International Journal on Recent Researches In Science, Engineering & Technology
(Division of Computer Science and Engineering)
A Journal Established in early 2000 as National journal and upgraded to International journal in 2013 and is in existence for the last 10 years. It is run by Retired Professors from NIT, Trichy. It is an absolutely free (No processing charges, No publishing charges etc) Journal Indexed in JIR, DIIF and SJIF.
Research Paper
Available online at: www.jrrset.com

ISSN (Print) : 2347-6729
ISSN (Online) : 2348-3105

Volume 5, Issue 1, January 2017.

JIR IF : 2.54
DIIF IF : 1.46
SJIF IF : 1.329

A Homomorphic Crypto System for Electronic Election Schemes

[1] K.Arun Prasad [2]K.Pushpavalli
[1] Associate Professor, Department of Information Technology, Agni College Of Technology, Chennai, India
[2] Assistant Professor, Department of Information Technology, Agni College Of Technology, Chennai, India

Abstract

This paper investigates the applications of homomorphic encryption systems in electronic voting schemes. We make use of paillier cryptosystem which exhibits additive homomorphic property. The other homomorphic cryptosystems RSA and Elgamal are not considered, since they exhibit only multiplicative homomorphic property. 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 schemes.

Key Words: Electronic Voting, Public key, Homomorphic, Paillier, Datapacking,

1 Introduction

The goal of encryption is to ensure confidentiality of data in communication and storage processes. This technique aggregates privacy and security for data processing in unreliable environments. Recently, its use in constrained devices led to consider additional features, such as the ability to delegate computations to untrusted computers. For this purpose, we would like to give the untrusted computer only an encrypted version of the data to process. The computer performs the computation on this encrypted data, hence without knowing any thing on its real value. Finally the system sends back the results, and the receiver decrypts it. For coherence, the decrypted result has to be equal to the intended computed value if performed on the original data. For this reason, the encryption scheme has to present a particular structure. Rivest et al. Proposed in 1978 to solve this issue through homomorphic encryption [1]. 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 [2]. Homomorphic encryption is an encryption which allows specific types of computations to be carried out on cipher text and generate an encrypted result which decrypted matches the result of operations performed on the plaintext. There are several efficient Partially homomorphic cryptosystems and a number of Fully homomorphic cryptosystems. Based on the applications and the basis homomorphism can be used to perform computations securely. High computational and communication complexity involved in using homomorphic encryption for the practical applications. The idea of homomorphic encryption has been around for about 30 years. 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 cipher texts 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. Suppose the cloud is to search a huge encrypted database for the handful of records that match an encrypted search term. Homomorphic encryption ensures that the server has no idea what the search term is or which records match it. As a consequence, however, it has no choice but to send back information on every record in the database. The user's computer can decrypt that information to see which records matched and which didn't, but then it's assumed much of the computational burden that it was trying to offload to the cloud in the first place.

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 (RSA [2], Elgamal [3]) or sum of the plaintext (Goldwasser-Micali [4] and Paillier [5]) Brickell and Yacobi pointed out in [6] 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 [7] invented a system of provable security encryption, with which unlimited number of additions but only one multiplication can be performed [8].

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 [9]. 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 Caesar cipher is simple, but not secure. Private Key encryption schemes can be used. But it uses 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. Encrypt algorithm encrypts the Plaintext into Ciphertext using the key. Decryption algorithm decrypts the Ciphertext uses the Key. A Homomorphic public key encryption scheme has four algorithms. The usual KeyGen, Encrypt and Decrypt and an additional algorithm Evaluate.
= (KeyGen, Encrypt, Decrypt, Evaluate)
Evaluate takes as input a public key pk, cipher text C= (C1,C2,…Cn) 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 (SHE)

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 [6].
DEC(SK, Eval(PK, F, C1,…, Cn)) = F(M1,…,Mn);
f can be a 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. Craig Gentry's technique is from a boot trappable 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=(C1,C2,..Cn) , fully homomorphic encryption should allow to output a cipher text that encrypts FC)=f(C1,C2,..Cn) for the function F, as long as that function can be efficiently computed. No information about (C1,C2,..Cn) or F(C1,C2,..Cn), or any intermediate plaintext values, should leak. The inputs, output and intermediate values are always encrypted. A fully homomorphic encryption scheme that 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

F (M0, M1) = M0 + M1

It is additive then the product of two cipher texts decrypt to the sum of their corresponding plaintexts.

