be used to protect the privacy, i.e., the computations of Public Key Encryption-encrypted data are outsourced, along with Hybrid to a cloud that has huge computing power and storage. The cloud performs the outsourced computations, and returns the resulting cipher text encrypted under Somewhat Homomorphic Encryption.

I. INTRODUCTION

A. Motivation of the Project

The concept of computation on encrypted data without decryption was first introduced by Rivest, Adleman and Dertouzos in 1978. Thirty years later, Gentry proposed a fully homomorphic encryption (FHE) based on ideal lattices.

This scheme is far from being practical because of its large computational cost and large ciphertexts. Since then, considerable efforts have been made to devise more efficient schemes. However, most FHE schemes still have very large ciphertexts (millions of bits for a single ciphertext). This presents a considerable bottleneck in practical deployments. The following situations are considered: several users upload data encrypted with a public-key FHE, a server carries out computations on the encrypted data and then sends them to an agency who has a decryption key for the FHE.This is common in typical FHE scenarios, such as medical and financial applications. In this situation, one approach to reduce the storage requirement is to use AES encryption to encrypt data, and then perform homomorphic computations on ciphertexts after converting to FHE-ciphertexts. This method has a great advantage in storage and communication, because only small AES-ciphertexts are transmitted from user to server, and these are homomorphically decrypted only when their homomorphic computations are required.

In this paper, an alternative method that encrypts messages with a public key encryption (PKE) and converts them into SHE-ciphertexts for homomorphic computations. In this approach, the ciphertext expansion ratio is only two or three regardless of the message size.

B. Network Security

Network security consists of the provisions and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources. Network security involves the authorization of access to data in a network, which is controlled by the network administrator. Users choose or are assigned an ID and password or other authenticating information that allows them access to information and programs within their authority. It secures the network, as well as protecting and overseeing operations being done. The most common and simple way of protecting a network resource is by assigning it a unique name and a corresponding password.

C. Fundamentals of Cryptography

Encryption schemes are broadly of two types: symmetric and asymmetric encryption schemes.

Symmetric encryption schemes: This implies that in order to communicate with different persons, different key for each person is used. Requirement of a large number of keys in these schemes makes their key generation and management relatively more complex operations. However, symmetric schemes present the advantage of being very fast and they are used in applications where speed of execution is a paramount requirement. Among the existing symmetric encryption systems, AES, One-Time Pad and Snow are very popular.

Asymmetric encryption schemes: In these schemes, every participant has a pair of keys private and public. While the private key of a person is known to only her, the public key of each participant is known to everyone in the group. Such schemes are more secure than their symmetric counterparts and they don’t need any prior agreement between the communicating parties on a common key before establishing a session of communication. RSA and ElGamal are two most popular asymmetric encryption systems.

D. Applications of Homomorphisms

Protection of mobile agents: One of the most interesting applications of homomorphic encryption is its use in protection of mobile agents. Since all conventional computer architectures are based on binary strings and only require multiplication and addition, such homomorphic cryptosystems would offer the possibility to encrypt a whole program so that it is still executable.

The protection of mobile agents by homomorphic encryption can be used in two ways:

(i) Computing with encrypted functions and

(ii) Computing with encrypted data.

Computation with encrypted functions is a special case of protection of mobile agents. In such scenarios, a secret function is publicly evaluated in such a way that the function remains secret. Using homomorphic cryptosystems, the encrypted function can be evaluated which guarantees its privacy. Homomorphic schemes also work on encrypted data to compute publicly while maintaining the privacy of the secret data. This can be done encrypting the data in advance and then exploiting the homomorphic property to compete with encrypted data.

Secret sharing scheme: In secret sharing schemes, parties share a secret so that no individual party can reconstruct the secret form the information available to it. However, if some parties cooperate with each other, they may be able to reconstruct the secret. In this scenario, the homomorphic property implies that the composition of the shares of the secret is equivalent to the shares of the composition of the secrets.

Election schemes: In election schemes, the homomorphic property provides a tool to obtain the tally given the encrypted votes without decrypting the individual votes.

