International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 5 Issue 1, January-2016

Robust Electronic Voting System using Homomorphic Encryption Protocol and Zero-Knowledge Proof

Dr. Mahmood Khalel Ibrahem

Al-Nahrain University - College of Information Engineering, Iraq

ABSTRACT

Electronic voting is the voting process held over electronic media. For such a sensitive issue like election, security is one of the main concerns, such as authentication, confidentiality and integrity. Simplicity is also necessary to ensure the participation of common people. Besides security and simplicity, other issues that need to be considered such as reliability, convenience, flexibility, mobility and cost.

In this paper, a prototype of a web-based robust electronic voting system is presented with two new techniques; the first technique is a new zero knowledge authentication protocol based on Diffie-Hellman (D-H) key exchange algorithm, to ensure a mutual authentication between the election authority server and the voters. The second technique is homomorphic encryption scheme to encrypt all the votes and perform the calculation of the votes without revealing any information about it to ensure the security of the votes and maintain the confidentiality. The proposed system provides secure voting over the Internet and maintains the requirements of the voting process.

Keywords: D-H Key exchange algorithm, Homomorphic Encryption, Zero-Knowledge Proof.

1. INTRODUCTION

Electronic voting, as the name implies, is the voting process held over electronic media, i.e. computers and communication technologies. For such a sensitive issue like election, security is one of the main concerns, but simplicity is also necessary to ensure the participation of common people. Besides security and simplicity, there may be some other issues that need to be considered. In that respect, we need to specify all such issues or properties that the election system must possess. A well-defined protocol is necessary to take care of all such requirements. Computers and facilitation of internet has spread its wings far and wide providing easy access everywhere. Thus in this context, holding election over the Internet seems logical from many different points of view. Relief from long queues, minimal chance of voting error and verifiability stands in favor of an electronic election. Recent improvements in network security have made it possible to design election system with high class security, but it is also important that carefully designed protocols and continuous improvements of the implementations are necessary to keep them out of reach from the network threats. From that point of view, an implementation of secure Internet voting protocol appears to be another application of cryptography and network security [1].
The most efficient voting protocols could be categorized by their approaches into two major types; schemes using blind signatures and schemes using homomorphic encryption. The suitability of each of these types varies with the conditions under which it is to be applied. In the schemes using blind signatures, the voter obtains a token, a blindly signed message unknown to anyone except himself. Next, the voter sends his token together with his vote anonymously over a secure channel. These schemes require voter's participation in more rounds. In homomorphic encryption schemes, the voter cooperates with the authorities to construct an encryption of his vote. Due to the homomorphic property, an encryption of the sum of the votes is obtained by summing the encrypted votes of all voters. Subsequently, the result of the election is computed from the sum of the votes which is jointly decrypted by the authorities [2].

In this paper, a theoretical background of zero-knowledge proof is presented, and modified Diffie-Hellman algorithm for mutual zero-knowledge is discussed. A prototype of secure electronic voting system using zero-knowledge with modified D-H algorithm and homomorphic encryption is presented with examples. Finally analysis of the proposed system is discussed.

2. CHARACTERISTICS OF SECURE E-VOTING

When designing an electronic polling system, it is essential to consider ways in which the polling tasks can be performed electronically without sacrificing voter privacy or introducing opportunities for fraud. In order to determine whether a system performs these tasks well, it is useful to develop a set of criteria for evaluating system performance. The following is one set of desirable characteristics for electronic polling systems which incorporates the characteristics of most systems described in the electronic voting literature [3]:

A. **Accuracy:** A system is accurate if; it is not possible for a vote to be altered, it is not possible for a validated vote to be eliminated from the final tally, and it is not possible for an invalid vote to be counted in the final tally.
B. **Democracy:** A system is democratic if; it permits only eligible voters to vote and, it ensures that each eligible voter can vote only once.
C. **Privacy:** A system is private if; neither election authorities nor anyone else can link any ballot to the voter who cast it, and no voter can prove that he or she voted in a particular way.
D. **Verifiability:** A system is verifiable if anyone can independently verify that all votes have been counted correctly.
E. **Convenience:** A system is convenient if it allows voters to cast their votes quickly, in one session, and with minimal equipment or special skills.
F. **Mobility:** A system is mobile if there are no restrictions (other than logistical ones) on the location from which a voter can cast a vote.

3. ZERO KNOWLEDGE AUTHENTICATION

Zero knowledge authentication protocols are one of the most trusted authentication protocols. In zero knowledge authentication, the claimant does not reveal anything that might endanger the confidentiality of the secret. The claimant proves to the verifier that he/she knows a secret, without revealing it. The interactions are so designed that they cannot lead to revealing or guessing the secret. After exchanging messages, the verifier only know that the claimant does or does not have the secret, nothing more. The result is a yes/no situation, just a single bit of information [4].

A Zero-Knowledge Proof (ZKP) is a proof of some statement which reveals nothing other than the veracity of the statement. The word “proof” here is not used in the traditional mathematical sense. Rather, a “proof”, or equivalently a “proof system”, is an interactive protocol by which one party (called the *prover*) wishes to convince another party (called the *verifier*) that a given statement is true. In ZKP, the *prover* proves that he/she knows a secret without revealing it [5].

ZKP model of computation defined as an interactive proof system (P,V), where P is a *prover* and V is a *verifier*. Protocol (P,V) is for proving a language membership statement for a language over $\{0.1\}$. Let L be a language over $\{0,1\}^*$, for a membership instance $x \in L$, P and V must share the common input x, Proof instance is denoted as $(P,V)(x)$ [5].

