Additively LWE Based Homomorphic Encryption for Compact Devices with Enhanced Security

Ratnakumari Challa[1] and Gunta Vijaya Kumari[2]
(Corresponding author: Ratnakumari Challa)
Department of Computer Science, Engineering, Rajiv Gandhi University of Knowledge Technologies[1]
IIIT-AP, RKValley, Kadapa Dist, Andhra Pradesh, India
Department of Computer Science, Engineering, Jawaharlal Nehru Technological University, Hyderabad[2]
Kukatpally, Hyderabad, Telangana, India
(Email: ratnamala3784@gmail.com)
(Received Nov. 03, 2017; Revised and Accepted July 7, 2018; First Online Dec. 10, 2018)

Abstract

LWE based homomorphic encryption scheme has been proven as secured technique and consequently widely implemented since the last decade. However, the scheme is not considered to be practical because of its larger size of public keys and ciphertexts. In this paper, LWE based additively homomorphic encryption technique is further modified to enhance the performance in minimizing the space required for public keys and ciphertext without compromising the security of the scheme. This makes the scheme suitable for implementation in low space devices. The practical implementation of the scheme is explored and its performance analysis has been presented here. The scheme is also compared with the standard LWE.

Keywords: Compact Devices; Learning with Errors; Prime Factoring; Pseudorandom Generator

1 Introduction

A Fully Homomorphic Encryption (FHE) is treated as the*holy grail* of cryptography, because, it allows arbitrary processing of data in encrypted mode [8, 13]. An encryption scheme is called homomorphic if, given two ciphertexts say $c_1 = E_k(m_1)$ and $c_2 = E_k(m_2)$ where m_1, m_2 are plaintexts and k is the key, one can compute $c = c_1 o c_2 = E_k(m_1 o m_2)$ for some operation o such that $D_k(c) = m_1 o m_2$ [11]. This idea of homomorphic encryption was initially proposed by Rivert, et al [19] in 1978. If o corresponds to a single operation of either addition or multiplication only, the scheme is said to be partially homomorphic. Several partially homomorphic encryption schemes were proposed and successfully used in applications such as electronic voting, private information retrieval, multiparty computation, oblivious polynomial evaluation and so on [11, 15]. However, to perform arbitrary computations over the encrypted data so that the scheme is suitable for any application in general, it must support both addition and multiplication operations over the ciphertexts without any limits in case of which it is called as fully homomorphic encryption (FHE) [8]. Solving a three-decade old long lasting cryptographic problem of designing an FHE scheme was a dream of cryptographers, which was first theoretically solved in a pioneering work by Craig Gentry in 2009 using an innovative construction method [8] Gentry's FHE is based on the algebraic lattice theory and consists a general blueprint that can be used for the construction of FHE schemes. However, the scheme was practically infeasible due to high computational complexities underlying the blueprint, specifically, the bootstrapping or ciphertext refreshing process. In a quest for devising a practical FHE scheme, several variants of the Gentry's scheme were explored [20–22]. Many new schemes based on different security assumptions and hard algebraic and number theoretic problems such as Approximate Greatest Common Divisors (AGCD) [22], Chinese Remainder Theorem (CRT) [5], identity based [13] and attribute based schemes [10] were proposed. Though all these works have shown progressive improvements one over the other, none of them could be a candidate for practical deployment. Therefore, devising an FHE scheme with practical time complexities is still an open problem.

The Learning with errors (LWE) based cryptographic scheme was first proposed and implemented by Regev [18] in 2009. LWE problem has been considered well suited for new research on public key cryptography. LWE based cryptosystem is proven as simple and fast for implementation. Moreover, the security of the LWE problems is proven to be hard [17] since it is related to the well-known "learning parity with noise" problem.

Several variants of LWE based Homomorphic encryption schemes such as FHE using standard LWE [2, 3], RLWE (Ring LWE) [2, 14], and other variants [1, 4, 9]

have been proposed. The theoretical implementation of the schemes has been considered to preserve privacy in cloud computing while practical implementation of LWE based homomorphic encryption is considered to be complex due to its larger public keys and ciphertexts. The space constraint also limits the implementation of these schemes on compact devices. However, there were some techniques [6, 7, 12, 16] proposed elsewhere to reduce the size of public keys and ciphertexts which are suitable for the implementation in low speed and low storage devices. The aim of this work is to propose the possible implementation of the LWE based cryptosystem suited for the devices with low storage capacity.

