EDU-VOTING: An Educational Homomorphic e-Voting System

L. TEKIN[1], H. G. OZGUR[1], B. SAYIN[1], A. KARATAS[1], P. SENKULA[1], E. IRTEM[1], and S. SAHIN[1]

[1]Izmir Institute of Technology, Izmir/Turkey, leylatekin@iyte.edu.tr
[1]Izmir Institute of Technology, Izmir/Turkey, huseyinozgur@iyte.edu.tr
[1]Izmir Institute of Technology, Izmir/Turkey, burcusayin@iyte.edu.tr
[1]Izmir Institute of Technology, Izmir/Turkey, arzumkaratas@iyte.edu.tr
[1]Izmir Institute of Technology, Izmir/Turkey, pelinsenkula@iyte.edu.tr
[1]Izmir Institute of Technology, Izmir/Turkey, emreirtem@iyte.edu.tr
[1]Izmir Institute of Technology, Izmir/Turkey, serapsahin@iyte.edu.tr

Abstract - As an instrument of democracy, voting is a critical issue. Although paper-based voting systems are still used commonly, e-voting systems have started to substitute under favor of improvements in the technology. This situation gives rise to need for secure, reliable, and transparent e-voting systems to make people trust. To do this, there are some security requirements that should be concerned and satisfied such as privacy, fairness, verifiability etc.

This study has an educational intuition that analyzes those requirements, theoretical background information related to cryptographic schemes behind them and creates a place-based e-voting design which was implemented for kiosk voting. As a contribution, Paillier homomorphic cryptosystem is used in our system. Moreover, our study includes a detailed criticism for the implemented system in terms of chosen cryptosystems and design modules with security and e-voting requirements.

Keywords: e-voting, Paillier homomorphic cryptosystem, privacy, secrecy, multi-party computation, elliptic curve cryptography, PKI, digital signature, open source software.

I. INTRODUCTION

Because of the advanced technology, e-voting becomes a hot topic that many countries and organizations try to adapt their voting systems to electronic environment. Especially, small organizations prefer to use e-voting systems because of its practicability. The most critical issue here is that how well the system is designed. Naturally, the need for a reliable, secure, and adaptable system remains.

Key requirements for an e-voting system implemented on any scale includes (i) *inalterability:* once a vote is casted, it cannot be modified. (ii) *non-reusability:* a voter must have only one valid vote. (iii) *eligibility:* only eligible voters should cast a vote. (iv) *fairness:* unless the voting process ends, counting process cannot be started. (v) *individual verifiability:* every voter should keep track of whether her own casted vote still in the system or not. (vi) *universal verifiability:* everyone can verify the correctness of the whole voting process and results according to announced system keys and data. (vii) *privacy:* votes cannot be correlated with voters with the help of vote anonymization. (viii) *authentication:* checking the credentials of anyone to see whether the proffered identity consistent or not. (ix) *integrity:* while applying some operations on data, protecting the accuracy and consistency of it. (x) *coercion-resistance:* providing a fake credential for every voter to use in any possible coercion case. (xi) *receipt-freeness:* attackers cannot find any receipt of a voter's casted vote. (xii) *secrecy*: ensuring that no one can read the message except the intended receiver.

In order to build a well-designed e-voting system, key requirements mentioned above should be satisfied. A variety of cryptographic algorithms propose solutions to solve security problems and meet requirements. According to the necessities of the domain, in which an e-voting system will be adapted, the most suitable algorithms should be chosen and applied.

The aim of the following study is to analyze the requirements for a local e-voting system (e.g. Student Association Elections for a university) using homomorphic encryption scheme, cryptographic background and possible vulnerabilities which may cause system to fail. Furthermore, this study was performed with an educational approach and has a modular architecture. Related open source code and completed documentation placed on GitHub[10]. In accordance with mentioned purposes, Helios system [1] was examined and used as a related study.

The remainder of the paper is organized as follows. Section 2 presents related e-voting studies. In Section 3, we give an enhanced explanation of our system design. A comprehensive summary of our work which also includes the selected cryptosystems and satisfied requirements for each system module is given in Section 4.