P and V are linked by a communication channel over which they exchange a sequence, called proof transcript a_1, b_1, a_2, b_2... a_n, b_n. Proof transcript interleaves *prover’s* transcript and *verifier’s* transcript. Each element a_i, b_i exchanged is bounded by polynomial in $|x|$ and Proof instance $(P,V)(x)$ must terminate in polynomial time in $|x|$. Upon completing the interaction, the output of the protocol should be of form $(P,V)(x) \in \{Accept, Reject\}$ representing *V’s* acceptance or rejection of *P’s* claim that $x \in L$ [6].

Three properties are expected from a zero-knowledge proof [7,8]:

A. Completeness: An interactive proof (protocol) is complete if, given an honest *prover* and an honest *verifier* (that is, one following the protocol properly), the protocol succeeds with overwhelming probability (i.e., the verifier accepts the prover’s claim).
B. Soundness: An interactive proof (protocol) is sound if there exists an expected polynomial time algorithm M with the following property: if a dishonest *prover* (impersonating P) can with non-negligible probability successfully execute the protocol with V, then M can be used to extract from this prover knowledge (essentially equivalent to P’s secret) which with overwhelming probability allows successful subsequent protocol executions.
C. Zero-knowledge: a protocol has zero-knowledge property if it is simulatable in the following sense; there exists an expected polynomial-time algorithm (*simulator*) which can produce, upon input of the assertion(s) to be proven but without interacting with the real prover, transcripts indistinguishable from those resulting from interaction with the real prover.

3.1 FIAT-SHAMIR ZKP PROTOCOL

In cryptography, the Fiat-Shamir identification scheme is a type of interactive zero-knowledge proof. Like all zero-knowledge proofs, the Fiat-Shamir scheme allows one party (prover), to prove to another party (verifier), that he possesses secret information without revealing to him what that secret information is [9].

In Fiat-Shamir protocol, a trusted third party selects two large prime numbers p and q to calculate the value of ($n = p.q$). The value of n is announced to the public; the values p and q are kept secret. Alice the prover choose a secret number ($1 < s < n-1$) and calculate ($v = s^2 \ mod \ n$). She keeps s as private key and register v as her public key with the third party. Figure-1 illustrates the steps of the protocol. Alice, the prover and Bob the verifier performs the following procedure [9]:

1) Alice, the prover, chooses a random number r (commitment) such that ($1 \leq r \leq n-1$), she then calculate the value of ($x = r^2 \ mod \ n$), x called the witness.
2) Alice sends x to Bob as the witness.
3) Bob, the verifier, sends the challenge c to Alice. The value of c is [$0, 1$].
4) Alice calculates the response ($y = rs^c$), where s is Alice's private key.
5) Alice sends the response (y) to Bob to prove that she knows her private key (she claims to be Alice).
6) Bob calculate y^2 and xv^c. If these two values are congruent, then Alice either knows the value of s (honest) or she calculated y in some other way (dishonest).
[$y^2 \ mod \ n = (rs^c)^2 \ mod \ n = r^2s^{2c} \ mod \ n = r^2(s^2)^c \ mod \ n = xv^c \ mod \ n$]
7) Repeat steps (1-6) several times with value of c equal to 0 or 1. The prover must pass the test in each round to be verified.

It is clear that a dishonest claimant has a 50 percent chance of fooling the verifier and passing the test (by predicting the value of the challenge) in other words, bob assigns a probability of ½ to each round of the test. If the probability decreases to $(1/2)^{20}$ or $9.54*10^{-7}$ then it is highly improbable that Alice can guess correctly 20 times. Unfortunately, the enhancement of the protocol will increase the computation cost [10, 11].

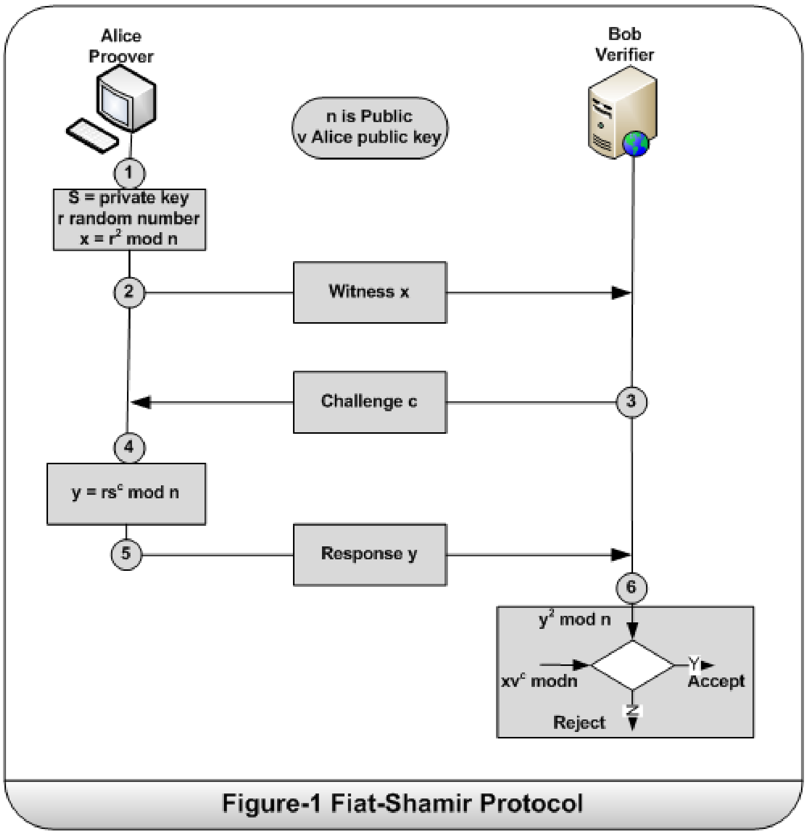


Figure-1 Fiat-Shamir Protocol

3.2 MODIFIED D-H ALGORITHM FOR ZKP

A new ZKP proposed by [12] based on Diffie-Hellman key exchange algorithm in the sense that both parties (the prover and the verifier) exchange non secret information without revealing secret information to get one identical secret key. This means that the prover can prove to the verifier that he knows the secret [12].