Contributions. The major contribution of this paper is to reduce the size of the public keys and ciphertexts. The scheme is theoretically presented with time complexities for the Encryption, Decryption, Key Generation and Additions functions in the previous work [16]. In this work, the extended version of the encryption is presented to further minimize the public key space without compromising the security.

2 Preliminaries

2.1 Notations

In this section, basic concepts related to LWE notations are presented for quick appreciation of the proposed work. An integer is denoted as small case letter in single quotations (*e.g.*, $'n'$). The bold upper case letters (*e.g.*, $\boldsymbol{V}$)) are used to denote vectors. The symbols $'+'$ and $'.'$ are used for the addition and multiplication operations respectively. The symbol $< \boldsymbol{V}_1, \boldsymbol{V}_2 >$ denotes the integer which resulted as the sum of individual products of elements of the vectors $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$.

2.2 LWE Based Homomorphic Encryption

Standard LWE based homomorphic encryption scheme [3] is constructed based on two major parameters: $'n'$ (dimension) and $'q'$ (modulus). First Secret vector $\boldsymbol{S}$ of $'n'$ integers is chosen and then set of public keys $(\boldsymbol{A}_i, b)$, denoted as $PK = \{PK_1, PK_2...\}$, is computed using key generation function, where $\boldsymbol{A}_i$ is an arbitrary vector of $'n'$ integers, and $'b'$ is an integer computed as $< \boldsymbol{A}_i, \mathbf{S} > +2.e_i$; where $'e_i'$ is small randomly chosen error. Now, for every j^{th} public key generation, the random arbitrary vector $\boldsymbol{A}_j$ is chosen, and j^{th} public key, PK_j, computed from the secret key $\mathbf{S}$ as follows: $PK_j = (\boldsymbol{A}_j, b_j) = (\boldsymbol{A}_j, < \boldsymbol{A}_j, \boldsymbol{S} > +2.e_j)$.

Given the plaintext message bit m_i, encryption algorithm computes the ciphertext $\boldsymbol{C}_i = (\boldsymbol{A}_i, v_i)$ using any random j^{th} public key $PK_j = (\boldsymbol{A}_j, b_j)$ where $v_i = b_j + m_i \pmod q = (< \boldsymbol{A}_j, \boldsymbol{S} > +2.e_j + m_i \pmod q)$ and $\boldsymbol{A}_i = \boldsymbol{A}_j$.

Decryption takes ciphertext $\boldsymbol{C}_i = (\boldsymbol{A}_i, v_i)$ and computes the plaintext message bit $'m_i'$ using the secret key $\boldsymbol{S}$. The decryption process is given as follows:

$$m_i = (v_i - < \boldsymbol{A}_i, \boldsymbol{S} >) mod 2,$$

Decryption eliminates two masks and leaves the message bit as output.

3 Scheme with Shorter Public Keys and Ciphertexts

In this section, we formally present the LWE based additively homomorphic encryption scheme with shorter public keys and ciphertexts. Seed based technique is proposed to minimize the each public key as well as ciphertext from $(n+1).log_2(q)$ bits down to $2.log_2(q)$ bits [7, 16]. A pseudo random number generator with initialized seed value $(seed_j)$ is used to generate the vector $\boldsymbol{A}_j$ of 'n' elements which in turn generates the j^{th} public key $PK_j = (\boldsymbol{A}_j, b_j)$.

Instead of publishing the public key $PK_j = (\boldsymbol{A}_j, b_j)$ of size $n+1$, the public key vector is shortened to two integers and published as $(seed_j, b_j)$. LWE based encryption with shortened public keys with the support of homomorphic encryption [16] is formally presented as follows:

KeyGen function. It takes initial parameters, modulus $'q'$ and dimension $'n'$ as inputs, then it chooses the secret key vector $\boldsymbol{S}$ of $'n'$ integers and generates set of public keys $PK(= \{PK_1, PK_2, ...\})$. Any i^{th} element of PK, using parameters $'n', 'q'$ and the secret key vector $\boldsymbol{S}$, is computed as follows:

1) Choose a prime value $'p_i'$ as a seed value and pass it to the pseudo random number generator for generating $'n'$ integers of vector $\boldsymbol{A}_i$.
2) Compute the public key $PK_i = (\boldsymbol{A}_i, b_i)$ as $(\boldsymbol{A}_i, < \boldsymbol{A}_i, \boldsymbol{S} > +2e_i)$ where $'e_i'$ is a small random error.
3) Publish the public key $PK_i = (p_i, b_i)$ instead of $(\boldsymbol{A}_i, b_i)$.