II. RELATED WORK

Helios [1] is the first web-based and open-audit e-voting system. It is wholly transparent that anyone can create an election on the system and get a result from there. Additionally, it is completely trackable by anyone (open-audit) so the voting process can be maintained securely and fairly. Due to these reasons, we selected this system as a related work for our study. We examined the whole system with its cryptographic background and tried to create a novel design according to our selected security requirements. Implementation steps of the Helios system is as follows:

- Admin creates an election and determines the election dates and name.
- ElGamal cryptographic scheme is used to construct the system private and public key.
- Admin arranges the ballot.
- Admin adds voters to the system together with their names

and email addresses.

- System creates a password for each voter. This password is sent to voters via e-mail.
- Admin suspends the system and takes the election details as a JSON format.
- When the voting process starts in the specified time, a notification together with the election link is sent to voters.
- When voters click to the link, a java applet is run.
- The casted votes are encrypted, and hash value of the encrypted vote is send to voters and kept in a bulletin board.
- When the voting process ends, casted votes are anonymized by shuffling with a proof.
- Admin decrypts the shuffled ciphertext by verifying with a proof.
- After counting process, auditors can reach to all election data and verify the whole process with the generated proofs.

Zeus [8] is another verifiable e-voting system, and it is based on Helios. Neither a new workflow, nor any new e-voting protocols are described in the scope of this project. Proven systems and ideas are used with some contributions as follows; (i) an extension is added to handle all kind of voting system, (ii) all cryptographic content is wholly documented, and (iii) audit system is improved to check both integrity of the vote, and authenticated communication channel.

Sailau [9] is another e-voting system. It is like a paper-based voting. Its difference from paper-based voting is that the voter does not mark the ballot on paper, instead she takes a smart card from the registration table and casts her vote on the voting terminal which is in a different place. Selection of the voter is saved into the smart card. Afterward, the voter inserts the smart card to the electronic box in order to make her vote recorded to the system. A four-digit number is returned to the voter, this number is used for verification after election is over. Then she signs the electronic poll book with her id card.

Unlike the mentioned related works, our proposed e-voting system is based on a Kiosk voting. The aim of it is to create a research environment for developing a viable e-voting system. It differs from the referenced Helios system by means of both selected cryptographic scheme i.e. Paillier Homomorphic cryptosystems and application domain as kiosks. Further information related to our system is explained in Section 3.

III.SYSTEM DESIGN

A. System Overview

The proposed e-voting system consists of seven essential modules which are public key infrastructure (PKI) implementation, secret sharing, registration, election preparation, vote casting, vote management and tallying.

Flowchart of our proposed system is shown in *Figure 1*. Functionality of system components are explained in detail below.

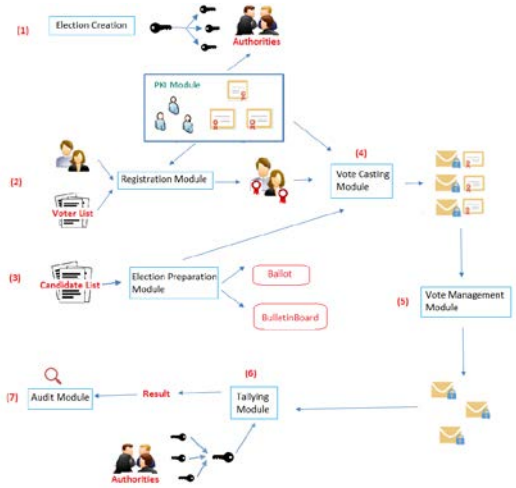

Figure 1. System Overview

Stage 1 - Election Creation: Admin determines election dates and name, then creates the election.
Eligible voters, candidates and authorities are predefined in the system with their names and email addresses and certificates with the help of PKI module. Paillier homomorphic scheme is used to construct the system private and public key. Then, private key is partitioned to shares and distributed to authorities.

Stage 2 – Registration module: This process starts at predefined registration time. Eligible voters who want to cast a vote must register to system. System creates and gives a registration code for eligible vote and a fake registration code to detect coercion case for each registered voter.

Stage 3 – Election Preparation module: After voter registration process ends, election preparation process starts. Predefined candidates are obtained from database as candidate list and an identification number is given for each. Next, ballot that contains the list of candidates is created. Later, the number of eligible voters is counted. Following this, a bulletin board is created.

Stage 4 – Vote Casting module: Voting process starts at defined voting time. When a voter casts a vote, it is encrypted by system public key of Paillier Homomorphic Cryptosystem. Then, login panel is opened, and username and registration code are asked. If the login is successful, voter signs the encrypted vote by her private key (Elliptic Curve Digital Signature – ECDSA) which is related her PKI certificate. Hash of the registration code, encrypted vote, and vote signature is concatenated. Then hash value of this concatenation is created and given to voter for tracking her vote. Voters can query their casted votes from the bulletin board via using given hash value.