The verifier needs to prove to the prover that he is honest by sending his reply R_1 together with encrypted R_1, then the verifier decrypt R_1' by his key and match R_1 and R_1', if they matched then the verifier is honest. The prover (Alice) needs to prove to the verifier (Bob) that she knows a secret by calculating the key (K) and resend Bob's reply (R_2) to the verifier (Bob) encrypted with the generated secret key (K). Bob will encrypt his own reply (R_2) with the generated secret key (K) and match the two encrypted information, if matched then Alice is verified, otherwise it is rejected.

The proposed algorithm has been developed to resists the man-in-the-middle attack. For more details refer to [12]. Figure-2 shows the procedure of the proposed ZKP protocol. The protocol performed as follows:

1. Alice (the prover) chooses a large random number *x*, such that $0<x<p$ and calculate $R_1 = g^x \bmod p$.
2. Alice sends R_1 to Bob.
3. Bob (the verifier) chooses another large random number *y*, such that $0<y<p$ and calculate $R_2 = g^y \bmod p$, $K_{Bob} = (R_1)^y \bmod p$, *and* $C_1 = E(K_{Bob}, R_2)$.
4. Bob sends ($R_2 \mid C_1$) to Alice.
5. Alice, calculates $K_{Alice} = (R_2)^x \bmod p$, decrypt ($R_2' = D(K_{Alice}, C_1)$) and verify ($R_2 = R_2'$) . If they matched then she proceeds; otherwise the verifier is dishonest.
6. Alice encrypt ($C_2 = E(K_{Alice}, R_1|R_2)$ and send it to Bob.
7. Bob decrypt C_2 to get R_1' and R_2'
8. Bob verify ($R_1 = R_1'$); if they are equal then Alice is verified (Accepted), otherwise it is a dishonest prover (rejected).