Encryption function. For encrypting any i^{th} plain text message bit m_i (0 or 1)

1) Choose any k^{th} public key $PK_k : (p_k, b_k)$
2) Compute ciphertext $\boldsymbol{C}_i$ using public key PK_k as follows:

$$\boldsymbol{C}_i = (p_i, v_i) = (p_k, b_k + m_j).$$

Homomorphic addition function. Given any two ciphertexts $\boldsymbol{C}_x = (p_x, v_x)$ and $\boldsymbol{C}_y = (p_y, v_y)$, compute the new sum cipher $\boldsymbol{C}_z$ as follows:

$$\boldsymbol{C}_z = (p_z, v_z),$$

where $p_z = p_x.p_y$ and $v_z = v_x + v_y$.

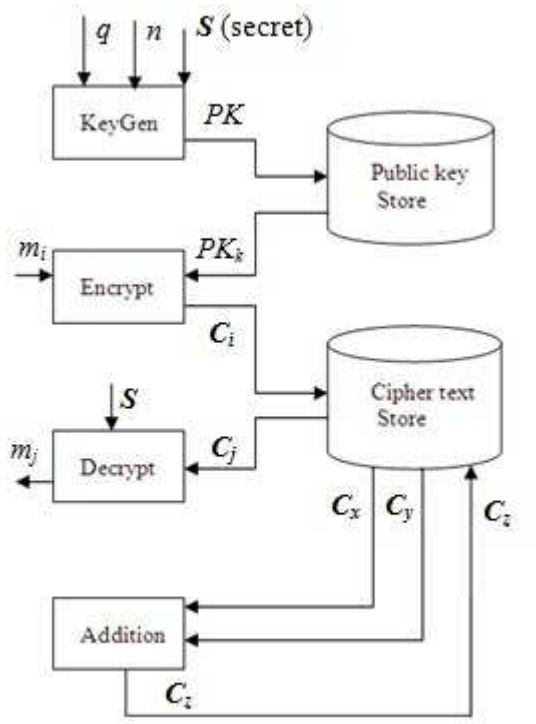

Figure 1: Proposed LWE based additively homomorphic encryption process

Decryption function. For any given ciphertext $\boldsymbol{C}_j = (p_j, v_j)$ decryption function computes the plaintext message bit m_j using the secret key $\boldsymbol{S}$. The process is given as follows:

1) Compute set of prime factors p_1, p_2, p_3 and so on from p_j using prime factoring technique.
2) Generate vector $\boldsymbol{A}_i$ of size $'n'$ using pseudo random number generator function from the seed value $'p'_i$ for all prime factors computed from $'p'_j$.
3) Compute sum vector $\boldsymbol{A}$ from all vectors $\boldsymbol{A}_1$, $\boldsymbol{A}_2$, $\boldsymbol{A}_3$,... as follows:
$$A = \sum_i A_i$$
4) Compute message m_j from ciphertext using the following relation,
$$m_j = (v_j - < \boldsymbol{A}, \boldsymbol{S} >) \bmod 2$$
where $\boldsymbol{S}$ is the secret key vector.

The whole process of key generation, encryption, and decryption and addition operation is illustrated as shown in the Figure 1.

4 An Extended Encryption

The key generation function is used to generate a set of $'x'$ public keys and these keys are stored in the public store.

In the encryption process, a public key, $PK_k = (p_k, b_k)$ is chosen randomly from the public key store to encrypt the plain text message bit m_i and to compute the ciphertext $\boldsymbol{C}_i = (p_i, v_i)$ as described under Encryption Function in Section 3. The ciphertext thus computed is stored in publicly accessed ciphertext store. Also, the second component $'v'_i$ of the ciphertext is computed by adding the message bit (0 or 1) to the second component $'b'_k$ of the public key, PK_k. If the message is 0, then $v_i = b_k$ and if the message is 1, then $v_i = b_k + 1$. This makes very little difference to the component $'v'_i$ of the ciphertext. Since the public key store and the ciphertext store are publicly accessible, an adversary can choose any ciphertext from the ciphertext store and search for its corresponding public key (used for its encryption) in the public key set. If the size of public key set is small, then it becomes easy for an adversary to search for its corresponding public key, hence, to compute the message bit. Therefore, big size of the public key set ensures that the scheme is secure. However, the huge size may affect the suitability of scheme in implementing over compact devices.