D(E(M1,r1).E(M2,r2) mod n2) = M1 + M2 mod n. The product of a cipher text with a plain text raising g decrypt to the sum of the corresponding plaintexts,

D(E(M1,r1).gM2 mod n2) = M1 + M2 mod n.

The Paillier encryption scheme is an Additive Homomorphic encryption [9].

2.3.2 Multiplicative Homomorphic encryption

F(M0,M1) = M0 X M1

It is multiplicative, then the product of two cipher texts decrypts to the product of their corresponding plaintexts.

C1 = M1e mod n, C2 = M2e mod n

C1.C2 = (M1.M2) e mod n

RSA, ELGamal are the Multiplicative Encryption Schemes.

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 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 transmitted 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 are also 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.

3.1 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. In electronic voting schemes, 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. In this work, we focus on the orthogonal question of providing integrity/authenticity 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 m1, m2 are the two messages. E (M) is the encryption of message m under encryption scheme.
Let $^{\delta}$ be an operation.
ztext C = E (M1 $^{\delta}$ M2). [9]
The operation $^{\delta}$ can be performed on the underlying messages without revealing them. 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. Paillier is additively homomorphic and computationally efficient to decrypt. One of the main advantages of Paillier encryption is that it is additively homomorphic scheme [10].

Paillier Algorithm

Step 1: Select two large primes, p and q.
Step 2: Calculate the product n = p x q, such that gcd(n,Φ(n)) = 1, where Φ(n) is
Euler Function.
Step 3: Choose a random number g, where g has order multiple of n or
gcd (L(gλ mod n2), n) = 1 where L(t)= (t-1) / n and λ(n)=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). λ = ((p-1)(q-))/gcd(p-1,q-1)
Step 5: The Encryption of a message M< n is given by c=gMrn mod n2
Step 6: The Decryption of cipher text c is given by:
m= (L(gλ mod n2) / L(gλ mod n2)) mod n
If we choose some M1, M2 ∈ Zn and r1, r2 ∈ Zn* and
Let
C1 = E[M1, r1], C2 = E[M2, r2], [11]
C3 = C1 * C2 (mod n2) = E[(M1 + M2 (mod n), r3], for some r3 ∈ Zn*
Let C1 = g M1y1n mod n2 , C2 = gM2y2 n mod n2.
C1C2 mod n2 = gM1y1 ngM2y2 n = gM1+M2y1y2 n mod n2 is a valid encryption of M1 + M2,
C1gk mod n2 = gM1y1 ngk = gM1+ky1n mod n2
C mod ns+1 = (gMrns) mod ns+1
= (gM) (rns) mod ns+1
= (1 + n)Mj (rns) mod ns+1
= (1 + n)Mj mod ns+1,

It satisfies the Additive Homomorphic property. 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