Stage 5 – Vote Management module: When the voting process ends, casted votes are verified by using registration code entered at casting time. If real registration code is used, the vote is verified as valid vote, else if coercion (fake) code is used, then the vote is verified as invalid vote. Thereafter, valid votes are anonymized by shuffling.

Stage 6 – Tallying module: Anonymized votes are counted without decryption by the facility of homomorphic Paillier cryptosystem. After counting process of the encrypted votes, each

authority sends its private key share to others by Shamir's secret sharing scheme [7]. Authorities who build the system private key starts to decrypt the votes (by multiparty computation).

Stage 7 – Audit module: Universal verifiability is satisfied by bulletin board. Bulletin board shows election steps and related counts and percentages for registered voters, before and after shuffling number of valid and invalid votes, and the result of election. For the auditing purposes; after decryption process ends, auditors can reach to all election data with last results from database, hence verify the whole process step by step with the system details and keys.

Each module as mentioned here above will be explained detailed in following subsection 3.4.

B. System Requirements

In our system, we addressed all fundamental requirements described in introduction part to satisfy the needs of an e-voting system: (i) inalterability, (ii) non-reusability, (iii) eligibility, (iv) fairness, (v) individual verifiability, (vi) universal verifiability, (vii) privacy, (viii) authentication, (ix) integrity, (x) coercion-resistance, (xi) receipt-freeness, (xii) secrecy. *Table 1* in conclusion part shows an overview to specify which requirements were satisfied and which cryptosystems were used in which module. Furthermore, we have an additional constraint for this design that the one election can be carried out at same time interval.

C. Assumptions

There are some assumptions to make our system viable, and simulate the core functionalities more smoothly:

1. Admin is a trusted actor.
2. Authorities, candidates and voters are known by system election poll before starting time of specific election.
3. Each voter and authority have a public-private key pair and certificate in system creation time. Those certificates are verified with using public key of Certification Authority (CA) for each interaction with voters, which is default module in our system to satisfy authentication requirement.
4. Authorities are honest in the whole election process. Authorities use principles of Shamir's scheme [7] and they are responsible for saving the decryption key shares as secret. However, in this simulation the shares are stored in text files.
5. Whole system is embedded in only one Kiosk machine. However, modules can be distributed by different architectures in future. In that case there would be a secure communication problem. For this reason, encrypted communication is used as a default to create secure communication channel between modules in this design and implementation.
6. Security requirements and risks of network and communication protocols are out of the scope.

D. System Modules

i. Stage 1 - PKI Module

PKI [11] is essential for an e-voting system because it ensures authentication, non-repudiation and integrity with Elliptic Curve Digital Signature Algorithm (ECDSA). Each certificate is signed by CA and includes related authority's, or related voter's public key. The PKI module includes two methods as generation and verification of signature. A key pair for CA and a signed certificate for each user are created in the system before election time. In the registration process, signed certificate of related user is verified by CA verification. There is no registration process for authorities, they are already defined in the system as specified in assumptions (section 3.3).

According to implementation of PKI scenario, voter should have signed certificate from CA. As a novelty, here we used ECDSA to generate signature with using CA's private key in the signing process. It is used for verification of generated signature with public key of CA by voter registration module.

Elliptic curve cryptography uses smaller key size for the same level of security when compared to other cryptosystems. Therefore, it is implemented in the project. One of the NIST recommended elliptic curves can be chosen for ECDSA in this implementation. Koblitz [2] shows detailed structure of elliptic curves, and FIPS PUB 186-4 [3] shows both detailed structure of elliptic curve and recommended elliptic curves that contain choices for private key length and underlying fields.

ii. Stage 2 - Secret Sharing

In this study, Shamir's (k, n)-threshold secret sharing scheme [7] has been applied to implement sharing and reconstructing the private key of the system. If there is one authority in the system to keep the private key, this can lead to key escrow problem since the result of all counting operation will depend on only this authority. To handle key escrow problem, the key should be distributed to multiple authorities. Thus, the private key of the system is split into n shares and distributed to n authorities, and any threshold k or more authorities together can recover the private key. To simulate the scheme, each authority has a public-private key pair and certificate for authentication and perform the following steps: Firstly, each authority saves its share given by a dealer to a text file in its local device in sharing step. Then, in reconstruction step, it sends its own share to other authorities by encrypting with their public keys via Elliptic Curve Diffie-Hellman message transfer protocol [4] to satisfy the secrecy requirement. So, authorities check certificates of other authorities before sending the shares for authentication. The reconstruction step is used by Tallying module at stage 6 (section 3.4.7 includes details).

