International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 21 (2017) pp. 10477-10484

An Efficient Symmetric Based Algorithm for Data Security in Cloud Computing Through Homomorphic Encryption Scheme

V Biksham
Research Scholar, Jawaharlal Nehru Technological University, Hyderabad,
Associate Professor, Department of Computer Science & Engineering,
Sreyas Institute Of Engineering & Technology, Bandlaguda,
Nagole, Hyderabad-500068, Telangana, India.

Orcid Id: 0000-0003-0532-594X

D Vasumathi
Professor, Department of Computer Science & Engineering,
Jawaharlal Nehru Technological University, Hyderabad,
Kukatpally, Hyderabad-500085, Telangana, India.

Abstract

The outstanding research topic on cryptography is called Fully Homomorphic Encryption (FHE), which allows computations to be carried out on encrypted data to the untrusted server of the security and privacy concerned related to emerging technologies like cloud computing. Various FHE schemes were developed after the first invention of Craig Gentry in 2009 which security is relying on bootstrapping and squashing. It is based on ideal lattices and it is more complex and huge computational cost and unpractical, but it is enough for theoretically feasible. In this paper, symmetric based lightweight Fully Homomorphic Encryption scheme is proposed for the Somewhat Homomorphic scheme that is based on GV system and uses on matrix instead of integers. we reduce the key size significantly by introducing Reduced Approximate GCD problem. Another part is proving the scheme is semantically secure under Approximate GCD. Finally, we proposes a new algorithm for key generation and refreshed for each computation with a stipulated time interval.

Keywords: Cryptography, Homomorphic Encryption, Untrusted server, Computation, Bootstrapping, Squashing.

INTRODUCTION

Cloud computing is an emerging technology where we can store and access the resources and infrastructures via Internet with low cost. The Cloud users can outsource the IT products from the Cloud Service Providers (CSPs) for development of various projects for a stipulated time [1]. Simply a cloud computing is a buying a resources like software's, tools , applications, servers, storage space and network setup through online with low cost and high efficiency on a contract period. Today's most of IT companies maintain their own cloud for making the easy and template based accessed products. The biggest challenge for researchers is providing security and privacy of cloud users/customers data that is kept stored on cloud servers. One of the existing methods for data security is data encryption. In data encryption the cloud users/customer can encryption for their data by using a key and algorithm. If the key is same for both encryption and decryption called as symmetric where as encryption key is public key and decryption key called as private key as the system in asymmetric or public key cryptography. The cloud user kept their sensitive data at cloud servers with encrypted format. However any computations like searching, sorting, addition or multiplication and XOR operations on cipher text. The client can decrypt the cipher text which is stored in cloud server by using key or client can give the credentials like key to the CSPs for doing manipulations on server. But client cannot trust the CSPs for updating the client's confidential data while exploiting security levels [2]. Then, to preserving the data secure at server side, the client itself do the decryption process to obtain plaintext and do the any type of modification, calculations and update the plaintext and later perform the encryption and generate the cipher text to preserve at server. Here, we observe the some security defect is that frequent decryption and computation may gain chance to the attackers to chosen cipher text attack and exploit the data integrity and authentication. To avoid the above, in 1978, the Rivert et all proposes the privacy homomorphism [3] which does the computations on cipher text instead of decrypting it.

HOMOMORPHIC ENCRYPTION

Homomorphic Encryption is a form of encryption where we perform some calculations on already encrypted data without decryption and matches the results of which perform calculations on plain text. Homomorphic Encryption is rich method in cloud computing while doing client server communication with multiple navigations. In cloud computing

customers can store their sensitive data on cloud, but to perform any computations like updating, searching and sorting, they supposed to decrypt the cipher text and perform operations and send to the cloud in the form of encryption. However multiple times decryption and updating the data, the customers need to depends on the cloud service providers (CSPs) [4][25]. In spite the cloud users depends on the CSPs for key distribution. Homomorphic Encryption provides an environment to the clients no need to share the keys to any cloud service providers while storing or accessing the cloud. Homomorphic encryption is the conversion of data into cipher text that can be analyzed and worked with as if it were still in its original form.