Watermarking and fingerprinting schemes: Digital watermarking and fingerprinting schemes embed additional information into digital data. The homomorphic property is used to add a mark to previously encrypted data. In general, watermarks are used to identify the owner/seller of digital goods to ensure the copyright. In fingerprinting schemes, the person who buys the data should be identified by the merchant to ensure that data is not illegally redistributed.

Zero-knowledge proofs: This is a fundamental primitive of cryptographic protocols and serves as an example of a theoretical application of homomorphic cryptosystems. Zero-knowledge proofs are used to prove knowledge of any private information. For instance, consider the case where a user has to prove his identity to a host by logging in with her account and private password.

E. Project Description

In this work explore an alternative method that encrypts messages with a public key encryption (PKE) and converts them into SHE-ciphertexts for homomorphic computations. In this approach, the ciphertext expansion ratio is only two or three regardless of the message size. Moreover, the decryption circuit is very shallow when the SHE allows large integers as messages. For example, the decryption circuit of ElGamal over Z*N* has a multiplicative depth of nine under a SHE with the message space Z*N*.2 The depth can be further reduced by representing the secret exponent *e* as log*w e* binary vectors of length *w*, which is an improvement over the Gentry-Halevi technique. When using additive (resp. Multiplier) homomorphic encryption as the underlying PKEs, the additional advantage that additions (resp. Multiplications) can be computed without converting to SHE is obtained.

F. Objective

To reduce the storage space in the cloud environment this approach is used. In this work includes small bandwidth, reduced storage requirements, and computational efficiency.

II. EXISTING SYSTEM

Several users upload data encrypted with a public-key FHE, a server carries out computations on the encrypted data and then sends them to an agency who has a decryption key for the FHE. This is common in typical FHE scenarios, such as medical and financial applications. In this situation, one approach to reduce the storage requirement is to use AES encryption to encrypt data, and then perform homomorphic computations on ciphertexts after converting to FHE-ciphertexts. This method has a great advantage in storage and communication, because only small AES-ciphertexts are transmitted from user to server, and these are homomorphically decrypted only when their homomorphic computations are required. However, this approach is not practical when the amount of messages transmitted simultaneously is small compared with the size of an FHE ciphertext. Moreover, the conversion on AES-ciphertexts into FHE-ciphertexts requires a leveled FHE with a multiplicative depth of at least forty.

Disadvantages:

- It's not efficient in using cloud environment.
- Take a large space in the storage requirement.
- Very slow process.

III. PROPOSED SYSTEM:

In this work explore an alternative method that encrypts messages with a public key encryption (PKE) and converts them into SHE-ciphertexts for homomorphic computations. In this approach, the ciphertext expansion ratio is only two or three regardless of the message size. Moreover, the decryption circuit is very shallow when the SHE allows large integers as messages. For example, the decryption circuit of ElGamal over Z*N* has a multiplicative depth of nine under a SHE with the message space Z*N*.2 The depth can be reduced further by representing the secret exponent *e* as log*w e* binary vectors of length *w*, which is an improvement over the Gentry-Halevi technique. When using additive (resp. Multiplier) homomorphic encryption as the underlying PKEs, the additional advantage that additions (resp. Multiplications) can be computed without converting to SHE is obtained.

Advantages:

- It covert very large data as small message.
- This technique very useful in cloud and large data storage environment.
- High speed in data retrieval.

IV. SYSTEM DESIGN AND IMPLEMENTATION

A. Architecture:

Systems design is simply the design of systems. It implies a systematic and rigorous approach to design an approach demanded by the scale and complexity of many system problems. A systems approach to design is entirely compatible with a user-centered approach. Indeed, the core of both approaches is understanding user goals. A systems approach looks at users in relation to a context and in terms of their interaction with devices, with each other, and with themselves. A systems approach to design is most appropriate for projects involving large systems or systems of systems. Such projects typically involve many people, from many disciplines, working together over an extended period of time. They need tools to cope with their project’s complexity: to define goals, facilitate communications, and manage processes. Solo designers working on small projects may find the same tools a bit cumbersome for their needs.