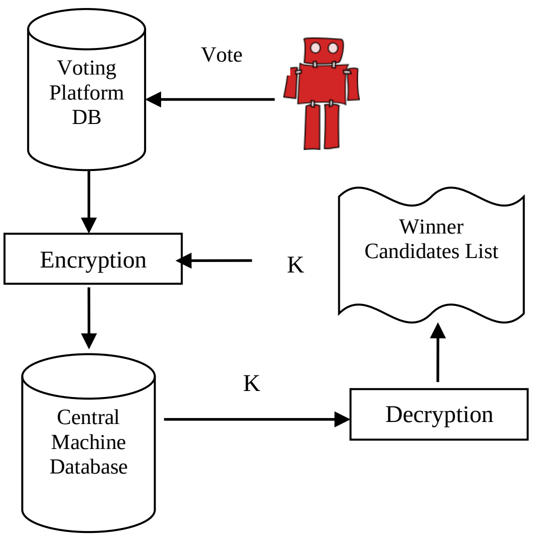
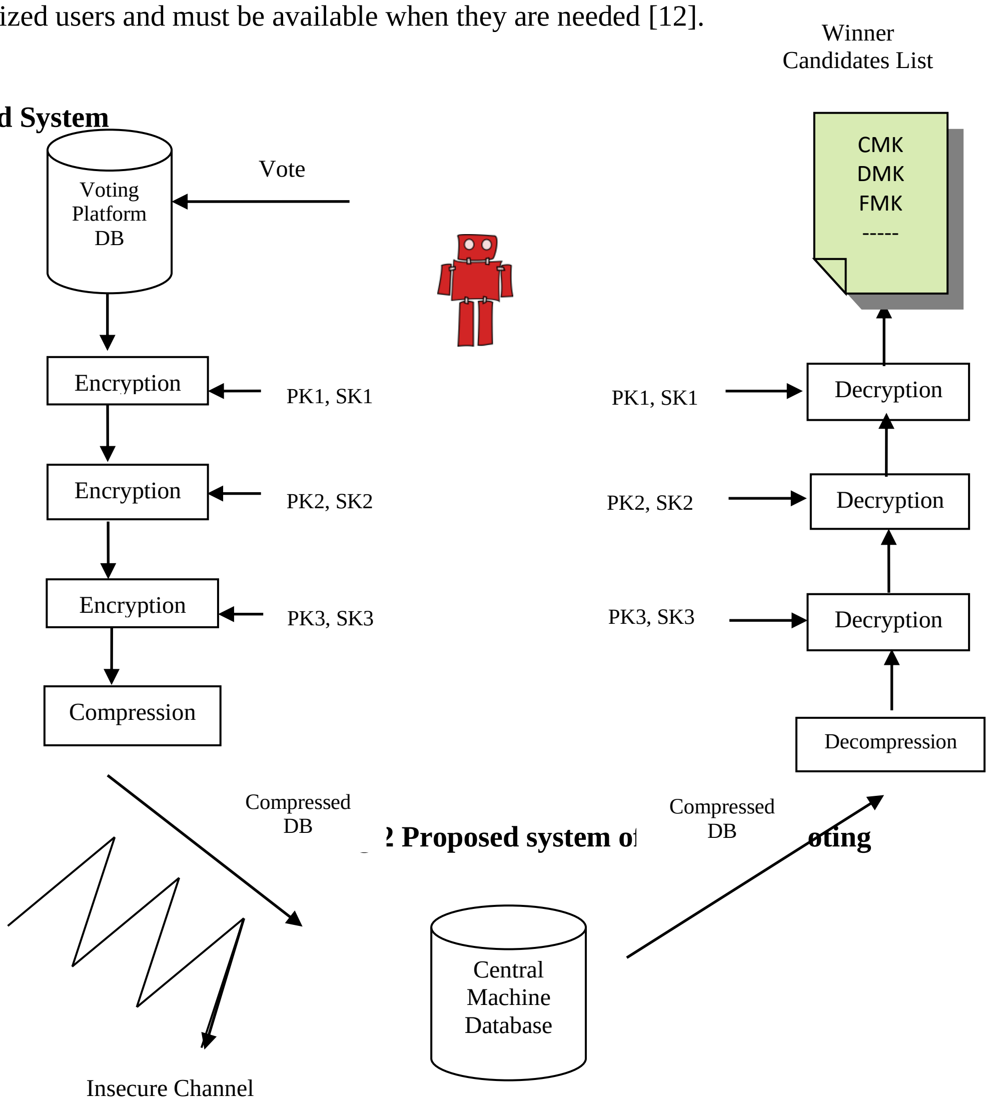


Fig.1 Existing Voting System

In Existing voting System 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 an important mechanism. 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 [12].