Homomorphic Encryption allows complex mathematical operations to be performed on encrypted data without compromising the encryption. In mathematics, Homomorphic describes the transformation of one data set into another while preserving relationships between elements in both sets. The term is derived from the Greek words for "same structure." Because the data in a Homomorphic encryption scheme retains the same structure, identical mathematical operations, whether they are performed on encrypted or decrypted data will yield equivalent results. Homomorphic Encryption is expected to play an important part in cloud computing, allowing companies to store encrypted data in a public cloud and take advantage of the cloud provider's analytic services [5] [26].

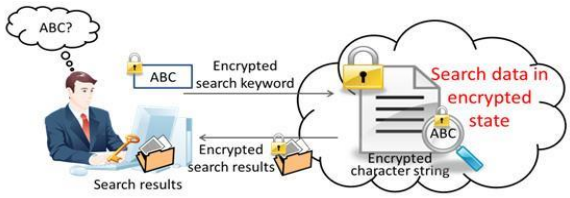

Figure 1: Homomorphic Encryption

The HE is defined as the arbitrary computations (i.e. Functions) are performed on encrypted data at server side without knowing the decryption key. However when results are decrypted by client using private key which matches the same results performed on the plaintext. The following equation shows the basic definition of Homomorphic Encryption. The operations include searching, sorting, filtering, addition, multiplication, XOR and any function which are performed on ciphertext by the server without knowing the key. When the client decrypt it matches the results of operations to the performed on plaintext.

$$\text{Operation (plain)} = = \text{decrypt (operation}^{\cdot} \text{ (encrypt (plain)))}$$

The figure1 shows that, client need to search for a data called ABC in cloud server and data in encrypted format called as query that is send to the cloud server for searching the ABC string on the cloud server. The Homomorphic Encryption principle searches the string according to the client requirement and if found the send the results back to the client in encrypted format only. The client by using their private key decrypts the encrypted search results. In cryptography, Homomorphic Encryption is process of performing similar operations on encryption data (cipher text) without decryption of cipher text and sends the results back to the client in encrypted format only. But the result matches when performing same operations on plaintext. Let *M* (or *C*) denote the set of the plaintexts (or cipher texts, respectively). An encryption scheme is said to be Homomorphic if for any given encryption key *k* the encryption function E satisfies [6]

$$m1, m2 \in M, E(m_1 \circ m2) = E(m_1) \circ E(m_2)$$

Informally speaking, Homomorphic cryptosystem is a cryptosystem with the additional property that there exists an efficient to compute an encryption of the sum or the product, of two messages given the public key and the encryptions of the messages but not the messages themselves. If M (or C) is an additive (semi-) group then the scheme is algorithm called Additively Homomorphic and the algorithm is called Addictive Homomorphic encryption. Otherwise the scheme is called multiplicatively Homomorphic and the algorithm is called multiplicative Homomorphic encryption. The following table (i) shows the various Homomorphic Encryption schemes with applications

Table i: Homomorphic Encryption methods and applications

Crypto system	Addictive	Multiplicative	XOR	Mixed	Application
Paillier	√	X	X	X	e-voting system, threshold scheme
RSA	X	√	X	X	To secure Internet, Banking and credit card secure transaction
ElGamal	X	√	X	X	In Hybrid systems
Goldwasser-Micali	X	X	√	X	Biometric Authentication
Boneh, Goh, and Nissim (BGN).	Many additions	One Multiplications	X	√	Stack Exchange
Brakerski, Gentry and Vaikuntanathan(BGV)	X	X	X	√	For the security of integer polynomials
EHC(Extended Homomorphic Cryptosystem)	X	X	X	√	Efficient Secure Message Transmission in MANETs

The advantages and applications of the Homomorphic Encryption scheme is include No longer usage of private of client by the Cloud service Providers (CSPs), In Medical records, Analyze disease/treatment without disclosing confidential data and able to search for DNA markers without revealing DNA [7], Spam filtering-Blacklisting encrypted mails and third parties can scan you PGP traffic and easy to implement the electronic voting system [8].

Somewhat Homomorphic Encryption (SHE)

The Somewhat Homomorphic Encryption is defined as the computations to be carried out on cipher text as either additive or multiplicative and quadratic functions etc, but not both combined. The SHE is also called as Partially Homomorphic cryptosystem. The following list of cryptosystem is examples of SHE system with either additive or multiplicative operations.

The Unpadded RSA [9]. If the RSA public key is modulus m and exponent e, then the encryption of a message x is given by $E(x) = x^e \bmod n$. The Homomorphic property is then $E(x_1).E(x_2) = x_1^e x_2^e \bmod n = E(x_1.x_2) \bmod n$.