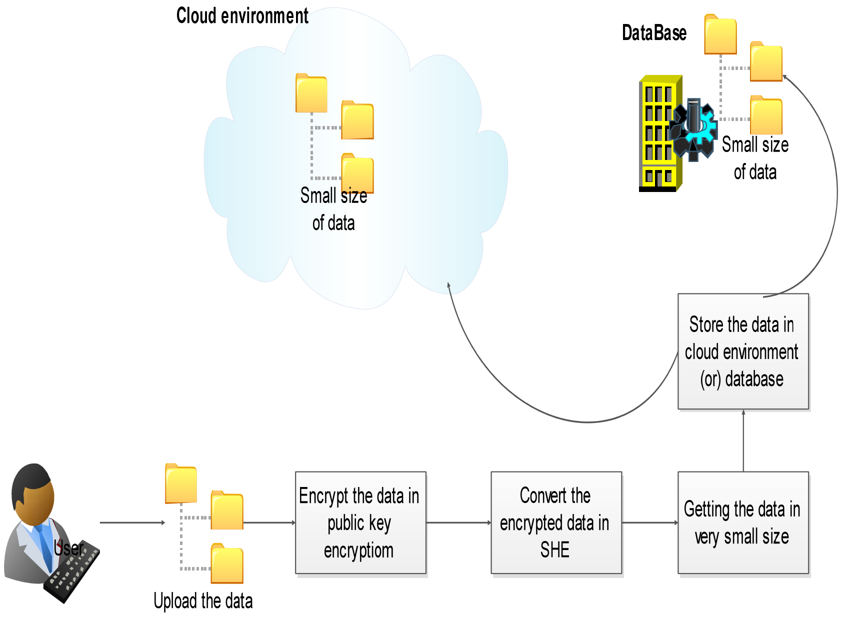


Fig 1. System Architecture

B. Module Description:

Modules:

1. User Login and Data Upload
2. User Data convert to Public Key Encryption (PKE)
3. PKE data convert to Somewhat Homomorphic Encryption (SHE)
4. Upload the file to Storage Environment

B.1. *User Login and Data Upload*

In this module each user registers their information and login to the process. The registered user information can be stored in the app engine database. After that User uploads their file wants to store and file size of the encryption process to reduce the storage requirements.

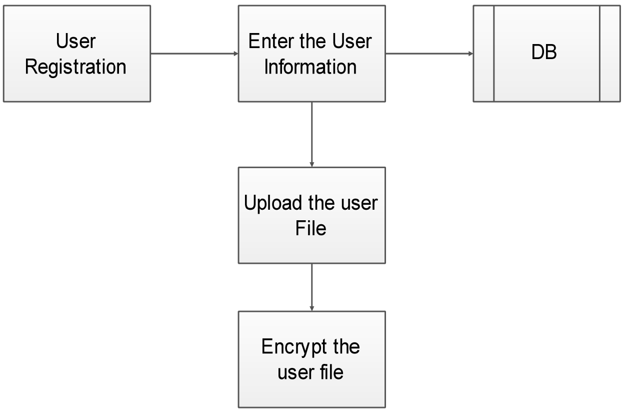


Fig 2. User Login and Data Upload

B.2. *User Data convert to Public Key Encryption (PKE)*

In this module user's data are converted to an encrypted file by using public key encryption. The public key encryption has been done through AES algorithm. First thing is generation of public keys. The Key generation is basically done by choosing random prime numbers.

Then the Process of AES algorithm the key value and user's data are converted into cipher text form. This encrypted file will be used for further processing. The encrypted file is large in size because of the key values.