Proposed System



2 Proposed system o oting

The three steps of our proposed system 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 Fig2 the system uses Private Key (SK1) and Public Key (PK1) 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 (PK2, SK2) and (PK3, SK3) [13]. Since it uses the Paillier encryption scheme, it's time to encrypt is somewhat less. Since it is an additively homomorphic system. Only authorized people can vote in our system. The Voter interface checks and allows only the authorized voter can vote. It checks for the duplicate vote also. No voter can vote more than once. 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 are checked for the validity. After voting has closed, the voting authorities use the Encryption Schemes. Then it is sent to the Main Database called the Central Voting Database.

VOTER NAME	Message M	AMK (10^0)	BMK (10^1)	CMK (10^2)	DMK (10^3)	EMK (10^4)
ANBU	$M = 10^0 = 1$	*				
BALA	$M = 10^3 = 1000$				*	
CITRA	$M = 10^1 = 10$		*			
DEVI	$M = 10^0 = 1$	*				
ESWAR	$M = 10^2 = 100$			*		
RAVI	$M = 10^4 = 10000$					*
	TOTAL=11211	2	1	1	1	1

Table1. Voters List

VOTER NAME	MESSAGE (M)	RANDOM VALUE r	ENCRYPTED VALUE C
ANBU	$M = 10^0 = 1$	660820	818466297129
BALA	$M = 10^3 = 1000$	468581	2439962883397
CITRA	$M = 10^1 = 10$	387219	2286056462773
DEVI	$M = 10^2 = 100$	35116	2732935861399
ESWAR	$M = 10^2 = 100$	948382	1145696910521
RAVI	$M = 10^4 = 10000$	337224	1787008921297

Table2. Vote value VS encrypted value

Encryption

$E(M_i) = C_i = g^{M_i}.r_i^n \bmod n^2$ [14]

$E(M_1) = C_1 = 818466297129$, $E(M_2) = C_2 = 2439962883397$, $E(M_3) = C_3 = 2286056462773$

$E(M_4) = C_4 = 2732935861399$, $E(M_5) = C_5 = 1145696910521$, $E(M_6) = C_6 = 1787008921297$

Decryption

$D(C_i) = (L(g^{\lambda} \bmod n^2) / L(g^{\lambda} \bmod n^2)) \bmod n$, $\lambda = 833228$, $n = 1669039$ [14]

$D(C_1) = 1$, $D(C_2) = 1000$, $D(C_3) = 10$, $D(C_4) = 100$, $D(C_5) = 100$, $D(C_6) = 10000$

Winner Candidate

$\prod_{i=1}^{l} E(Mi) = \sum_{i=1}^{l} Mi$ [15]

$\prod_{i=1}^{l} C_i \bmod n^2$ = (818466297129 * 2439962883397 * 2286056462773 * 2732935861399 * 1145696910521 * 1787008921297) mod 278569118352
= 96747685543
M = (L($g^{\lambda} \bmod n^2$) / L($g^{\lambda} \bmod n^2$)) mod n = $\sum_{i=1}^{l} Mi \bmod n$ = 11211
Table 1. Shows the List of Votes given by the voters.
Table 2. Shows the Voters value and its Encrypted Values.

This verifies the sum of all plain votes is equivalent to the encrypted value of all the votes [16]. The Voters select a certain winner from the candidate list. Each and every vote of the voter is encrypted. Only valid votes are considered. Finally,some talliers count the votes and declare the voting result. 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. Since it is encrypted and compressed one it is more tedious to access. Three pairs of keys are used for decryption. After Unpacking the database, it is decrypted to see the Winner Candidate List. E-Voting is also an excellent mechanism that doesn't require geographic proximity of the voters. For example, soldiers abroad can participate in elections by voting online. A scalable and accurate e-voting scheme satisfying a potential candidate for voting over a public network such as the Internet.

5.1 Benefits of Proposed System

1. It has the homomorphic property which is useful for voting [17].
2. It is semantically secure.
3. 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.
4. Voters feel confident that their votes are counted.
5. It is very simple to use, hence it needs only the basic requirements such as; PC, internet connection and a valid proof [18].