ElGamal, In the ElGamal cryptosystem[10], in a cyclic group G of order q with generator g, if the public key is (G,q,g,h), where $h=g^x$, and x is the secret key , then the encryption of a message m is $E(m)=(g^r, m.h^r)$, for some random $r \in \{0,1,\ldots,q-1\}$. The Homomorphic Encryption property is then defined as $E(m_1).E(m_2)=(g^{r1}, m_1.h^{r1})(g^{r2}, m_2.h^{r2})=(g^{r1+r2}, (m_1.m_2)h^{r_1+r_2})=E(m_1.m_2)$.

Goldwasser-Micali, In the Goldewasser-Micali cryptosystem [11], if the public key is the modulus m and quadratic non-residue x, then the encryption of a bit b is $E(b) = x^b r^2 \bmod m$, for some random $r \in \{0, 1\ldots m-1\}$. The Homomorphic property is then $E(b_1).E(b_2) = x^{b1} r_1^2 x^{b2} r_2^2 \bmod m = x^{b1+b2} (r_1 r_2)^2 \bmod m = E(b_1 \oplus b_2)$ where $\oplus$ denotes addition modulo 2, (i.e. exclusive –or).

Benaloh, In the Benaloh cryptosystem[12], if the public key is the modulus m and the base g with a block size of c, then the encryption of a message x is $E(x)=g^x r^c \bmod m$, for the random $r \in \{0,\ldots,m-1\}$. The Homomorphic property is then $E(x_1).E(x_2) \bmod m = (g^{x1} r_1^c)(g^{x2} r_2^c) \bmod m = g^{x1+x2}(r_1 r_2)^c = E(x_1+x_2 \bmod m)$

Paillier, In the Paillier cryptosystem [13], if the public key is the modulus m and the base g, then the encryption of a message x is $E(x) = g^x r^m \bmod m^2$, for some random $r \in \{0,\ldots, m-1\}$. The Homomorphic property is then $E(x_1).E(x_2) = (g^{x_1} r_1^m)(g^{x_2} r_2^m) \bmod m^2 = g^{x_1+x_2}(r_1 r_2)^m \bmod m^2 = E(x_1 + x_2)$.

Fully Homomorphic Encryption (FHE)

A cryptosystem that supports arbitrary computation*s* (i.e. both additive and multiplicative functions) on cipher texts is known as fully homomorphic encryption (FHE) and is far more powerful. The existence of an efficient and fully homomorphic cryptosystem would have great practical implications in the outsourcing of private computations in the cloud computing. The utility of fully homomorphic encryption has been long recognized: the problem of constructing such a scheme was first proposed in 1978, within a year of the development of RSA. A solution proved more elusive; for more than 30 years, it was unclear whether fully homomorphic encryption was even possible. During that period, partial results included the Boneh–Goh–Nissim cryptosystem [14] that supports evaluation of an unlimited number of addition operations but at most one multiplication, and the Ishai-Paskin cryptosystem [15] that supports evaluation of (polynomial-size).

Table ii: Existing Homomorphic Encryption scheme and their remarks

Year	Homomorphic Encryption scheme	Remarks
2009	First FHE by Gentry	Gentry & Van Dijk simplify: No more lattices, integers instead
2010	Smart & Vercauteren	Simplify: Reduce key size
2011	Gentry & Halevi	Present working FHE implementation (the one w/2.3GB public key, runs on workstation)
	Coron, Naccache & Tibouchi	Reduces public key size
	Gentry & Brakerski	Shows removal bootstrapping(Leveled FHE)
	Smart & Vercauteren	Implement Single Instruction Multiple Data(SIMD)
2012	Gentry,Smart & Halevi	Elevate SIMD to general circuits.
	Boneh,Gentry & Halevi	Show another PoC using SWHE.
	Brakerski,Gentry & Vaikuntanathan	Improves Leveled FHE
2013	Shai Halevi	Improves speed and reduces noise by modulus switching

2014	GV Method	Reduced size of the public key and time complexity of both encryption and decryption based on Error Approximate GCD.
2015	Cheon, Jung Hee	The polynomial approximate common divisor problem and its application to the fully homomorphic encryption
2016	Dasgupta Smaranika, and S. K. Pal.	Design of a polynomial ring based symmetric homomorphic encryption scheme