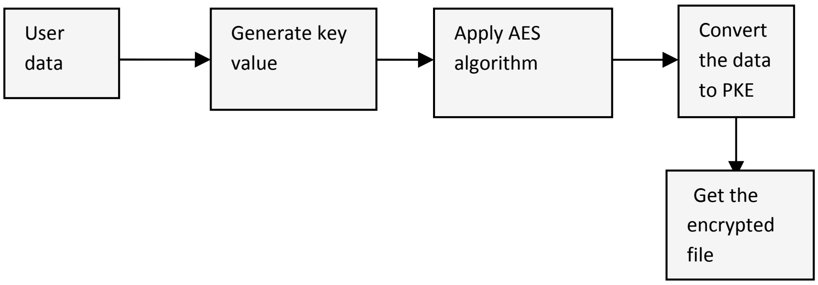


Fig 3. User Data convert to PKE

B.3. *PKE Convert to Somewhat Homomorphic Encryption* (SHE):

In this module the Public key encryption, data is converted into somewhat homomorphic encrypted data using ElGamal Algorithm. The SHE (SomeWhat Homomorphic Encryption) key has been generated. This key value and the PKE encrypted file are converted to somewhat homomorphic encryption . Finally the SHE ciphertext is received. This process reduces the bit rate of file size.

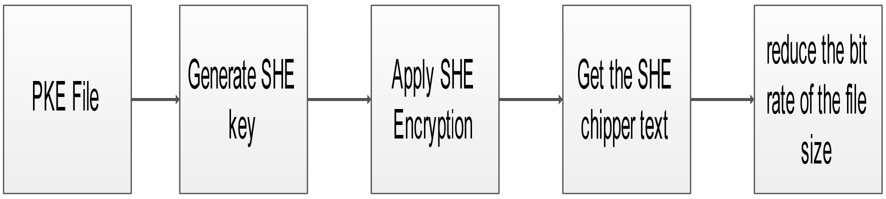


Fig 4. PKE to SHE

B.4. *Upload File to Storage Environment*:

In this module user upload their chipper text file to the storage environment. The user has the very small size of the chipper text file and which environment they want to store the data. If their retrieve the data to see the original plaintext. Enter the correct key to decrypt the data.

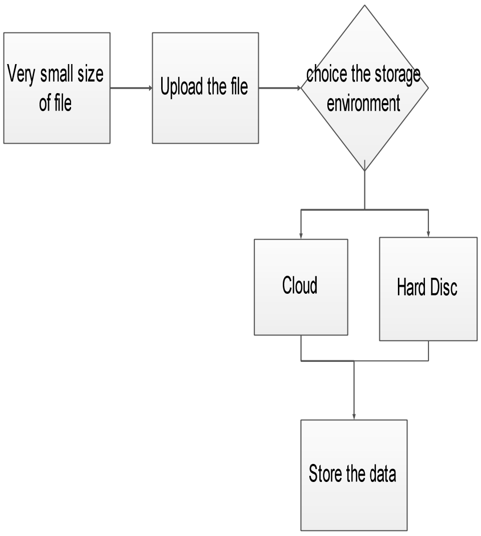
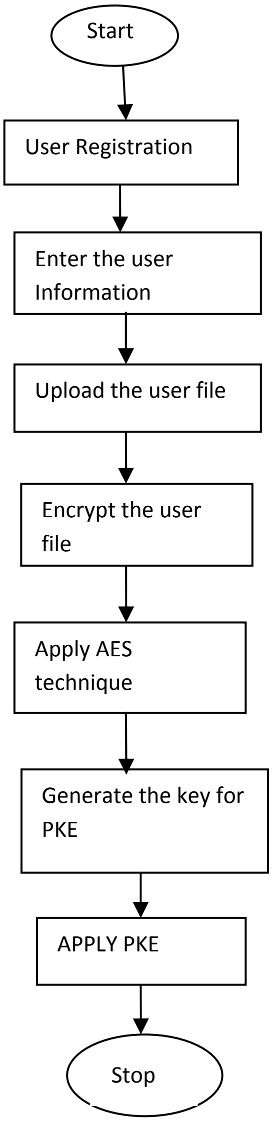


Fig 5. Upload to Storage

5.3 DATA FLOW DIAGRAM

Level 1:



Level 2:

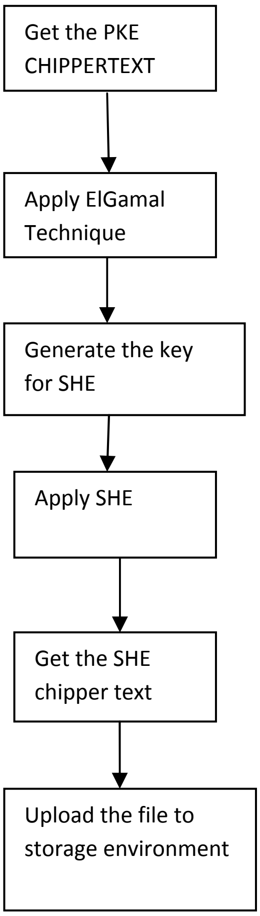


Fig 6. Data Flow Diagram(Level 2)

V. CONCLUSION AND FUTURE ENHANCEMENT

A. C*onclusion*:

A hybrid scheme that combines public key encryption and somewhat homomorphic encryption is proposed. The proposed scheme is suitable for cloud computing environments since it has small bandwidth, low storage requirement, and supports efficient computing on encrypted data. The solution provides a trade-off between the size of the transmitted ciphertexts and the conversion costs. While the ciphertext expansion of PKE is larger than that of AES, it can be homomorphically evaluated with a SHE of much smaller multiplicative depth. The parameters of the hybrid scheme are very large when the message space of the underlying FHE is ZN. For an efficient implementation, a method to evaluate mod N arithmetic using an FHE whose message space is ZM for small $M > 2$ is needed.

B. Future Enhancement:

A hybrid scheme that combines public key encryption and somewhat homomorphic encryption is proposed. The proposed scheme is suitable for cloud computing environments since it has small bandwidth, low storage requirement, and supports efficient computing on encrypted data. But the decryption of the cipher text data retrieval time too long. In future the technique for quickly decrypt the SHE encrypted data is enhanced.

REFERENCES

[1] R. Barbulescu, P. Gaudry, A. Joux, and E. Thomé. (2013). “A quasipolynomial algorithm for discrete logarithm infinite fields of small characteristic.” [Online]. Available: http://eprint.iacr.org/2013/400

[2] J. H. Cheon *et al.*, “Batch fully homomorphic encryption over the integers,” in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 7881, T. Johansson and P. Nguyen, Eds. Berlin, Germany: Springer-Verlag, 2013, pp. 315–335.

[3] K.-M. Chung, Y. Kalai, and S. Vadhan, “Improved delegation of computation using fully homomorphic encryption,” in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 6223, T. Rabin, Ed. Berlin, Germany: Springer-Verlag, 2010, pp. 483–501.

[4] J.-S. Coron, T. Lepoint, and M. Tibouchi, “Scale-invariant fully homomorphic encryption over the integers,” in *Public-Key Cryptography* (Lecture Notes in Computer Science), H. Krawczyk, Ed. Berlin, Germany: Springer-Verlag, 2014, pp. 311–328.

[5] T. ElGamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 196, G. R. Blakley and D. Chaum, Eds. Berlin, Germany: Springer-Verlag, 1984, pp. 10–18.

[6] J. Fan and F. Vercauteren, “Somewhat practical, fully homomorphic encryption,” in *Proc. IACR Cryptol.*, 2012, p. 144.

[7] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in *Proc. 41st Annu. ACM Symp. Theory Comput. (STOC)*, 2009, pp. 169–178.

[8] C. Gentry and S. Halevi, “Fully homomorphic encryption without squashing using depth-3 arithmetic circuits,” in *Proc. IEEE 52nd Annu. Symp. Found. Comput. Sci. (FOCS)*, Oct. 2011, pp. 107–109.

[9] C. Gentry, S. Halevi, and N. P. Smart, “Homomorphic evaluation of the AES circuit,” in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 7417, R. Safavi-Naini and R. Canetti, Eds. Berlin, Germany: Springer-Verlag, 2012, pp. 850–867.

[10] R. L. Rivest, L. Adleman, and M. L. Dertouzos, “On data banks and privacy homomorphisms,” *Found. Secure Comput.*, vol. 4, no. 11, pp. 169–180, 1978.