6 Experiments and Analysis

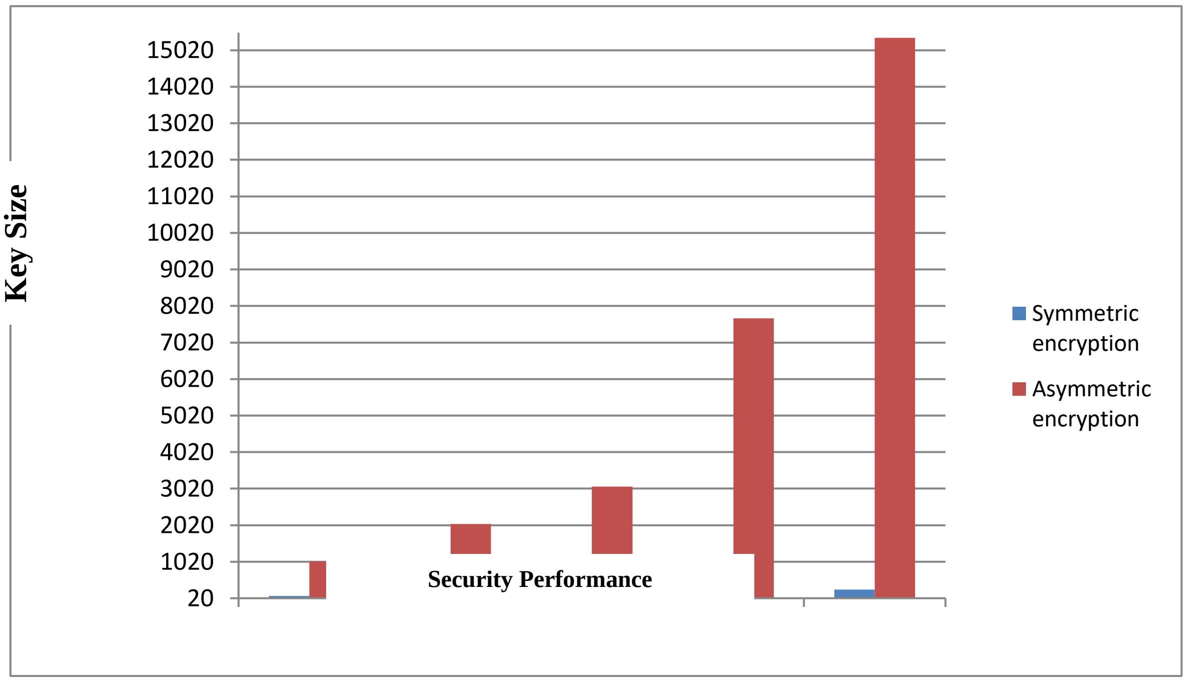
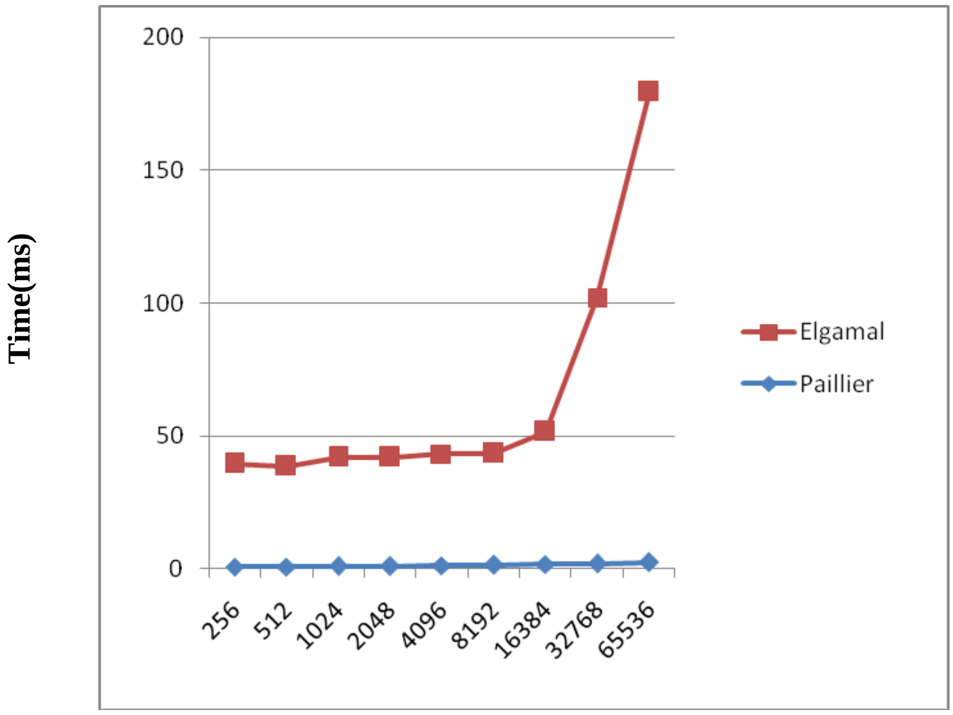