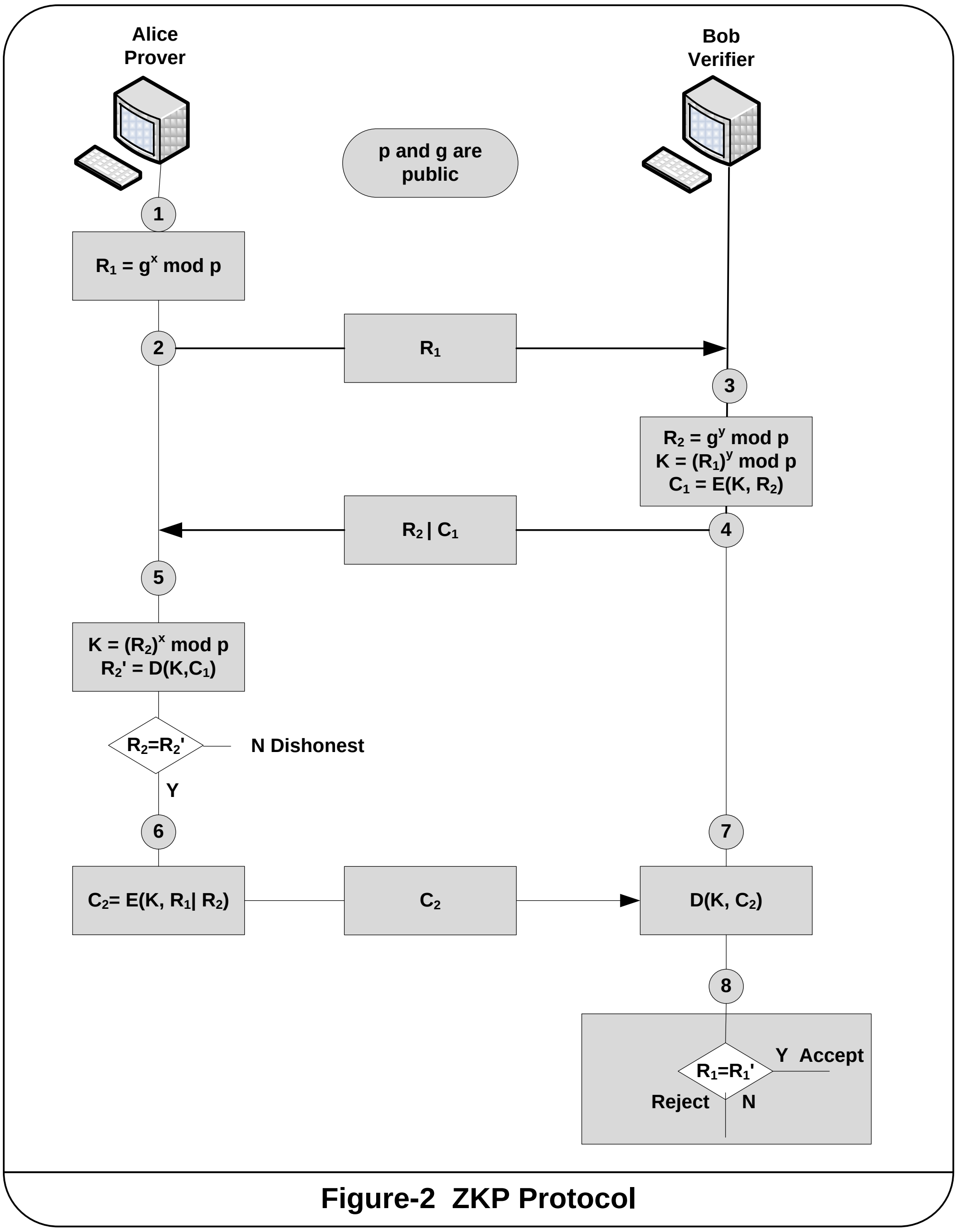


Figure-2 ZKP Protocol

4. HOMOMORPHIC ENCRYPTION

Homomorphic encryption is special type of cryptography where a specific algebraic operation is performed on the plaintext and another (possibly different) algebraic operation is performed on the ciphertext. In homomorphic encryption, the sum of two encrypted values is equal to the encrypted sum of the values. This encryption method is useful in e-voting applications in the sense that the sum of a group of encrypted values (votes) is verified without revealing their contents. By use of this method namely, the tally of the ballots is verified without revealing what those ballots are.
The encryption algorithm E() is homomorphic if; given E(x) and E(y), one can obtain E(x ⊥y) without decrypting x and y for some operation (⊥). The principal guiding factor in this scheme is the homomorphic property given by [13, 14];

$$E(m1) \perp E(m2) = E(m1 \perp m2); \qquad (1)$$

Where; *E* represents the encryption done on messages *m1* and *m2*. *E(m1) ⊥ E(m2)*, is a calculation in a group *G*, whereas *E(m1 ⊥ m2)* is a calculation in a group *H*. The `⊥', is a group operator corresponding to each group, and may be different for *G* and *H*. Advantage of the homomorphic property is; the votes can be counted and verified in group G without knowledge of the individual votes. After the election, the encrypted votes are combined into a single, encrypted, quantity. The authorities then decrypt this tally, in the group *H*. Due to the homomorphic property, this quantity should be equal to the quantity resulting from the decryption of each of the individual votes in group *G*. In this way, tallying is done without learning the individual values of the votes. Thus, anonymity is maintained [15].

The operation (⊥) could be multiplication (*) and/or addition (+). Fully homomorphic systems uses both operations, multiplications and additions (i. e. ⊥ = * and +). A cryptosystem which supports both addition and multiplication (thereby preserving the ring structure of the plaintexts) would be far more powerful. Using such a scheme, any circuit could be homomorphically evaluated, effectively allowing the construction of programs which may be run on encryptions of their inputs to produce an encryption of their output. Since such a program never decrypts its input, it could be run by an untrusted party without revealing its inputs and internal state. The existence of an efficient and fully homomorphic cryptosystem would have great practical implications in the outsourcing of private computations, for instance, in the context of cloud computing [16].

However, fully homomorphic schemes are impractical for many applications, because ciphertext size and computation time increase sharply as one increases the security level. To obtain 2^k (*k* is an integer from some range of candidates) security against known attacks, the computation time and ciphertext size are high-degree polynomials in *k*. Gentry's Ph.D. thesis [17], provides additional details.

Another homomorphic schemes are partially Homomorphic encryption schemes, which uses addition operation only (+). The homomorphic property is preserved such that;

$$E(m1) + E(m2) = E(m1 + m2); \qquad (2)$$

Where, *E* represents the encryption done on messages *m1* and *m2*. *E(m1) + E(m2)*, is a calculation in a group *G*, whereas *E(m1 + m2)* is a calculation in a group *H*. In partially homomorphic schemes, the homomorphic property is preserved and works as follows; votes are encrypted individually as they enters and added together to form summation counters for each candidate [18]. At the end of the election, the encrypted summation is decrypted to reveal the final results of the votes. Hence, no one can relate the votes to their voters. Partially homomorphic scheme is implemented in this research to maintain efficiency in terms of time and space.

5. SYSTEM DESIGN

To achieve the system robustness, flexibility and resistance to potential change, the popular three-tier (layer) architecture is deployed in the proposed system. The architecture is composed of three layers: the user interface layer, the application logic layer and the database layer. The three-layer architecture aims to solve a number of recurring design and development problems, and make the application development work more easily and efficiently [19].

The interface layer in the three-layer architecture offers the user a friendly and convenient entry to communicate with the system while the application logic layer performs the controlling functionalities and manipulating the underlying logic connection of information flows; finally, the data modeling job is conducted by the database layer, which can store, index, manage and model information needed for this application [19, 20]. Figure-3 illustrates the architecture of the proposed system. Figure-4 illustrates system flowchart.