Proposed Fully Homomorphic Encryption

Main Idea: In this section, we have presented our proposed system for fully Homomorphic encryption, which is based on symmetric key. Our scheme consists of sub-modules, which are as below:-

-Key generation step

-Encryption procedure

-Decryption procedure

-Refresh procedure

The complete description and steps involved in our FHE scheme are presented as below:-

Algorithm 1 KeyGen (l, a)-

1. Randomly choose 2a odd number pairs x_i and y_i, $1 \leq i \leq a$. Selected numbers. Should be pair wise co-prime with each other, consist the size of l-bits.
2. Compute $S=\prod_{i=1}^{a} n_i$, where $n_i=x_i * y_i$
3. Choose an orthogonal matrix K, with its dimension as 4, in Z_S. Follow below approach to choose K -

   Randomly choose a matrix in the search space of Z_S. Check if it is following the orthogonality property i.e. $K.K^T = K^T.K = I$ or $K^{-1} = K^T$, then search is completed, otherwise repeat until an orthogonal matrix is found.
4. Compute the transpose of matrix K -

   $K^T \longleftarrow$ transpose (K) mod Z_S

   Note: From matrices orthogonality property, $K^{-1} = K^T$
5. Orthogonal matrix K will act as symmetric key in our cryptosystem.

Algorithm 2 Enc_Procedure (M, n_i, S, K, K^T)-

1. Consider plaintext as M.
2. Choose an integer R randomly, where R € ZS and R ≠M.
3. Create a matrix Y with its dimension as (a×3), where each row consists of single occurrence of M and rest of two occurrences as R.
4. Employ CRT to obtain solution of simultaneous equations -

   $\alpha_j(1 \leq j \leq 3) \equiv Y_{ij} \bmod n_i$ where, $1 \leq i \leq a$.
5. Use Coppersmith-Winograd algorithmic procedure [16] for below step of matrix multiplication computation.
6. Obtained cipher text $C = K^T \times d(M, \alpha_1, \alpha_2, \alpha_3) \times K$ where, d $(M, \alpha_1, \alpha_2, \alpha_3)$: denotes the diagonal matrix with diagonal elements as parameters.

Algorithm 3 Dec_Procedure(C, S, K, K^T)-

1: Compute plaintext as, $P = K \times C \times K^T$

2: $M \longleftarrow [P]_{11}$

Algorithm 4 Refresh Procedure(S)-

1: Refresh the symmetric key, which is an orthogonal matrix as-

$$K' \longleftarrow \text{randortho (t)}$$

Here, randortho () is a randomized function generating new K of dimension t randomly in the space of Z_S.

Analysis of Proposed algorithm:

Some observations and analysis of our proposed scheme is presented as below:-

Correctness of Decryption algorithm – we observe that,

Dec_Procedure(C, S, K, K^T)=>$K \times C \times K^T$

Since we know the property of an orthogonal matrix as, $K.K^T=K^T.K=I$ or $K^{-1}=K^T$

So, Dec_Procedure(C, S, K, $K^{T)}$ =>$K \times K^T \times d(M, \alpha_1, \alpha_2, \alpha_3) \times K \times K^T$

- ⇨ $I \times d(M, \alpha_1, \alpha_2, \alpha_3) \times I$
- ⇨ $d(M, \alpha_1, \alpha_2, \alpha_3)$
- ⇨ [p]11
- ⇨ M

Homomorphic Properties

Our proposed Fully Homomorphic encryption scheme is satisfying both multiplicative as well as additive Homomorphic properties.

- **Multiplicative Homomorphic property**

Consider C_1, C2 are cipher texts corresponding to plaintexts M_1, M2.

$C_1=K^T\times d\ (M_1, \alpha1, \alpha2, \alpha3) \times K$

$C_2= K^T\times d\ (M_2,\alpha'_1, \alpha'_2, \alpha'_3) \times K$

Now,C_1 x $C_2=K^T\times d\ (M_1,\alpha'_1,\alpha'_2,\alpha'_3)\times K\times K^T\times d(M_2, \alpha'_1,\alpha'_2,\alpha'_3) \times K$

$=K^T\times d(M_1,\alpha_1,\alpha_2,\alpha_3)\times I\times d(M_2, \alpha'_1,\alpha'_2,\alpha'_3) \times K$

