Circuits and Systems, 2016, 7, 3193-3203
Published Online August 2016 in SciRes. http://www.scirp.org/journal/cs
http://dx.doi.org/10.4236/cs.2016.710272

A Homomorphic Crypto System for Electronic Election Schemes

Kannan Balasubramanian, M. Jayanthi

Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi, India
Email: jayanthimathanan@mepcoeng.ac.in

Received 5 May 2016; accepted 15 May 2016; published 23 August 2016

Abstract

This research investigates the applications of homomorphic encryption systems in electronic voting schemes. We make use of Paillier cryptosystem which exhibits additive homomorphic properties. The other homomorphic cryptosystems RSA and Elgamal are not considered, since they exhibit only multiplicative homomorphic property. Our proposed method increases the level of security when compared to Elgamal method. It is more flexible when compared to previous schemes. We also propose data packing for efficient storage of election data. Finally, we demonstrate the advantages of the homomorphic encryption in voting schemes by comparing with other electronic voting scheme.

Keywords

Electronic Voting, Public Key, Homomorphic, Paillier, Data Packing

1. Introduction

Homomorphic encryption is the encryption on the already encrypted data rather than on the original data by providing the result as it is done on the plain text. The complex mathematical operations can be performed on the cipher text without changing the nature of the encryption. There are several efficient partially homomorphic cryptosystems and a number of fully homomorphic cryptosystems. High computational and communication complexity involved in using homomorphic encryption for the practical applications. The homomorphic property of various cryptosystems can be used to create secure voting systems, collision-resistant hash functions, and private information retrieval schemes and enable widespread use of cloud computing by ensuring the confidentiality of processed data.

This paper focuses on the problem of Data Protection which allows the computation of encrypted data, so that secure Database Storage is achieved. Noise refers to the distortion of cipher texts (*i.e.*, encoded text) that occurs

after each operation (e.g., addition or multiplication) is performed. As more and more additions and multiplications are performed, the noise level becomes too high, and the resulting ciphertexts become indecipherable. Ciphertexts can be refreshed easily by decrypting them, but the idea behind homomorphic encryption is to not share the secret key required to do the decryption.

2. Related Work

The introduction of Homomorphic encryption schemes was done by Rivest, Adleman and Dertouzos in [1]. They allow only computing over encrypted data either the product [2] or sum of the plaintext (Goldwasser-Micali and Paillier [3]) Brickell and Yacobi pointed out in [4] some security flaws in the first proposals of Rivest *et al*. In 1999 Pascal Paillier proposed a provable secure encryption system that was an additive Homomorphic encryption. In 2005, Dan Boneh, EU-Jin Goh and Kobi Nissim [5] invented a system of provable security encryption, with which unlimited number of additions but only one multiplication can be performed [6].

2.1. Encryption Schemes

Encryption schemes are designed to preserve confidentiality. There are two kinds of encryption schemes: symmetric and asymmetric encryption. Symmetric means that encryption and decryption are performed with the same key. Therefore, two persons who never met before cannot use this scheme directly. It has the advantage of being really fast and used as often as possible. In this category block cipher (AES) and stream ciphers (One-time pad, Snow 2.0), which are even faster [7]. In asymmetric the encryption key is public, as the decryption key remains private. It has a big drawback that is they are based on nontrivial mathematical computations, and much slower than the symmetric. The two most prominent examples are RSA and ElGamal. The Asymmetric encryption key is a Privacy preserving public key encryption [8].

Private Key encryption schemes can be used for our purpose, but they use only one key for both encryption and decryption. We believe that conventional public-key encryption schemes with modular exponentiations are secure, but modular exponentiation is not a very simple operation. But it uses two keys each for encryption and decryption. It has three algorithms: KeyGen, Encrypt and Decrypt. Keygen algorithm is used to create Keys for encryption and decryption. The Encrypt algorithm encrypts the Plaintext into Ciphertext using the key. The Decryption algorithm decrypts the Ciphertext using the Key. A Homomorphic public key encryption scheme ε has four algorithms. The usual KeyGen, Encrypt and Decrypt and an additional algorithm Evaluate. ε = (KeyGen, Encrypt, Decrypt, Evaluate) Evaluate takes as input a public key *pk*, cipher text $C = (C_1, C_2, \cdots, C_n)$ and outputs another cipher text *C*.

2.2. Types of Homomorphic Encryption Schemes

There are two types of homomorphic encryption: fully homomorphic encryption (FHE) and somewhat homomorphic encryption (SHE). Each type differs in the number of operations that can be performed on encrypted data.