Fig3. Security Performance Vs Key size of Symmetric and Asymmetric Encryption

This Fig3 shows the Security performance of Symmetric and Asymmetric Encryption algorithms with varying key sizes. In Asymmetric encryption when the key size increases the security level is also increased.



Key Size (bits)

Fig4. Encryption time of Paillier Vs Elgamal with various Key sizes

Fig4 shows the time taken for Addition and Multiplication for various Key Sizes. When compared to Multiplication the Paillier encryption is faster. In Existing system, it gets the votes from the voting platform and it stores in the database. That database is encrypted using Key [19].

Then it is transferred to the Central Machine Database. The Central Machine database is decrypted to get the Winner Candidates List. 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.

Key Size (Symmetric)	Key Size (Asymmetric)
80	1024
112	2048
128	3072
192	7680
256	15360

Table3. Various Key sizes for encryption

Table3 specifies the various Key sizes for the Symmetric and Asymmetric encryption for the proposed work of the Homorphic election scheme [20].

Key	Paillier Encryption	Elgamal Encryption
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

Table4. Paillier Vs Elgamal Encryption time

Table4 denotes the Key variance used in the Algorithm and it's time to do the encryption in its scheme.

7 Applications of Homomorphic Encryption

The cloud has more storage capabilities and computing power. A major application of FHE is to cloud computing [21]. Cloud computing allows users with limited storage capacity to securely outsource their data to a remote server. Meanwhile, it also allows the server to reliably perform computations over the data. 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.

Fully Homomorphic Encryption can be used to query a search engine, without revealing what is being searched for. The technique of homomorphic encryption also lends itself to other important cryptographic applications such as multi-party computation [22].

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[23]. A practical Fully homomorphic encryption solution would see widespread use by cloud service providers, significantly hardening cloud security and making cloud storage a more viable option for consumers. Homomorphic encryption is one of the most exciting new research topics in cryptography, which promises to make cloud computing perfectly secure [24]. With it, a Web user would send encrypted data to a server in the cloud, which would process it without decrypting it and send back a still-encrypted result.

8. Conclusion:

We demonstrated the use of Paillier homomorphic encryption in 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. 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 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] Rivest, R.L., Adleman, L., Dertouzos, "On data banks and privacy homomorphisms" in Foundations of Secure Computation. Academic Press (1978) 169-180

[2] ElGamal, T. "A public key cryptosystem and a signature scheme based on discrete logarithms" in IEEE Transactions on Information Theory 31(4) (1985) 469-472

[3] S. Goldwasser and S. Micali "Probabilistic encryption" in Journal of Computer and System Sciences 28(2) (1984) 270-299

[4] E. Brickell and Y. Yacobi, "On privacy homomorphisms," in Advances in Cryptology (EUROCRYPT '87), vol. 304 of Lecture Notes in Computer Science, pp. 117–126, Springer, New York, NY, USA, 1987.