Modification in the encryption function, proposed earlier [16], helps to reduce the number of keys down to small number and to make the scheme suitable for compact devices without compromising the security. The proposed extended version of the scheme utilizes the modified encryption function.

Modified encryption function. For encrypting any i^{th} message bit m_i (0 or 1)

1) Choose any k^{th} public key $PK_k : (p_k, b_k)$
2) Compute ciphertext $\boldsymbol{C}_i$
$$\boldsymbol{C}_i = (p_i, v_i) = (p_k, b_k + m_i + 2.e_i)$$
where $'e'_i$ is randomly chosen error.

Now, the newly added error term 'e'_i in the encryption seems to make the scheme significantly more secure; further, the public key set can be minimized to fit for the low storage devices.

Security of the scheme.

- The security of the scheme is totally dependent on modulus $'q'$, dimension $'n'$ and the secret vector $\boldsymbol{S}$. Hence, its hardness is equivalent to that of the LWE problem.
- It is important to consider the privacy of the operations on the ciphertext as it is important for homomorphic encryption scheme. The size of the ciphertext is reduced to two and it maintains the privacy even after performing the many addition operations on it.

5 Results and Discussion

In the proposed scheme, the total size required for storing $'x'$ public keys or ciphertexts is minimized to $[2x.\log_2(q)]$,

whereas the size required in standard LWE schemes is $[x.(n+1).\log 2(q)]$. The comparison of the storage (in bits) required for publishing 1024 public keys in standard LWE scheme and the proposed scheme is given in Table 1. For the parameter $n = 10$ and for an integer $'q'$ with 10 bits the maximum storage required for 1024 public keys in the proposed scheme is 24477 bits, whereas for the standard LWE scheme, the storage required is 112624 bits. A key implication of this minimization is that the size of the public keys or ciphertexts becomes independent of the security parameter $'n'$.

In the practical implementation of the proposed LWE based additively homomorphic encryption scheme, it is observed that the execution time for the encryption operation is in the same range as that of the previously proposed work [16]. The comparison of the time complexities of standard LWE scheme [3] and the proposed version of the LWE scheme for different values of $'n'$ and $'q'$ are given in Figure 2. It is observed that the execution time for the homomorphic encryption functions of the proposed LWE are comparable to the standard LWE. The execution time for the Key Generation, Encryption and Decryption operations are in the same range too as that of the standard LWE. However, the execution time for the addition operation is significantly low in the proposed LWE which becomes more prominent at higher $'n'$ and $'q'$ values. At $n = 100000$ and $'q'$ to be a 50 bit number, the execution time for the addition operation in standard LWE is 2.1×10^8 whereas for the proposed LWE, it is 4×10^3.

6 Conclusions

An efficient LWE based additively Homomorphic Encryption has been proposed and explored with practical implementation. The prime number used as the seed value for the pseudo random generator helps shorten the public keys and ciphertexts. The proposed enhanced encryption function provides the enhanced security and further minimization of public key space large extent. In the practical implementation of the scheme, the execution time complexities for every function are reasonably small even for higher values of the security parameters. This makes the scheme suitable to implement over compact devices.

Acknowledgments

The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] S. Agrawal, D. M. Freeman, and V. Vaikuntanathan, "Functional encryption for inner product predicates from learning with errors," in *Proceedings of The 17th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'11)*, pp. 21–40, Dec. 2011.

[2] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," in *Proceedings of The 3rd Innovations in Theoretical Computer Science Conference (ITCS'12)*, pp. 309–325, Jan. 2012.

[3] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *Proceedings of The IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS'11)*, 2011. DOI: 10.1109/FOCS.2011.12

[4] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in *Proceedings of The 31st Annual Conference on Advances in Cryptology (CRYPTO'11)*, pp. 505–524, Aug. 2011.