$= K^T\times d(M_1,\alpha_1,\alpha_2,\alpha_3)\times d(M_2, \alpha'_1,\alpha'_2,\alpha'_3) \times K$

$=K^T\times d(M_1\times M_2), \alpha''_1, \alpha''_2, \alpha''_3) \times K$

Now,Dec_Procedure(C,S,K,K^T)=>$K\times K^T\times d((M_1\times M_2), \alpha''_1, \alpha''_2, \alpha''_{3)} \times K\times K^T$

=>$I\times d((M_1\times M_{2)}, \alpha''_1, \alpha''_2, \alpha''_3) \times I$

=>$\times d((M_1\times M_{2)}, \alpha''_1, \alpha''_2, \alpha''_3)$

=>$\times (M_1\times M_2)$

- **Additive Homomorphic property**

Consider, C_1, C_2 are cipher texts corresponding to plaintexts M_1, M2.

$C_1=K^T\times d(M_1, \alpha_1,\alpha_2,\alpha_3)\times K$

$C_2= K^T\times d(M_2, , \alpha'_1,\alpha'_2,\alpha'_3) \times K$

Now, $C_1+C_2= (K^T\times d(M_1, \alpha_1,\alpha_2,\alpha_3)\times K)+(K^T\times d(M_2, , \alpha'_1,\alpha'_2,\alpha'_3) \times K)$

$=K^T\times d((M_1+M_2), \alpha''_1, \alpha''_2, \alpha''_{3)} \times K$

Now,Dec_Procedure(C,S,K,K^T) => $K\times K^T\times d(M_1+M_2), \alpha''_1, \alpha''_2, \alpha''_3) \times K\times K^T$

=> $I\times d((M_1+M_2), \alpha''_1, \alpha''_2, \alpha''_3)\times I$

=> $d((M_1+M_2), \alpha''_1, \alpha''_2, \alpha''_3)$

=> (M_1+M_2)

Example: Algorithm 1: KeyGen (l, a)-

1. Randomly choose a =2 then 2a=2.2=4 odd numbers and two pairs (x_1,y_1) and (x_2,y_2), choose (5,7),(11,13)
2. Compute $S =\prod^a_{i=1} n_i$ where $n_i =x_i * yi$ => n=n1 * n2 n1=5*11=55, n2=7*13=91. Therefore S=55*91=5005 Zs limit is {0,1,….5004}
3. Choose Orthogonal Matrix K, with it's dimension as 4, in Zs

$$k=\begin{pmatrix} 0.28 & -0.26 & 0.07 & 0.09 \\ 0.09 & -0.3 & 0.04 & 0.17 \\ -0.01 & 0.05 & 0.9 & -0.07 \\ 0.27 & 0.89 & -0.02 & 0.34 \end{pmatrix}\quad 4X4$$

Orthogonal matrix property: $K.K^T=I$, $K^{-1}=K^T$

4. Compute K^T of matrix K
5. Orthogonal matrix K will act as symmetric key in our cryptosystem.

Algorithm 2: Enc_Procedure (M, n_i, S, K, K^T)-

1. Consider plaintext as M=257
2. Choose an integer R randomly, R=291 where R € ZS and R ≠M.
3. Create a matrix Y with its dimension as (a×3), where each row consists of single occurrence of M and rest of two occurrences as R.

$$Y=\begin{pmatrix} 291 & 257 & 291 \\ 291 & 291 & 257 \end{pmatrix}\quad 2X3$$

291 mod 55 257 mod 55 291 mod 55----- row1
291 mod 91 257 mod 91 291 mod 91----- row2

4. Employ Chinese Remainder Theorem (CRT) [24] to obtain solution of simultaneous equations –

$\alpha_j(1 \le j\le 3) \equiv Y_{ij} \bmod n_i$ where, $1 \le i \le a$.