The system security and reliability are increased by using an efficient threshold cryptosystem because at least k authorities are required to reconstruct the secret which means that an individual authority cannot count the votes with only its share and cannot obtain any early results which could affect the remaining voters. So, fairness increases. As our future work, digital signature scheme can be added to ensure the integrity and authenticity of the shares, and verifiable secret sharing techniques can also be incorporated to provide the consistency of the shares.

iii. Stage 3 - Registration Module

Registration is the first step for eligible voters. In real system, signed certificate of voter is brought by the voters via portable

devices such as smart cards or flash memories. However, voter information and signed certificates are stored in our system for simulation purpose. Eligible voters have to register by filling registration request form with their genuine information and certificate name. Then they have to sign the form to be a registered voter. This module performs the following tasks in registration time specified at the election creation (stage 2 as described in Figure 1):

- checks certificate validity by using PKI module
- checks whether the information sent via form compromises the certificate information and stored voter information.
- generates two unique 6-digit alphanumeric codes for every registered voter: (i) registration code (satisfies *authentication* and *eligibility* requirements) and (ii) coercion code (ensures *coercion-resistance requirement*). It is explained in vote casting module at section 3.4.5 that how to satisfy those requirements. The uniqueness of both registration and coercion codes helps to increase voter *privacy* by adding randomness factor for a casted vote.

iv.Stage 4 - Election Preparation Module

After registration stage, a preparation should be done for voting process. In election preparation module at stage 3 as shown in Figure 1; six fundamental tasks must be performed, which are (i) collecting information of predefined candidates (ii) creating identification numbers to represent the candidates in database, (iii) generating ballot which contains candidate list, (iv) creating bulletin board, (v) counting number of eligible voters and (vi) showing the number of eligible voters on the board.

v.Stage 5 - Vote Casting Module

The steps of a vote-casting can be explained in six steps as below:

Step 1: During voting process, a voter first selects a candidate on ballot without logging in to system. This is important to protect voter privacy.
.

Step 2: Then identifier number of the selected candidate is encrypted with system public key. Paillier homomorphic cryptosystem [5] is used for encryption of the casted vote.

Step 3: After encrypting the vote, the voter should login to system. Username and registration codes are needed for login process. If the voter is under coercion, she should use the fake registration code (coercion-resistance requirement). If the voter first casts the vote with fake registration code, system accepts it and does not allow the voter to cast a fake vote again. If there is no coercion case, the voter should enter the actual registration code.

Step 4: After specifying the login credentials, system checks whether login is successful or not. This check operation provides authentication and eligibility requirements. If it is successful, then the validity of the casted vote is checked. System controls if there is already a valid casted vote by this voter. If so, system does not accept the casted vote and shows a message that says, "you have already casted a vote". If not, the casted vote should be validated before saving to system. So, non-reusability requirement is satisfied. Also, system does not give any receipt to the voter to prove how she voted to meet the receipt-freeness property.

Step 5: Signing is needed for validating casted vote. The voter must sign encrypted vote to make it valid. Signature generation steps here uses ECDSA [3] to assure secrecy and integrity requirements. The most important point for signing a vote is that each voter has her own private, and public key pair as explained in PKI module. Voter's private keys are long enough, so they cannot memorize. There is some hardware such as smart cards to store private keys. But, for simulation, we did not use any hardware. To simulate signing of the encrypted vote, voters' private keys are stored in the database. In the real system, this module can be exchanged with a storage material to keep private keys.

Step 6: After signing operation, system concatenates the hash of the entered registration code, encrypted vote, and signed encrypted vote. Then, system takes the hash value of this triple concatenation. The resulting hash value is saved to the database and returned to the voter.