[5] J. H. Cheon, J. S. Coron, J. Kim, M. S. Lee, T. Lepoint, and M. Tibouchi, "A batch fully homomorphic encryption over the integers," in *Proceedings of The Advances in Cryptology (EUROCRYPT'17)*, pp. 315–335, 2013.

[6] D. H. Duong, M. P. Kumar, and M. Yasuda, "Efficient secure matrix multiplication over lwe-based homomorphic encryption," *Tatra Mountain Mathematical publication*, vol. 67, pp. 69–83, 2016.

[7] S. D. Galbraith, *Space-efficient Variants of Cryptosystems based on Learning with Errors*, 2013. (`https://www.math.auckland.ac.nz/~sgal018/compact-LWE.pdf`)

[8] C. Gentry, "A fully homomorphic encryption scheme," *ACM Digital Library*, 2009. ISBN: 978-1-109-44450-6

[9] C. Gentry, S. Halevi, and V. Vaikuntanathan, "A simple bgn-type cryptosystem from LWE," in *Proceedings of The EUROCRYPT*, pp. 506–522, 2010.

[10] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Proceedings of TheAdvances in Cryptology (CRYPTO'13)*, pp. 75–92, 2013.

[11] I. Jabbar and S. N. Alsaad, "Design and implementation of secure remote e-voting system using homomorphic encryption," *International Journal of Network Security*, vol. 19, no. 5, pp. 694–703, 2017.

[12] R. Lindner and C. Peiker, "Better key sizes (and attacks) for lwe-based encryption," in *Proceedings of The Cryptographers Track at the RSA (CT RSA'11)*, pp. 319–339, 2011.

[13] L. Liu and J. YeA, "Homomorphic universal reencryptor for identity-based encryption," *International Journal of Network Security*, vol. 19, no. 1, pp. 11–19, 2017.

[14] V. Lyubashevsky, C. Peikertand, and O. Regev, "On ideal lattices and learning with errors over rings," in *Proceedings of The EUROCRYPT*, pp. 1–23, 2010.

[15] D. Rappe, "Homomorphic cryptosystems and their applications," *ResearchGate*, 2004. DOI: 10.17877/DE290R-15728

Table 1: Storage required for 1024 public keys at different security levels

	Parametres		Storage space in bits	
Security Level	n	q **is an integer of**	**Standard LWE scheme**	**Proposed scheme**
Toy	10	10 bits	112624	24477
Small	100	20 bits	2068323	40957
Medium	1000	30 bits	30748377	61436
Large	10000	40 bits	409609747	81914
Very Large	100000	50 bits	5121138606	102422