291 mod 55=α1 257 mod 55= α2 291 mod 55= α3
291 mod 91=α1 257 mod 91= α2 291 mod 91= α3

Find α1, α2, α3 values, using CRT, we get α1=291, α2=236 α3=312

5. Use Coppersmith-Winograd algorithmic procedure for below step of matrix multiplication computation.

$C = K^T \times d\ (M, \alpha_1, \alpha_2, \alpha_3) \times K$

$$\begin{pmatrix} K^T \end{pmatrix} X \begin{pmatrix} 257 & 0 & 0 & 0 \\ 0 & 291 & 0 & 0 \\ 0 & 0 & 236 & 0 \\ 0 & 0 & 0 & 312 \end{pmatrix} X \begin{pmatrix} K \end{pmatrix}$$

6. Obtained cipher text $C = K^T \times d(M, \alpha_1, \alpha_2, \alpha_3) \times K$ where, $d(M, \alpha_1, \alpha_2, \alpha_3)$: denotes the diagonal matrix with diagonal elements as parameters

Algorithm 3: Dec_Procedure(C, S, K, K^T)-

1. Compute plaintext as, $P = K\times C\times K^T$
2. $M \Longleftarrow [P]_{11}$

Algorithm 4: Refresh Procedure(S)-

1: Refresh the symmetric key, which is an orthogonal matrix as-

$K' \Longleftarrow$ randortho(t)

Here, randortho() is a randomized function generating new K of dimension t randomly in the space of Z_S.

$X_k= (aX_{(k-1)} +C) \bmod m$

Here X is new key, k=1….m, C, m are Constants

Constraints:

i) Both m and C must be co primes with each other
ii) (a-1) must be divisible by all prime factors of m

iii) New generated key series should be <m with uniqueness

C=11, m=23 both co-primes (satisfy constraint (i), then choose a=47, (a-1) satisfy constraint (ii) then possible m= {0 ….22}, the generated new key series is unique and equal gap size key and <m (satisfy constraint(iii))

$X_1 = (47X_0+11) \bmod 23$

$X_2 = (47X_1+11) \bmod 23$

.

.

$X_n = (47X_{n-1}+C) \bmod 23$

RESULTS& DISCUSSION

Proposed FHE scheme is lightweight in nature and utilizes matrices computational operations, which are lightweight in nature as compare to working with polynomial computations. Proposed FHE scheme is parallelizable. As our proposed FHE scheme utilizes matrices computational operations while encryption as well as decryption, which give the advantage of performing outer product matrix vector multiplication that is very much parallelizable. Computational Complexity improvement, we are utilizing Coppersmith-Winograd algorithmic procedure [16] for matrix multiplication computation in encryption step. This method gives a significantly improved matrix multiplication computational complexity as $O(n^{2.376})$.

Performance of SHE

The performance of Somewhat Homomorphic Encryption is based on the time taken to generate Key [17] [27]. Estimated time for Encryption, Decryption and degree of polynomials with respect to size of plaintext [18][19][28]. The following table (iii) the various Homomorphic Encryption methods and their performance and comparisons various items with existing approaches respectively

Table iii: Execution time of Keygen(), Encry() and Decry()

Size of S(in bits)	Key gen()	Engcry()	Decry()
16-bits	0.332 Sec.	0.173 Sec.	0.096 Sec.
32-bits	28.41Sec	55.20 Sec.	15.0 Sec
50-bits	59.85 Sec.	71.19 Sec	22.40 Sec.
64-bits	1118.42 Sec.	204.91 sec.	102.45 Sec.

The following figure (ii) shown the x -axis represents size of s (in bits) of plaintext and Execution time (in seconds) for encryption, decryption and key refresh at y-axis. it shows the size is proportional to the execution time.

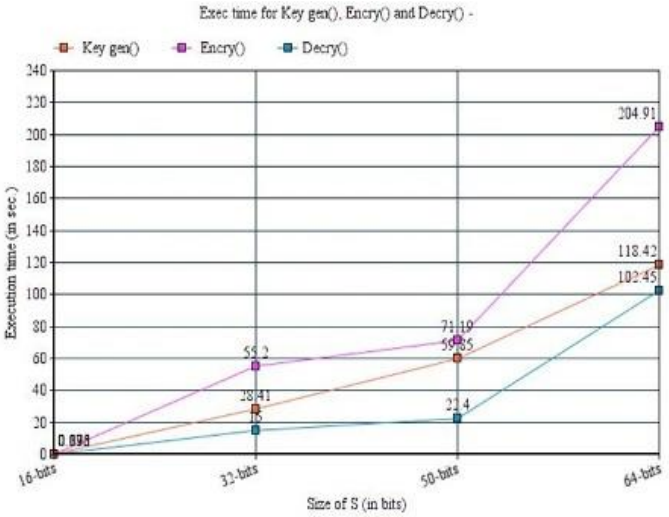

Figure ii: Execution time for encryption, decryption and refreshing a key.