[5] Rivest, R., Shamir, A., Adleman, L."A method for obtaining digital signatures and public key cryptosystems" Communications of the ACM 21(2) (1978) 120-126

[6] Fontaine and F. Galand," A Survey of Homomorphic Encryption for Nonspecialists", in Journal of Information Security, 2009, 1, pp. 50

[7] Sakshi Duggal, Vandana Mohindru, "A Comparative Analysis of Private Key Cryptography Algorithms: DES, AES and Triple DES", International Journal of Advanced Research in Computer Science and Software Engineering Volume 4, Issue 6, June 2014

[8] B. Schoenmakers and P. Tuyls. "Efficient binary conversion for Paillier encrypted values", In Advances in Cryptology - EUROCRYPT '06, volume 4004 of LNCS, pages 522-537, Berlin, 2006. Springer-Verlag.

[9] Ron Rivest, "Lecture Notes 15 : Voting, Homomorphic Encryption, Computer and Network Security" October 29, 2002

[10] O. Baudron, P. Fouque, D. Pointcheval, "Practical multi-candidate election system" In Proc. of the ACM Symp. on Principles of Distributed Computing, Philadelphia, 2001.

[11] Sha Ma, Qiong Huang, Mingwu Zhang "Efficient Public Key Encryption With Equality Test Supporting Flexible Authorization" in ieee transactions on information forensics and security, vol. 10, NO. 3, march 2015

[12] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim "Evaluating 2-DNF formulas on ciphertexts" in Theory of Cryptography Conference, TCC'2005, volume 3378 of Lecture Notes in Computer Science, pages 325-341. Springer, 2005.

[13] Yang, Jing, Mingyu Fan, Guangwei Wang, and Zhiyin Kong. "Simulation Study Based on Somewhat Homomorphic Encryption" in Journal of Computer and Communications (2014): 109.

[14] Melchor, Carlos Aguilar, et al. "Improving Additive and Multiplicative Homomorphic Encryption Schemes Based on Worst-Case Hardness Assumptions" IACR Cryptology ePrint Archive 2011 (2011): 607.

[15] Aditya, Riza, Boyd, Colin," Multiplicative Homomorphic E-Voting" In Canteaut, A & Viswanathan, K (Eds.) Progress in Cryptology - INDOCRYPT 2004. 5th International Conference on Cryptology in India., 20-22 December 2004, Chennai, India

[16] SrinivasDevadas,MartenvanDijk "Onion ORAM: A Constant Bandwidth and Constant Client Storage ORAM (without FHE or SWHE)." IACR Cryptology ePrint Archive 2015: 5 (2015)

[17] Paillier, P."Public-key cryptosystems based on composite degree residuosity classes" in 18th Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic. Volume 1592 of Lecture Notes in Computer Science., Springer (1999)223-238

[18] Hayes B. "Alice and Bob in cipherspace." American Scientist. 2012;100(5). Available at: http:// www.americanscientist.org/issues/pub/2012/5/

[19] Jonathan Katz and Aishwarya Thiruvengadam "Feasibility and Infeasibility of Adaptively Secure Fully Homomorphic Encryption" IACR Cryptology ePrint Archive 2015:280 (2015)

[20] Dan Boneh and Kevin Lewi "Key Homomorphic PRFs and Their Applications" IACR Cryptology ePrint Archive 2015:220 (2015)

[21] Sunanda Ravindran and Parsi Kalpana, "Data Storage Security Using Partially Homomorphic Encryption in a Cloud" International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013, pp. 603-606.

[22] M. Bellare and A. O'Neill. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general de_nition. In CANS 13, LNCS 8257, pages 218{234. Springer, November 2013. (Page 6.)

[23] D. Boneh, A. Sahai, and B. Waters. Functional encryption: De_nitions and challenges. In TCC 2011, LNCS 6597, pages 253{273. Springer, March 2011. (Pages 1 and 6.)

[24] S. Garg, C. Gentry, S. Halevi, and M. Zhandry. Fully secure attribute based encryption from multilinear maps. Cryptology ePrint Archive, Report 2014/622, 2014. http://eprint.iacr.org/2014/622. (Pages 1 and 4.)