2.2.1. Somewhat Homomorphic Encryption

It can evaluate low degree polynomials homomorphically. SHE cryptosystems support a limited number of operations (*i.e.*, any amount of addition, but only one multiplication) and are faster and more compact than FHE cryptosystems [9].

$$\mathrm{DEC}\left(SK, \mathrm{Eval}\left(PK, F, C_1, \cdots, C_n\right)\right) = \mathrm{F}\left(M_1, \cdots, M_n\right) \tag{1}$$

F can be an addition or multiplication function. (*sk*, *pk*) are generated by the KeyGen function. A scheme is additively homomorphic if it considers addition operators, and multiplicatively homomorphic if it considers multiplication operators. Unpadded RSA, ElGamal, Goldwasser-Micali, Benaloh, Paillier are coming under this encryption scheme.

2.2.2. Fully Homomorphic Encryption Schemes

FHE allows for an unlimited, arbitrary number of computations (both addition and multiplication) to be performed on encrypted data. Fully homomorphic encryption can be trivially realized from any secure, encryption

scheme, by an algorithm Evaluate that simply attaches a description of the C to the ciphertext tuple, and a Decrypt procedure that first decrypts all the ciphertexts and then evaluates C on the corresponding plaintext bits. Craig Gentry firstly constructed a "somewhat homomorphic" encryption (SHE) scheme that supports evaluation of low degree polynomials. Then he "squashed" the decryption algorithm to obtain a lower circuit depth so that the somewhat scheme is capable of evaluating its own decryption circuit. Finally, he used a "bootstrapping" technique to achieve a fully homomorphic encryption scheme [10].

Craig Gentry's technique is from a bootstrappable somewhat homomorphic scheme to the fully Homomorphic. The essence of fully homomorphic encryption is simple. In Fully Homomorphic Encryption, parties that do not know the plaintext data can perform computations on it by performing computations on the corresponding ciphertexts. Given ciphertext $C=\left(C_1,C_2,\cdots,C_n\right)$, fully homomorphic encryption should allow to output a cipher text that encrypts $\mathrm{F}\left(C\right)=\mathrm{f}\left(C_1,C_2,\cdots,C_n\right)$ for the function F, as long as that function can be efficiently computed. No information about $\left(C_1,C_2,\cdots,C_n\right)$ or $\mathrm{F}\left(C_1,C_2,\cdots,C_n\right)$, or any intermediate plaintext values, should leak. The inputs, output and intermediate values are always encrypted.

A fully homomorphic encryption scheme uses only simple integer arithmetic. However, constructing fully homomorphic signatures or even homomorphic signatures for more complex functions remains an important open problem.

2.3. Properties of Homomorphic Encryption Schemes

It has two properties, namely Additive Homomorphic encryption and Multiplicative Homomorphic encryption.

2.3.1. Additive Homomorphic Encryption Schemes

$$M_1,M_2\in Z_n \text{ and } r_1,\ r_2\in Z_n^*, \tag{2}$$

Z_n denote the set of nonnegative integers less than n. Z_n^* denote the set of integers that are relatively prime to n. g is a random number where it has ordered multiple of n. n is the product of two large primes p and q.

$$\mathrm{F}\left(M_1,M_2\right)=M_1+M_2$$

$$\mathrm{D}\left(\mathrm{E}\left(M_1,r_1\right)\cdot\mathrm{E}\left(M_2,r_2\right)\bmod n_2\right)=M_1+M_2 \bmod n.$$

$$\mathrm{D}\left(\mathrm{E}\left(M_1,r_1\right)\cdot gM_2 \bmod n_2\right)=M_1+M_2 \bmod n.$$

It is Additive [11]. The product of two cipher texts decrypts to the sum of their corresponding plaintexts. The product of cipher text with a plain text raising g decrypt to the sum of the corresponding plaintexts. The Paillier encryption scheme is an Additive Homomorphic encryption.

2.3.2. Multiplicative Homomorphic Encryption

The following property illustrates multiplicative homomorphism.

$$\mathrm{F}\left(M_0,M_1\right)=M_0*M_1 \tag{3}$$

The product of two cipher text decrypts to the product of their corresponding plaintext. RSA, ELGamal are the Multiplicative Encryption Schemes. (e, n) are public keys.

$$C_1=M_1^e \bmod n,\ C_2=M_2^e \bmod n \tag{4}$$

$$C_1\cdot C_2=\left(M_1\cdot M_2\right)^e \bmod n \tag{5}$$

3. Electronic Election Schemes

Paper-based voting systems have been the standard since the mid-19th century. In Elections like National or Local government elections, voters vote for a number of candidates. After voting the winning candidates are computed from the set of votes. Most of the citizens are registered as voters. The rest of them must register as voters. After the end of voting talliers count their tallies. In an e-voting the voters and talliers use the technology to speed up the voting process. First the voters enter their votes to the voting platform. Then the votes get trans-

mitted to a central machine that computes the winning candidate. Some information like the number of votes for a candidate, Number of votes in a particular city is displayed. Both the Voting platform Database and the Central Machine Database are encrypted using the encryption techniques. Central machine gets the encrypted, compressed database to improve the secrecy.