Table (iv) represent the comparison of existing methods with newly proposed method with various parameters as shown below.

Table iv: Performance: SWHE

Item of Comparison	DGHV SHE	CMNT SHE	GV SHE	Proposed SHE
Compactness	No	Yes	Yes	Yes
Dimension	2048 (8000,000-bit integers)	8192 (3,200,000-bit integers)	32768 (13,000,000-bit integers)	33024
KeyGen	1.25Sec	10 Sec	95 Sec	59.85 Sec.
Enc amortized	.060 Sec	.7 Sec	5.3 Sec	71.19 Sec.
Mult/ Dec	.023 Sec	.12 Sec	.6 Sec	22.40 Sec.
Degree	~200	~200	~200	~200
Security base	Approximate GCD	PAGCD(Error-free approximate GCD	Two-element PAGCD	Reduced Approximate GCD

Performance of FHE

The performance of Fully Homomorphic Encryption is based on the time taken to generate Key[20], Estimated time for Encryption, Decryption and degree of polynomials [21][22][23] with respect to size of plaintext. The following table (v) shows the various Homomorphic Encryption methods and their performance.

Table v: comparison of proposed algorithm with existing methods.

Item of Comparison	DGHV SHE	CMNT SHE	GV SHE	Proposed FHE
Compactness	No	Yes	Yes	Yes
Dimension	2048	8192	32768	33024
KeyGen	40Sec	8 Min	2 hours	7 min
PK Size	70MByte	285 MByte	2.3GByte	325MBytes
Enc amortized	.060 Sec	0.7 Sec	5.3 Sec	6.5 Sec
Mult/Dec	.023 Sec	.12 Sec	.6 Sec	.8 sec
Recrypt	31 Sec	3 Min	30 Min	20 min
Degree	~200	~200	~200	~200
Security base	Approximate GCD	PAGCD(Error-free approximate GCD	Two-element PAGCD	Reduced Approximate GCD

Performance of Computations:

Table vi: Comparison of various schemes over computations

Item of Comparison	DGHV SHE	CMNT SHE	GV SHE	Proposed FHE
Modulus	257	8209	65537	65793
Time for addition(ms)	0.7	0.7	2.9	1.9
Time for multiplication (ms)	39	38	177	42

CONCLUSION

In this paper, an efficient, conceptually simple, semantically secure and possibility for practical applications like Homomorphic Encryption principles at cloud users is proposed and reduced the key sizes and time for encryption and recrypt operations. In future we proposes this system can implement for asymmetric crypto system where we can use the two keys one for encryption and other for decryption that can more reliable and scalable with high security measures.

REFERENCES

[1] W. Liu, "Research on cloud computing security problem and strategy," 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), Yichang, 2012, pp. 1216-1219.doi: 10.1109/CECNet.2012.6202020

[2] Kevin Hamlen, Latifur Khan, Murat Kantarcioglu, Bhavani Thuraisingham, The University of Texas at Dallas, USA, "Security Issues for cloud computing" International Journal of Information Security and Privacy, 4(2), 39-51, April-June 2010 pp 39-51

[3] Ronald L. Rivest, Len Adleman Michael L. Dertouzos "On Data Banks and Privacy Homomorphisms" Massachusetts Institute of Technology Cambridge, Massachusetts, © 1978 by Academic Press, Inc.

[4] Dishita Dave,Rikkin Thakkar "Homomorphic Encryption In Cloud Computing" 2015 IJIRT | Volume 1 Issue 12 | ISSN: 2349 -6002.

[5] A.P.Nirmala, Dr.R.Sridaran "Cloud Computing Issues at Design and Implementation Levels-A Survey" Int.J.Advanced Networking and Applications, Volume: 03, Issue: 06 Pages 1444-1449(2012) ISSN: 0975-0290.

[6] Frederik Armknecht, Colin Boyd, Christopher Carr, KristianGjøsteen, Angela J aschke, Christian A. Reuter, and MartinStrand "A Guide to Fully Homomorphic Encryption" https://eprint.iacr.org/2015/1192.pdf

[7] K. Lauter, A. L´opez-Alt, and M. Naehrig. Private computation on encrypted genomic data. In Proceedings of Progress in Cryptology - LATINCRYPT 2014, volume 8895, pages 3–27

[8] S. M. Anggriane, S. M. Nasution and F. Azmi, "Advanced e-voting system using Paillier Homomorphic encryption algorithm," 2016 International Conference on Informatics and Computing (ICIC), Mataram, 2016, pp. 338-342.doi: 10.1109/IAC.2016.7905741

[9] Maha TEBAA,Said EL HAJII , V–Agdal, "Secure Cloud Computing through Homomorphic Encryption " International Journal of Advancements in Computing Technology(IJACT) Volume5, umber16, Dec- 2013.