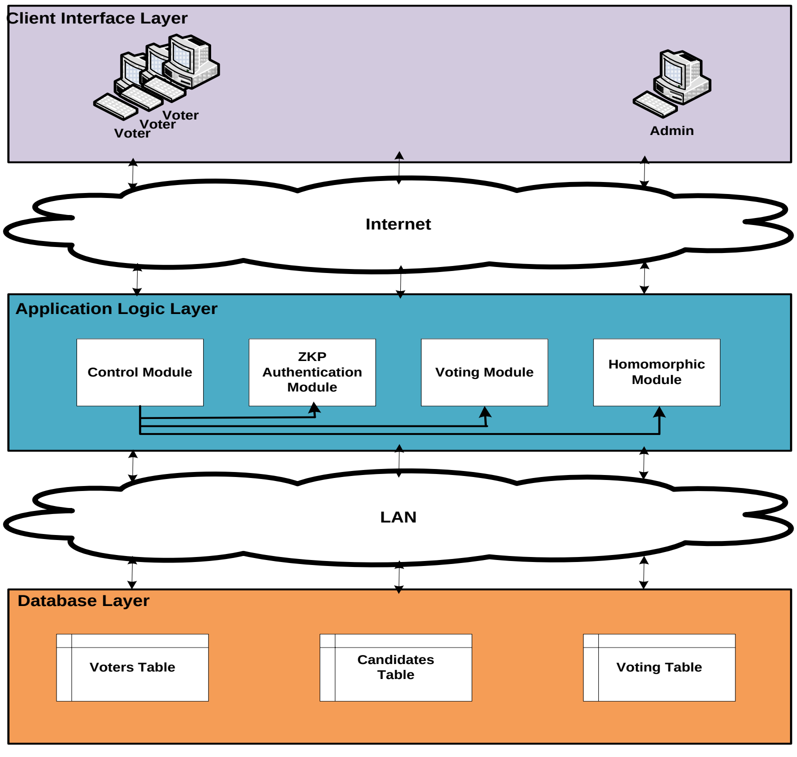


Figure-3 Secure E-Voting System Archetecture

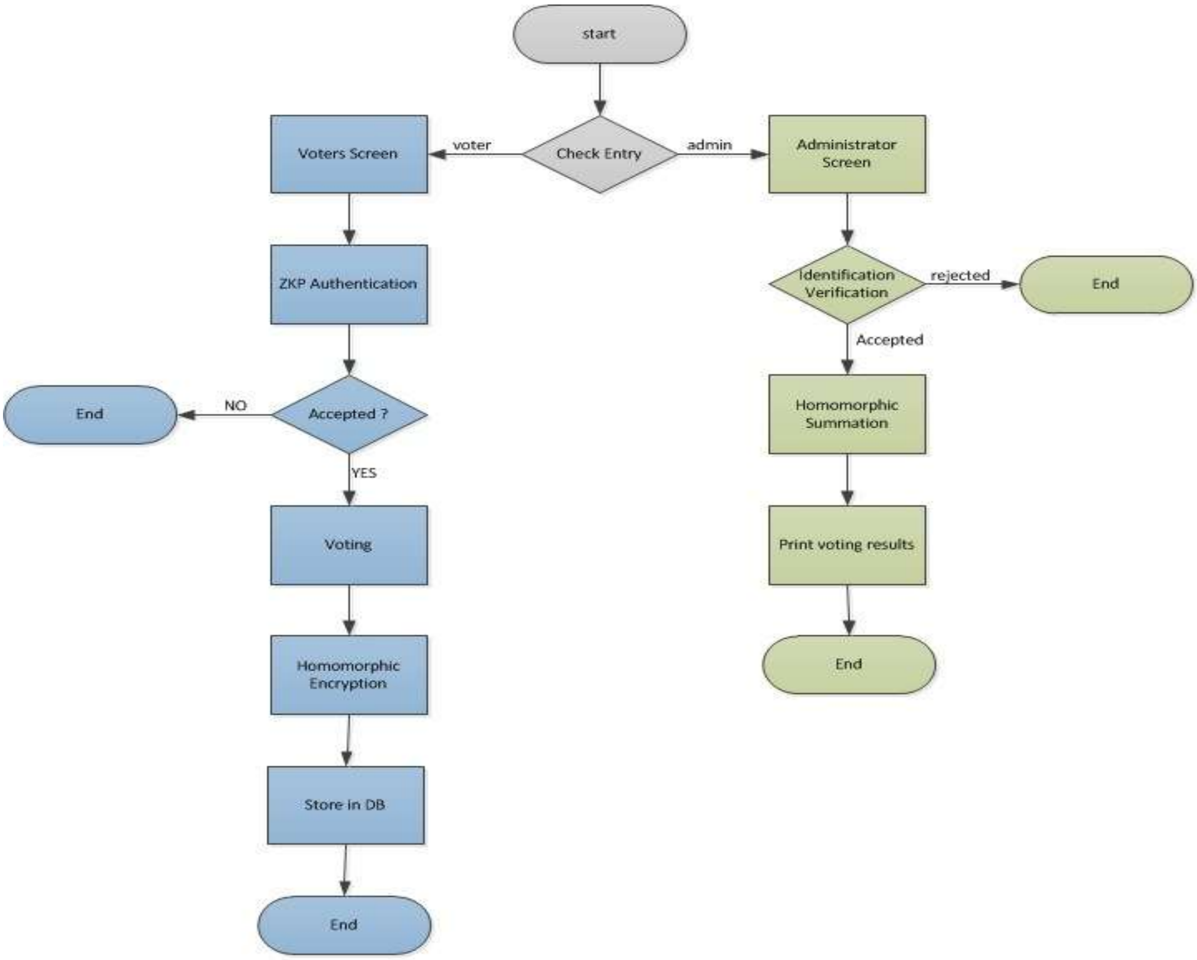


Figure-4 Control Module Flowchart

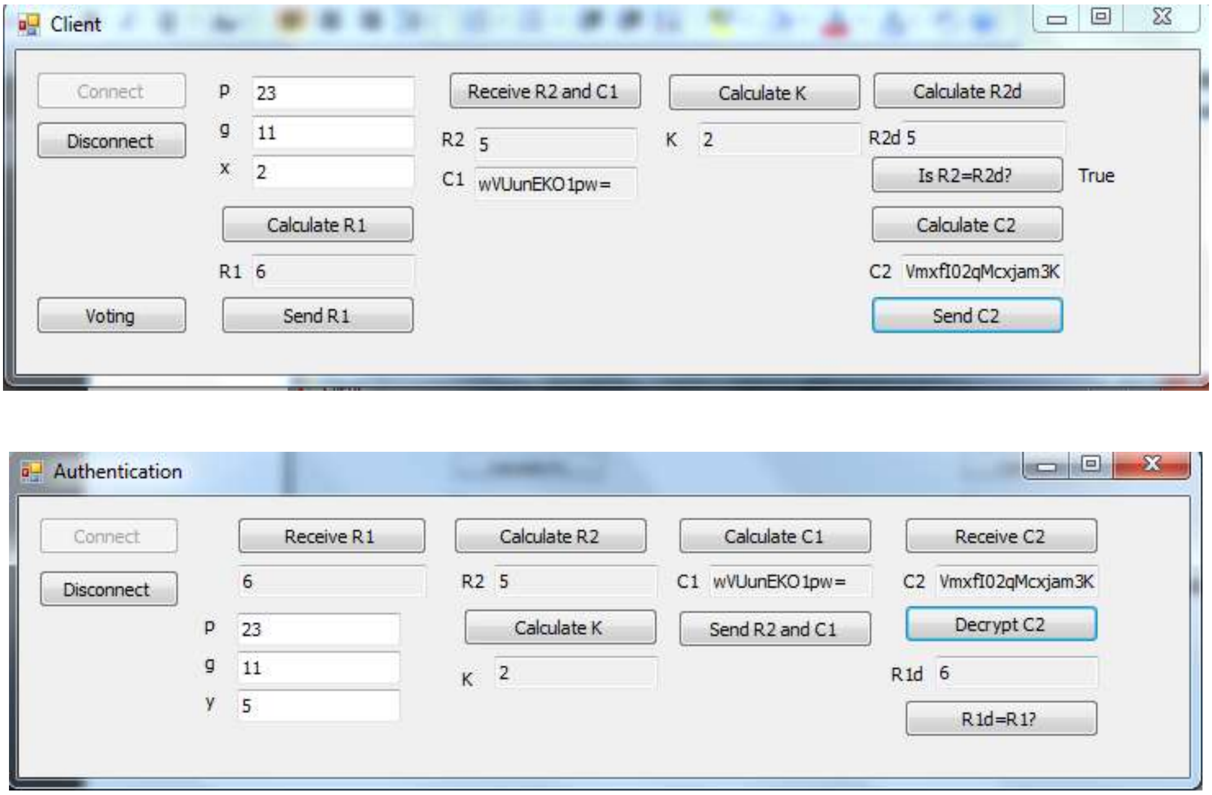
5.1 USER INTERFACE LAYER

The first layer is the user interface layer. This layer manages the input/output data and their display. With the intention of offering greater convenience to the user, the system is prototyped on the Internet. The users are allowed to access the system by using any existing web browser software. The user interface layer contains HTML components needed to collect incoming information and to display information received from the application logic layers. The voters communicate with the web server via application protocols, such as HTTP and SSL, sending requests and receiving replies. Clients of the system are voters and administrator, as they described below.

A. Voter

Voter has mainly a user interface which communicates with application layer. In this interface, there are two textboxes, and a vote button. The voter is supposed to enter his name and national number that has been previously delivered to him through election offices together with his randomly generated secrete number in a sealed card. The National number entered by the voter is compared with the stored national number in the database. Invalid name or national number will be rejected and a second chance for the voter will be given. Valid data will lead the voter to second stage.

The voter enters his secret number and communicates with the honest verifier through ZNP protocol to establish a secure mutual authentication and prove to each other that they are illegible. Figure-5 illustrates communication screen between voter and the verifier. On successful completion of this step, candidate names are dynamically retrieved from the database and displayed to the voter. The user has an option to log out or complete the vote. If the user completes the vote, the (has voted) field in the database is updated so that the user cannot log in once more.



p 23
g 11
x 2
R1 6
R2 5
C1 wVUunEKO1pw=
K 2
R2d 5
True
C2 VmxfI02qMcxjam3K

6
p 23
g 11
y 5
R2 5
K 2
C1 wVUunEKO1pw=
C2 VmxfI02qMcxjam3K
R1d 6

Figure-5: Voter-Verifier Communication Screen

B. Administrator

Administrator has a special username and password for login to system. After he/she has logged into the system, he will get new screen which allow him to add, update or delete voters or candidates. Details of information are shown in figure-6.