Implementation of Homomorphism and Data Packing

Secure e-voting can be achieved by using the homomorphic encryption. Homomorphism is an algebraic property, particularly useful in electronic voting schemes because it allows applying operations on sets of encrypted ballots without the need of decrypting them. It allows the votes to be tabulated before decryption and improving privacy. The recent groundbreaking work of homomorphic encryption shows how to maintain privacy of outsourced data. With homomorphic encryption scheme one can electronically access the outsourced data by the way of accessing it. For example, in additive homomorphic encryption, the product of two cipher texts is a third cipher text that encrypts the sum of the two original plaintexts. Let M_1, M_2 are the two messages. E (M) is the encryption of message m under encryption scheme [12].

$$\text{Ciphertext } C = E(M_1, M_2). \tag{6}$$

Electronic Voting Phase Flow Chart

The Flow Chart (**Figure 1**) displays the Electronic Voting Phase usage. Initially the User gives his votes to the Proposed re-encryption Scheme using Voting Platform. It is connected to the Internet and do all the authorizations. So that only eligible persons can vote. Multiple votes are not allowed. The Proposed Schemes work on Z^2 workspace. When it is applied once again it works on Z^4 workspace.

Encryption and Re-encryption both occupies more space. So the Data Packing is used to pack the encrypted data. The Packed data is once again unzipped to get back the data. Then the encrypted data is retrieved to find out the Winner Candidate. The Voting Platform has both the encryption and data packing methods. It is our Proposed scheme where the votes are encrypted, re-encrypted, zipped and sent through the Insecure Channel. When needed, it is decrypted and unzipped to find the winning candidate [13].

Vote Validation is done on the Voting Platform after getting the Vote. If it is not valid, it is not added. If it is valid it is taken as a valid vote. The verifier authority in the Voting Platform checks the credential of the voter and take care of it. This proposed voting scheme is secure since it satisfies eligibility, privacy, fairness, robustness, individual verifiability and universal verifiability. Any Participant or passive observer finally can check whether these calculations are correct. The voter can see whether his vote is valid or not and do one more time if wrong.

Since it is entered in the Voting Table the voter can verify that his vote is considered or not. But all votes remain secret. Only eligible voters are allowed to cast votes. The Proposed scheme has more security than the traditional voting scheme.

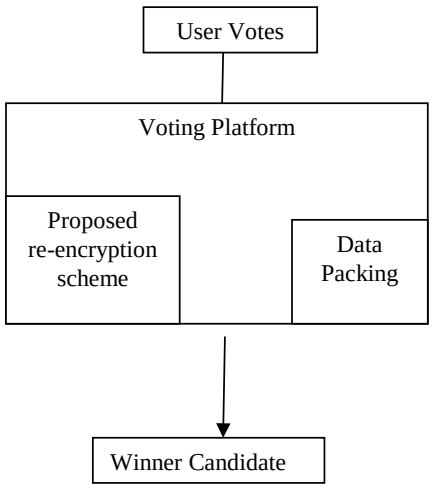

Figure 1. Flow chart of proposed system.

The operation can be performed on the underlying messages without revealing them [10]. For Election scheme additive encryption is most useful. Voting applications may use additive homomorphism to allow tallying to be done before decryption. With other forms of encryption, all the ballots are dissociated from their identifying pieces of information and then decrypted and tallied. If homomorphic encryption is used, the tallying can be done while the votes are still encrypted, and the final total can then be decrypted. This effectively hides the contents of the original ballots while providing a publicly computable tally. Basically a value is set to represent a particular candidate. If there are two candidates, then the summation gives the winner candidate. By comparing the resultant value and the present value the winning candidate is announced. A Unique identifier is used for each voter to avoid multiple voting. Paillier is additively homomorphic and computationally efficient to decrypt. One of the main advantages of Paillier encryption is that it is an additively homomorphic scheme [13].

Algorithm 1: Paillier Encryption Algorithm-Additive Homomorphic Algorithm

Step 1: Select two large primes, p and q.