[10] T.ElGamal. A Public –Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. Crypto'84 pp.469-472.

[11] S.Goldwasser, S. Micali (1982). "Probabilistic encryption and how to play mental poker keeping secret all partial information". Proc. 14th Symposium on Theory of Computing: 365–377. doi:10.1145/800070.802212.

[12] J.Benaloh.Verifiable secre-ballot elections. Ph.D thesis, Yale University, Dept. of Computer Science, 1988.

[13] P.Pailliar. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. Eurocrypt'99 pp-223-238

[14] D.Benoh,E.Goh, and K.NIssim. Evaluating 2-dnf formulas on ciphertexts, In proceedings of Theory of Cryptography(TCC)'05, LNCS 3378,pages 325-341.

[15] Yuval Ishai and Anat Paskin. 2007. Evaluating branching programs on encrypted data. In Theory of Cryptography. Springer, 575–594

[16] D. Coppersmith and S. Winograd. "Matrix `multiplication via arithmetic progressions", J. Symbolic Computation, 9(3):251-280, (1990).

[17] C. Gentry "Fully Homomorphic encryption using ideal lattices" In STOC pp 169-178 ACM 2009.

[18] Y.Govinda Ramaiah,G.Vijaya kumari "Complete Privacy preserving Auditing for Data Integrity in Cloud Computing" published in 2013 12th IEEE International Conference on Trust,Security and Privacy in Computing and Communications, 2013, DOI: 10.1109/TrustCom.2013.191

[19] N. P. Smart F. Vercauteren "Fully Homomorphic encryption with relatively small key and ciphertext sizes" In Public Key Cryptography-PKC'10 Vol. 6056 of LNCS pp. 420-443 Springer 2010

[20] M. V. Dijk C. Gentry S. Halevi V. Vaikuntanathan "Fully Homomorphic encryption over the integers" Proceedings of Eurocrypt Vol. 6110 of LNCS pp. 24-43 Springer 2010.

[21] Dasgupta Smaranika, and S. K. Pal. "Design of a polynomial ring based symmetric homomorphic encryption scheme", Perspectives in Science 8 (2016), ELSEVIER: pp. 692-695.

[22] Yagisawa, Masahiro. "Fully Homomorphic Public-key Encryption Based on Discrete Logarithm Problem", IACR Cryptology e-Print Archive (2016): 54.

[23] Cheon, Jung Hee. "The polynomial approximate common divisor problem and it's application to the fully homomorphic encryption", Information Sciences-326 (2016): pp.41-58.

[24] Y.H.KuXiaoguangSun, Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, PA 19104, USA, "The chinese remainder theorem" Journal of the Franklin Institute Volume 329, Issue 1, Jan 1992, Pages 93-97,https://doi.org/10.1016/0016-032(92)900993,http://www.sciencedirect.com/science/article/pii/0016003292900993

[25] Kristin Lauter, Michael Naehrig and Vinod Vaikuntanathan "an Homomorphic Encryption be Practical?" ACM CCSW 2011 paper. https://eprint.iacr.org/2011/405.pdf

[26] Shashank Bajpai and Padmija Srivastava "A Fully Homomorphic Encryption Implementation on Cloud Computing" International Journal of Information & Computation Technology. ISSN 0974-2239 Volume 4, Number 8 (2014), pp. 811-816© International Research Publications House http://www.irphouse.com, http://www.irphouse.com/ijict_spl/ijictv4n8spl_05.pdf

[27] Mr. V. Biksham, Dr. D. Vasumathi, "Query based computations on encrypted data through homomorphic encryption in cloud computing security," International Conference on Electrical, Electronics, and optimization Techniques (ICEEOT) ©2016.

[28] Y Govinda Ramaiah, G.Vijaya kumari "Efficient public key Homomorphic Encryption Over Integer Plaintexts" 978-1-4673-2588-2/12/$31.00 ©2012 IEEE