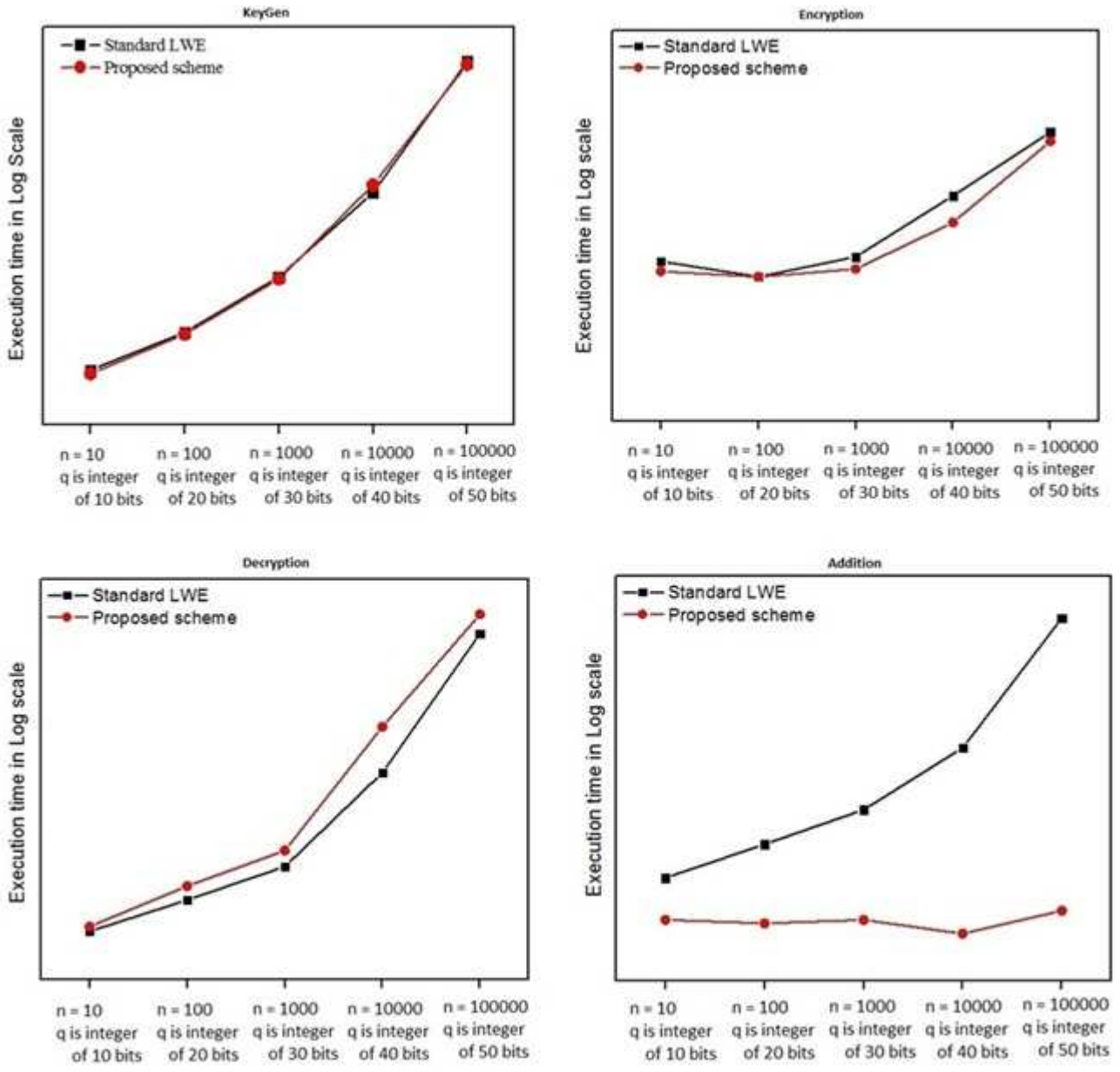

Figure 2: Practical performance of key generation, encryption, decryption and addition functions of the schemes (standard and proposed) at different security levels

[16] C. Ratnakumari and G. VijayaKumari, "An efficient LWE-based additively homomorphic encryption with shorter public keys," in *Proceedings of The Progress in Intelligent Computing Techniques: Theory, Practice, and Applications (ICACNI'16)*, pp. 171–177, 2018.

[17] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM*, vol. 56, no. 6, 2009.

[18] O. Regev, "The learning with errors problem (invited survey)," in *Proceedings of The IEEE 25th Annual Conference on Computational Complexity (CCC'10)*, pp. 191–204, June 2010.

[19] R. Rivest, L. Adleman, and L.M. Dertouzos, "On data banks and privacy homomorphisms," *Academic Press, Massachusetts Institute of Technology, Cambridge, Massachusetts*, vol. 56, no. 6, pp. 169–180, 1978.

[20] N.P. Smart and F. Vercauteren, "A fully homomorphic encryption with relatively small key and ciphertext sizes," in *Proceedings of The Public Key Cryptography (PKC'10)*, pp. 420–443, 2010.

[21] D. Stehle and R. Steinfeld, "Faster fully homomorphic encryption," in *Proceedings of The Advances in Cryptology (ASIACRYPT'10)*, pp. 377–394, 2010.

[22] M. VanDijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "A fully homomorphic encryption over the integers," in *Proceedings of The Advances in Cryptology (EUROCRYPT'10)*, pp. 24–43, 2010.

Biography

RatnaKumari Challa biography. She has received her M.Tech degree in Computer Science from University of Hyderabad, India in 2009. She is an Assistant Professor in Department of Computer Science and Engineering, RGUKT, IIIT-AP, Andhra Pradesh, India. She is currently pursuing Ph.D in JNTUH, Hyderabad, India. Her research interests include Security, Cloud Computing and Image processing.

VijayaKumari Gunta biography.She has received her Ph.D degree from University of Hyderabad, India in 2002 . She is a professor in Department of Computer Science and Engineering, JNTUH, Hyderabad. India. Her research interests include Algorithms, Security and Cloud Computing.