Step 2: Calculate the product $n = p \times q$, such that gcd(n, $\Phi(n)$) = 1,where $\Phi(n)$ is Euler Function.

Step 3: Choose a random number g, where g has order multiple of n or

$$\gcd\left(\mathrm{L}\left(\mathrm{g}^{\lambda} \bmod n^2\right), n\right) = 1$$

where $\mathrm{L}(t) = (t-1)/n$ and $\lambda(n) = \mathrm{lcm}(p-1, q-1)$

Step 4: The public key is composed of (g, n), while the private key is Composed of (p,q, λ).

$\lambda = ((p-1)(q-1))/\gcd(p-1, q-1)$

Step 5: The Encryption of a message $M< n$ is given by $C = g^M r^n \bmod n^2$

Step 6: The Decryption of cipher text C is given by:

$$m = \left(\mathrm{L}\left(g^{\lambda} \bmod n^2\right) / \mathrm{L}\left(g^{\lambda} \bmod n^2\right)\right) \bmod n \tag{7}$$

Choose some M_1, $M_2 \in Z_n$ and $r_1, r_2 \in Z_n^*$

Let $C_1 = \mathrm{E}[M_1, r_1]$, $C_2 = \mathrm{E}[M_2, r_2]$,

$$C_3 = C_1 * C_2 \left(\bmod n^2\right) \tag{8}$$

$$= \mathrm{E}\left[M_1 + \lambda M_2 \left(\bmod n\right), r_3\right], \; r_3 \in Z_n^* \tag{9}$$

Let $C_1 = g^{M_i} y_1^n \bmod n^2$, $C_2 = g^{M_2} y_2^n \bmod n^2$.

$$C_1 C_2 \bmod n^2 = g^{M_i} y_1^n g^{M_2} y_2^n \tag{10}$$

$$= g^{M_1+M_2} y_1 y_2^n \bmod n^2 \tag{11}$$

It is a valid encryption of $M_1 + M_2$,

$$C_1 g^k \bmod n^2 = g^{M_i} y_1^n g^k = g^{M_i+k} y_1^n \bmod n^2 \tag{12}$$

$$\begin{aligned} C^{\lambda} \bmod n^{s+1} &= \left(g^M r^{ns}\right)^{\lambda} \bmod n^{s+1} \\ &= \left(g^M\right)^{\lambda} \left(r^{ns}\right)^{\lambda} \bmod n^{s+1} \\ &= (1+n)^{Mj\lambda} \left(r^{ns}\right)^{\lambda} \bmod n^{s+1} \\ &= (1+n)^{Mj\lambda} \bmod n^{s+1} \end{aligned} \tag{13}$$

It satisfies the Additive Homomorphic property

$$\prod_{i=1}^{l} E\left(M_i\right) = \sum_{i=1}^{l} M_i \tag{14}$$

Evaluation of compression on the resultant cipher texts is Data packing. Hence the compression technique can be evaluated on the output cipher texts, after all applications of the Evaluate algorithm have been completed.

4. Existing System

In the existing voting System (shown in **Figure 2**) Symmetric key encryption scheme using a private key (*K*). The Database which receives the voting interface data used the Key *K* for encryption. It is less secure than our system. The length and strength of the Cryptography keys are considered important. The keys used for encryption and decryption must be strong enough to produce strong encryption. They must be protected from unauthorized users and must be available when they are needed [14].

5. Proposed System

Table 1 denotes the List of Votes and its value. CMK value is 10^0, BMK is 10^1 and so on.

Table 2 has both the vote and encrypted value of the vote. The three steps of our proposed system [**Figure 3**] are: The system access control process that is to authenticate the voter to the election server, the voting process, and collecting data process. The Proposed system uses the Public Key encryption technique for encryption. In **Figure 2** the system uses Private Key (*SK*1) and Public Key (*PK*1) pair. The Database which gets information from the Voting interface uses the Key pair for encryption. Since it uses the Public key encryption schemes it is more secure. Once again, it is encrypted with the next Public Key encryption pair (*PK*2, *SK*2) and (*PK*3, *SK*3) [15].