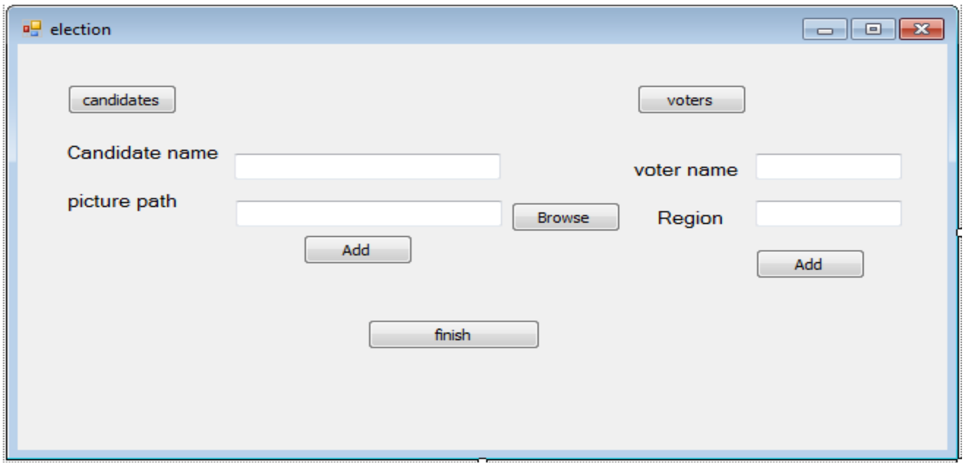


Figure-6 Administrator Screen

5.2 APPLICATION LOGIC LAYER

The application logic layer is the middle layer, which bridges the gap between the user interfaces and the underlying database, hiding technical details from the users. Components in this layer receive requests coming from the interface layer and interpret the requests into apropos actions controlled by the defined work flow in accordance with certain pre-defined rules. Application logic layer consists of a controller module and three functional modules. The control module controls the flow of functions execution and transferring required information between them and the database layer. The functional modules are;

A. Zkp Authentication Module

Each voter must be authenticated to vote through zero knowledge proof, using the proposed protocol discussed earlier in (3.2). A claimant (voter) and the verifier (server) will interact communicating messages to prove to each other that they are illegible. During this process a secret key is generated in both sides securely. The interactions are so designed that they cannot lead to revealing or guessing the secret. After exchanging messages the voter will be either accepted and proceeds to the voting stage, or rejected and halt the system.

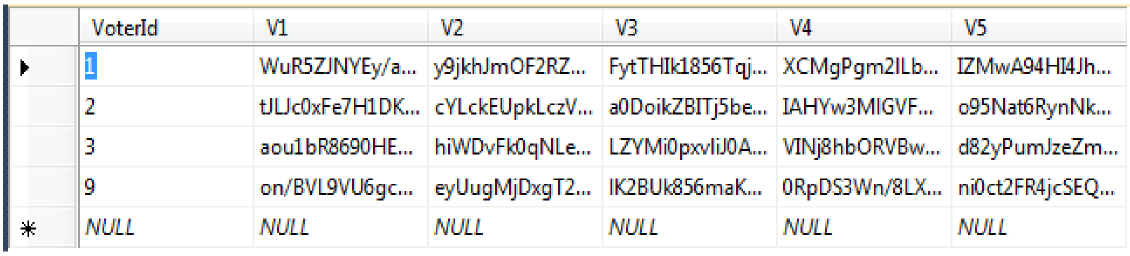
B. Voting Module

The second module is the voting module which is responsible for conducting the process of voting. Authenticated voter can selects his candidates in his/her region from a pull down menu and cast his vote. The votes will be encrypted and stored in the database using homomorphic module.

C. Homomorphic Module

The third module is the homomorphic encryption module which is responsible for encrypting the votes and storing them in the database as illustrated in figure-7. In this module the sum of a group of encrypted votes is verified without revealing those encrypted votes using homomorphic protocol, after all the voting procedure completes**.** Homomorphic protocol is discussed in (4).

At the end of the election process, this module will be executed again to process the votes and calculate the results homomorphically (i. e. using the homomorphic properties) to produce the final election results.



	VoterId	V1	V2	V3	V4	V5
▶	1	WuR5ZJNYEy/a...	y9jkhJmOF2RZ...	FytTHIk1856Tqj...	XCMgPgm2ILb...	IZMwA94HI4Jh...
	2	tJLJc0xFe7H1DK...	cYLckEUpkLczV...	a0DoikZBITj5be...	IAHYw3MIGVF...	o95Nat6RynNk...
	3	aou1bR8690HE...	hiWDvFk0qNLe...	LZYMi0pxvliJ0A...	VINj8hbORVBw...	d82yPumJzeZm...
	9	on/BVL9VU6gc...	eyUugMjDxgT2...	IK2BUk856maK...	0RpDS3Wn/8LX...	ni0ct2FR4jcSEQ...
✳	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure-7: encrypted Votes table

5.3 DATABASE LAYER

The database layer is responsible for modeling and storing information needed for the system and for optimizing the data access. Data needed by the application logic layer are retrieved from the database, and then the computation results produced by the application logic layer are stored back in the database. Since data are one of the most complex aspects of many existing information systems, it is essential in structuring the system. Both the facts and rules captured during data modeling and processing are important to ensure the data integrity.

Database consists of four tables. The first table called “voter”, stores the voters data. Second table called "candidate Table" used to store the candidates data. The third table called "Votes table, used to store encrypted votes as illustrated in figure 7 above. The fourth table is called "regions" and used to store election regions.

6. SYSTEM ANALYSIS

In this paper, two techniques are used to implement a secure E-voting system and maintain its requirements; the first technique is a new zero knowledge protocol used to ensure secure mutual authentication between the voter (prover) and the election authority server (verifier) with the addition to exchange key securely. This technique is used to ensure that both parties are legal and hence prevent election fraud and maintain democracy which allows legal voters to vote once only.

The second technique is homomorphic encryption protocol which ensures the confidentiality of the votes, in the sense that neither election authorities nor anyone else can link any ballot to the voter who cast it, and no voter can prove that he/she voted in a particular way.