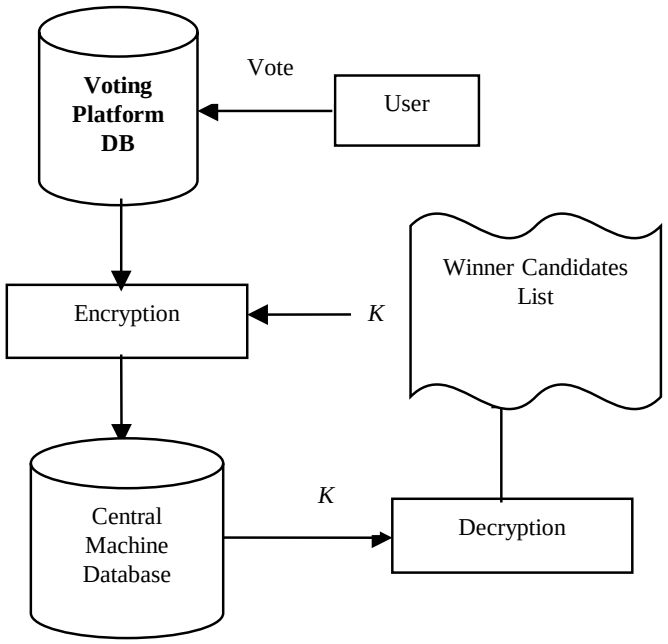

Figure 2. Existing encryption scheme.

Table 1. Voters list.

VOTERS NAME	Message M	CMK (10^0)	BMK (10^1)	AMK (10^2)	DMK (10^3)	EMK (10^4)
A	$M = 10^0 = 1$			*		
B	$M = 10^3 = 1000$				*	
C	$M = 10^1 = 10$		*			
D	$M = 10^0 = 1$			*		
E	$M = 10^2 = 100$	*				
R	$M = 10^4 = 10{,}000$					*
	TOTAL = 11,211	1	1	2	1	1

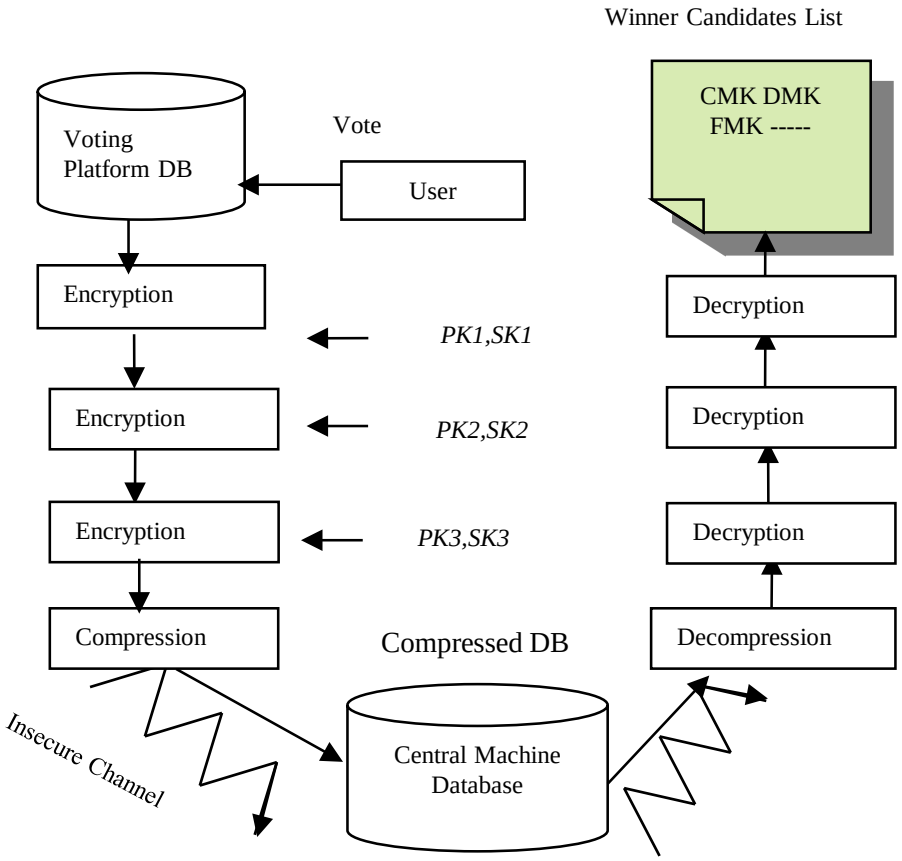

Figure 3. Proposed system of Electronic voting.

Table 2. Vote values vs encrypted value.

VOTER NAME	MESSAGE (M)	RANDOM VALUE r	ENCRYPTED VALUE C
ANBU	$M = 10^0 = 1$	660,820	818,466,297,129
BALA	$M = 10^3 = 1000$	468,581	2,439,962,883,397
CITRA	$M = 10^1 = 10$	387,219	2,286,056,462,773
DEVI	$M = 10^2 = 100$	35,116	2,732,935,861,399
ESWAR	$M = 10^2 = 100$	948,382	1,145,696,910,521
RAVI	$M = 10^4 = 10,000$	337,224	1,787,008,921,297

Z_n—Set of integers n;

Z_n^*—Set of integers coprime to n;

$Z_n 2^*$—Set of integers coprime to n^2;

N_v—Number of Voters;

N_c—Number of Candidates.

Vote is in numeric form and it gets encrypted using Paillier encryption. Each encryption needs a random number, so that same vote will be encrypted in different ways [17].

Vote messages considered are,

1st Candidate: 10^0.

2nd Candidate: 10^1.

N_c th Candidate: 10^{Nc-1}.

Only authorized people can vote in our system. The authorities calculate the encrypted vote, which is the product of all encrypted votes modulo n^2. The Voter interface with unique identifier checks and allows only the authorized voter can vote. Replacing a vote is also not allowed in our system. In cryptographic algorithm the procedures of Key generation, Encryption and Decryption is used. It also provides zero knowledge proofs that the contents of the encrypted vote check for the validity [16]. After voting has closed, the voting authorities use the Encryption Schemes. Then it is sent to the Main Database called the Central Voting Database.

Encryption

$$\mathrm{E}\left(M_i\right) = C_i = g^{M_i} r_i^n \bmod n^2 \tag{15}$$

$\mathrm{E}(M_1) = C_1 = 818466297129$, $\mathrm{E}(M_2) = C_2 = 2439962883397$, $\mathrm{E}(M_3) = C_3 = 2286056462773$
$\mathrm{E}(M_4) = C_4 = 2732935861399$, $\mathrm{E}(M_5) = C_5 = 1145696910521$, $\mathrm{E}(M_6) = C_6 = 1787008921297$

Decryption

$\mathrm{D}\left(C_i\right) = \left(\mathrm{L}\left(\mathrm{g}^{\lambda} \bmod n^2\right) \Big/ \mathrm{L}\left(g^{\lambda} \bmod n^2\right)\right) \bmod n$, $\lambda = 833228$, $n = 1669039$, $\mathrm{D}(C_1) = 1$, $\mathrm{D}(C_2) = 1000$, $\mathrm{D}(C_3) = 10$, $\mathrm{D}(C_4) = 100$, $\mathrm{D}(C_5) = 100$, $\mathrm{D}(C_6) = 1000$

Winner Candidate

$$\prod_{i=1}^{l} E\left(M_i\right) = \sum_{i=1}^{l} M_i \tag{16}$$

$$\begin{aligned}\sum_{i=1}^{l} C_i \bmod n^2 &= \left(818466297129 \times 2439962883397 \times 2286056462773 \times 2732935861399\right. \\ &\quad \left. \times 1145696910521 \times 1787008921297\right) \bmod 278569118352 \\ &= 96747685543\end{aligned}$$

$$\mathrm{M} = \left(\mathrm{L}\left(g^{\lambda} \bmod n^2\right) \Big/ \mathrm{L}\left(g^{\lambda} \bmod n^2\right)\right) \bmod n = \sum_{i=1}^{l} M_i \bmod n = 11211 \tag{17}$$

The winner candidate is 2×10^2 = AMK. **Table 1** shows the List of Votes given by the voters. **Table 2** shows Voters value and its Encrypted Values. This verifies the sum of all plain votes is equivalent to the encrypted value of all the votes [17]. The Voters select a certain winner from the candidate list. Each and every vote of the voter is encrypted. Finally, some talliers count the votes and declare the voting result. E(M) encrypts the Message M, D(C) decrypts the Ciphertext. In this system, tallying is performed without revealing any vote. It also uses the Data packing technique to make the encrypted data to be compressed. So it occupies minimum space than the earlier one. Any hacker who tries to access the voter database data cannot easily find the Key Pairs. Three pairs of keys are used for decryption. After Unpacking the database, it is decrypted to see the Winner Candidate List [17].

6. Benefits of the Proposed System

The following are the benefits of the Proposed System:

1. It has the homomorphic property which is useful for voting. It is semantically secure.
2. It is more efficient than others E-Voting system. It allows the voter to vote for his/her own personal computer (PC) without any extra cost and effort.
3. Voters feel confident that their votes are counted.
4. It is very simple to use, hence it needs only the basic requirements such as; PC, internet connection and a valid roof.

7. Experiments and Analysis

Table 3 shows the speed of Encryption on Paillier and Elgamal algorithms. It is noted in seconds. When compared to Elgamal encryption, Paillier is somewhat very faster. When the Key size increased the seconds are also increased.

Table 4 shows the various key sizes of Paillier and Elgamal.

This shows the symmetric and asymmetric key size comparison.