The proposed e-voting system prevents incomplete ballots and hence can be easily verified by comparing total voters with total votes as shown in the final report of the system.

The proposed system is a web-based application and hence it is flexible and mobile that allows voters to cast their votes quickly, in one session, and with minimal equipment and skills with no restrictions (other than logistical ones) on the location from which a voter can cast a vote.

With the addition to the previous characteristics, the system provides dramatical reduction in the monetary cost, effort, and time required compared with traditional voting systems.

7. CONCLUSIONS

This paper describes a complete design scheme, and evaluation of a provable secure remote e-voting system. The proposed system consists of three phases; the first phase is the registration phase that focuses on how the citizen can be register in the scheme with high level of security. In this phase the system assure many of security properties such as transparency, trust, eligibility, and privacy.

The second phase is the voting phase, which is based on homomorphic cryptography techniques. In this phase we assure many of security properties such as privacy, confidentiality.

The third phase is the counting phase, it consists of two steps counting and auditing, that count and audit the votes for each candidate .In this phase encrypted votes are counted using homomorphic properties to assure accuracy, verifiability, reliability, universal verifiability and fairness.

Convenience, flexibility and mobility are also maintained throughout the design of the system.

Perhaps the most important contribution of this work is strong evidence that, contrary to conventional voting systems, secure electronic voting is possible and even feasible. We are optimistic about the future of voting systems constructed, using principled techniques.

REFERENCES

[1] Ivan Damgard, et al, "***The theory and Implementation of Electronic Voting System***", Advances in Information Security Vol. 7 Springer 2003, ISBN 1-4020-7301-1.

[2] Yong-Sork, Her et al., "***Ballot-Cancellation Scheme In E-Voting System***", E-Government Workshop ’05 (eGOV05), Brunel University, UK, September, 2005.

[3] Kapali Viswanathan, "***Towards Robustly Secure Electronic Voting Systems***", the 2nd Annual IIT Kanpur Hacker's Workshop (IITKHACK05). India, 2005.

[4] Endre Bangerter, et al**, "*On the Design and Implementation of Efficient Zero-Knowledge Proofs of Knowledge***", Proceedings of the 2nd ECRYPT Conference on Software Performance Enhancement for Encryption and Decryption and Cryptographic Compilers (SPEED-CC'09), Berlin, Germany, Octobre 2009.

[5] Mohr, Austin "***A Survey of Zero-Knowledge Proofs with Applications to Cryptography***". http://austinmohr.com/work/files/zkp.pdf, cited January, 2010.

[6] Maurer Ueli, "***Unifying Zero-Knowledge Proofs of Knowledge***", Africacrypt, pp: 272–286, 2009.

[7] Krantz, Steven G., **"*Zero Knowledge Proofs*",** AIM Preprint Series, Volume 10 No. 46, July, 2007.

[8] Michael Backes and Dominique Unruha, "***Computational Soundness of Symbolic Zero-Knowledge Proofs***", Journal of Computer Security, Vol. 18, No. 6, pp: 1077-1155, 2010.

[9] Forouzan, Behrouz A. "***Cryptography and Network Security***", McGraw-Hill, Int. Ed. 2008.

[10] Fischer, Michael J., "***Cryptography and Computer Security***", Lecture Notes, Department of Computer Science, Yale University, March 29, 2010**.**

[11] Kizza, Joseph M, "***Feige-Fiat-Shamir ZKP Scheme Revisited***", International Journal of Computing and ICT Research, Vol. 4, No. 1, pp: 9-19, June 2010.

[12] Mahmood Khalel Ibrahem, "***Modification of Diffie–Hellman Key Exchange Algorithm for Zero Knowledge Proof***", International Conference on Future Communication Networks, pp 147-152, IEEE 2012.

[13] Craig Gentry, "***Fully Homomorphic Encryption Using Ideal Lattices***", Proc. Of the 41st ACM Symposium on Theory of Computing (STOC), pp: 169-178, NY, USA, 2009.

[14] Louis J. M. Aslett, Pedro M. Esperan¸ca and Chris C. Holmes, "***A review of homomorphic encryption tools for encrypted statistical machine learning***", Cornell University, **arXiv:1508.06574, ,** Aug 2015.

[15] Nigel Smart and Fre Vercauteren, "***Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes***", Springer Berlin Heidelberg, Vol. 60, PP: 420-443, 2010.

[16] N. Vamshinath, K. Ruth Ramya, Sai Krishna, P. Gopi Bhaskar, Geofrey L. Mwaseba, and Tai-hoon Kim, "***Homomorphic Encryption for Cluster in Cloud",*** International Journal of Security and Its Applications, Vol. 9, No. 5 (2015), pp. 319-324.

[17] Craig Gentry, ***"A Fully Homomorphic Encryption Scheme***", Ph. D. theses, Stanford University, USA, 2009.

[18] Raghul.H, Ramagopal.R.N, Saravanan.B, Guhapriya.T and , Anitha.R, "***Data Security in Federated Cloud Environment using Homomorphic Encryption Technique***", International Journal of Emerging Technology and Advanced Engineering, Volume 5, Issue 4, April 2015).

[19] Shu-Ching Chen, et al., "***A Three-Tier System Architecture Design and Development for Hurricane Occurrence Simulation***", Proc. of the IEEE Int. Conference on Information Technology Research and Education (ITRE), USA, 2003.

[20] G.O. Ofori-Dwumfuo and E. Paatey, **"*The Design of an Electronic Voting System***", International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume-2, Issue-1, March 2013.