In Proposed system the Existing system is revised. The encrypted Voters DB is given to the Central machine by doing Compression. So it is more secured and Compact one. Then it is decrypted to get the Winner Candidate. **Table 3** specifies the various Key sizes for the Symmetric and Asymmetric encryption for the proposed work of

the Homomorphic election scheme. **Table 4** denotes the various Keys used in the Algorithm and the time to do the encryption in its scheme. **Figure 4** shows the Security performance of Symmetric and Asymmetric Encryption algorithms with varying key sizes.

Figure 5 shows the time taken for Addition and Multiplication for various Key Sizes. When compared to Elgamal the Paillier encryption is faster. In the existing system, it gets the votes from the voting platform and it stores in the database [18]. That database is encrypted using the Key. Then it is transferred to the Central Machine Database. The central Machine database is decrypted to get the Winner Candidates List. When the size of the Voter list is very high, then Paillier method is very much useful.

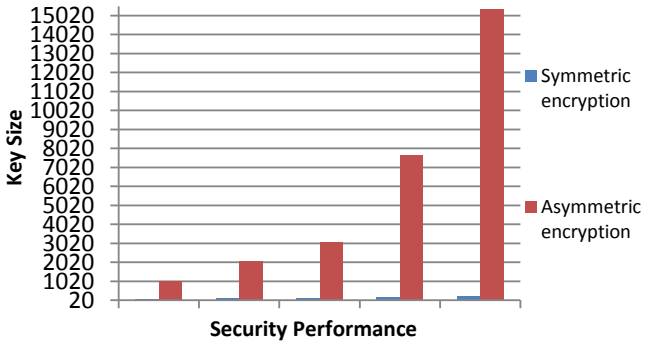

Figure 4. Security performance vs key size of symmetric and asymmetric encryption.

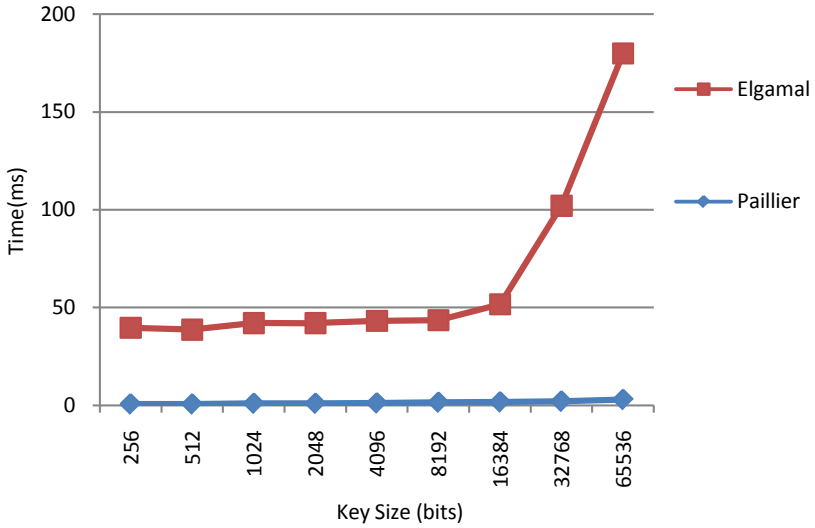

Figure 5. Encryption time of paillier vs elgamal.

Table 3. Paillier and elgamal encryption time.

Key	Paillier Encryption(sec)	Elgamal Encryption(sec)
256	0.7	39
512	0.7	38
1024	1	41
2048	1	41
4096	1.2	42
8192	1.5	42
16384	1.5	50
32768	2	100
65536	2.9	177

Table 4. Various key sizes for encryption.

Key Size (Symmetric) bits	Key Size (Asymmetric) (bits)
80	1024
112	2048
128	3072
192	7680
256	15,360

8. Applications of Homomorphic Encryption

The cloud has more storage capabilities and computing power. A major application of FHE is to cloud computing [13]. One solution for providing secure cloud computing on untrusted public clouds is the use of homomorphic encryption: a method of encryption which allows computations on encrypted data, without the need to fully decrypt the data on the cloud. The technique of homomorphic encryption also lends itself to other important cryptographic applications such as multi-party computation [15]. Solutions of Homomorphic encryption dedicated to numerous application contexts like secret sharing schemes, threshold schemes, zero-knowledge proofs, oblivious transfer commitment schemes, anonymity, privacy, electronic voting, electronic auctions, lottery protocols, protection of mobile agents, multiparty computation, mix-nets, watermarking or finger printing protocols and so forth.

9. Conclusion

We demonstrated the use of Paillier homomorphic encryption in the Electronic Voting scheme. The RSA, Elgamal public key cryptosystem which exhibit multiplicative homomorphic property cannot be used in electronic voting. Furthermore, we compared symmetric key encryption and Asymmetric key encryption for electronic voting and compared their key sizes [19]. Our work demonstrates the homomorphic encryption schemes which are faster than other encryption methods. We also made use of data packing techniques to efficiently store data in electronic voting. So the combination of homomorphic encryption using Paillier cryptosystem and data packing can significantly improve the storage and processing performance in the electronic voting scheme.

References

[1] Fontaine and Galand (2009) A Survey of Homomorphic Encryption for Nonspecialists *Journal of Information Security*.

[2] Colin, A.R.B. (2004) Multiplicative Homomorphic E-Voting. *Progress in Cryptology-INDOCRYPT* 5*th International Conference on Cryptology in India*, 20-22.

[3] Bellare, M. and O'Neill, A. (2013) Semantically-Secure, Functional Encryption: Possibility Results, Impossibility Results and the Quest for a General Definition. *Cryptology and Network Security*, 218-234. http://dx.doi.org/10.1007/978-3-319-02937-5_12

[4] Boneh, D., Goh, E. and Nissim, K. (2005) Evaluating 2-DNF Formulas on Cipher Texts. *Theory of Cryptography Conference*, *TCC*'2005, *Lecture Notes in Computer Science*, 325-341.

[5] Boneh, D. and Lewi, K. (2015) Key Homomorphic PRFs and Their Applications. *IACR Cryptology ePrint Archive*, 220.

[6] Garg, S., Gentry, C., Halevi, S. and Zhandry, M. (2014) Fully Secure Attribute Based Encryption from Multilinear Maps. *Cryptology ePrint Archive*, *Report*. http://print.iacr.org/2014/622

[7] Hayes, B. (2012) Alice and Bob in Cipher Space American Scientist. http://www.americanscientist.org/issues/pub/2012/5/

[8] Katz, J. and Thiruvengadam, A. (2015) Feasibility& Infeasibility of Adaptively Secure Fully Homomorphic Encryption. *IACR Cryptology ePrint Archive*, 280.

[9] Aguilar, M.C., *et al.* (2011) Improving Additive and Multiplicative Homomorphic Encryption Schemes Based on Worst-Case Hardness Assumptions. *IACR Cryptology ePrint Archive*, 607.

[10] Schoenmaker, B. and Tuyls, P. (2006) Efficient Binary Conversion for Paillier Encrypted Values. *Advances in Cryptology-EUROCRYPT*'06, Springer, Berlin, 522-537.

[11] Qiong, S.M. and Zhang, M.W. (2015) Efficient Public Key Encryption with Equality Test Supporting Flexible Authorization. *IEEE Transactions on Information Forensics and Security*, **10**.

[12] Martenvan Dijk, S.D. (2015) Onion ORAM: A Constant Bandwidth and Constant Client Storage ORAM (without FHE or SWHE) IACR Cryptology ePrint Archive.

[13] Ravindran, S. and Kalpana, P. (2013) Data Storage Security Using Partially Homomorphic Encryption in a Cloud. *International Journal of Advanced Research in Computer Science and Software Engineering*, **3**, 603-606.

[14] Han, W.W. (2014) A Provably Secure Public Key Encryption Scheme Based on Isogeny Star. *The International Arab Journal of Information Technology.*

[15] Jing, Y., Fan, M.Y., Wang, G.W. and Kong, Z.Y. (2014) Simulation Study Based on Somewhat Homomorphic Encryption. *Journal of Computer and Communications.*

[16] Naresh, V.S. and Murthy, N.V.E.S. (2015) A New Two-Round Dynamic Authenticated Contributory Group Key Agreement Protocol Using Elliptic Curve Diffie-Hellman with Privacy Preserving Public Key Infrastructure. *Sadhana*, **40**, 2143-2161.

[17] (2011) HSR Hochschule fur Technik Rapperswil Homomorphic Tallying with Paillier Cryptosystem. *Publicationes Mathematicae Debrecen*, 479-496.

[18] Damgard, I., Jurik, M. and Nielson, J. (2010) A Generalization of Paillier's Public Key System with Applications to Electronic Voting. *International Journal of Information Security*, **9**, 371-385. http://dx.doi.org/10.1007/s10207-010-0119-9

[19] Goldwasser, S. and Micali, S. (1984) Probabilistic Encryption. *Journal of Computer and System Sciences*, **28**, 270-299. http://dx.doi.org/10.1016/0022-0000(84)90070-9