These six steps are necessary to cast and save a vote. Additional functionalities are included in this module according to requirements. For example, we added a "Query My Vote" panel to our system. By using this module, voters can query their casted vote to see whether it is saved to system or not by checking the hash value given to them at Step 6. This option satisfies individual verifiability requirement. If the given and queried hash values are equal, then it is proved that no one changes her vote which means that inalterability requirement is assured.

vi.Stage 6 - Vote Management

After vote casting process ends, casted votes should be managed with satisfying privacy, secrecy, and integrity requirements, and using coercion-resistance requirement. Vote management is done in two steps: vote verification and shuffling.

Vote verification process has two objectives which are (i) check whether the vote was casted by a registered voter and (ii) check whether the vote was casted under coercion. These objectives are checked by the following steps. First, it collects the casted votes. Then it determines the validity of the votes by checking the signatures on each vote through certificate (authentication requirement) and the related registration codes entered to system at casting time of that vote (eligibility requirement). If the authentication is done via coercion code, it is inferred that the vote is casted under coercion (coercion-resistance requirement). Votes have to be anonymized to satisfy privacy requirement. Therefore, each vote is made disjoint from its signature as in seen Figure 1. Then, it is put into invalid-vote pool if there is coercion. Otherwise, it is put into valid-pool, so real registration code was used at casting time. At the end, counts of valid and invalid votes are shown on the bulletin board for universal verifiability.

Anonymized valid votes are shuffled before tallying process to destroy the order of casted votes. This operation makes sure that there is no link between anonymized votes and the corresponding voters anymore. Shuffling is the last step to ensure voter privacy requirement.

In shuffling process, all anonymized votes are given to a shuffler. The shuffler uses a blind function to perform shuffling. It

is performed in four steps [6]: (i) keep the given votes as ciphertext, (ii) keep the permuted order of them as a vector, (iii) re-encrypt the ciphertext and the permutation vector with Paillier encryption scheme and (iv) decrypt it to get shuffled votes. We performed only one level shuffling, but more than one level can be used for extra security. Naturally, shuffling process is used with a commitment scheme to add a verification step such as Groth's protocol [6]. However, we did not apply any commitment scheme so there is no prover in our shuffling process, yet. Shuffling process in our system assures that secrecy and integrity of the votes are preserved without revealing permutation order.

vii.Stage 7 - Tallying Module

All the computation on computer-based systems is related with data. Data processing can produce wide range of results. Not only efficiency of processing, but also secrecy of the data is essential. Also, personal information of users must be used just for intended computation for privacy. Therefore, confidentiality of data should be regarded at the environment in which data is processed. In e-voting systems, the vote confidentiality can be served via processing encrypted form of votes without decryption. In this context, vote counting refers calculating sum of all encrypted votes, so encrypted votes must be processed. Homomorphic cryptosystems allow to handle encrypted data.

Both Paillier, and ElGamal are considered partially homomorphic cryptosystem. Because, for given any two ciphertexts, they support the generation of a ciphertext associated with either the addition or the multiplication of the two underlying plaintexts, but not both. Paillier supports addition of the plaintexts, ElGamal supports multiplication of the plaintexts [12].

According to Paillier additive property, decryption of the products of two or more ciphertexts result in the sum of their corresponding plaintexts. Also, votes are encrypted using system public key, and encrypted votes are multiplied in our system by homomorphism facility. It means that sum of encrypted votes is calculated by this way. Before decryption of the result, private key must be reconstructed among authorities. Reconstruction procedure is explained in secret sharing part (3.4.2). Decryption of this sum is only done by one who builds the system private key.

As a novelty our contribution is using of Paillier Homomorphic Cryptosystem to create an e-voting solution. Our system design is built according to the facilities of Paillier. But, we have to know its efficiency from this implementation. As presents in Figure 2, more number of voters causes increasingly growing elapsed time. These results are computed on a computer with 2.4 GHZ CPU speed, 16GB memory capacity, Arch Linux operating system.

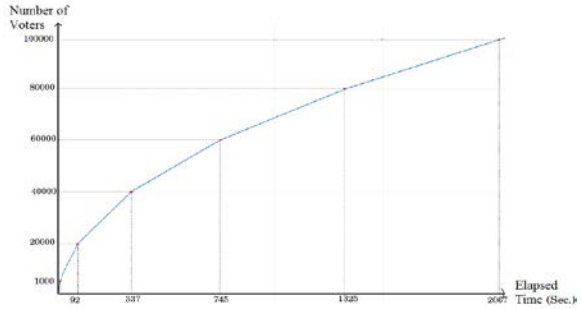

Figure 2: Relation between Elapsed Time and Number of Voters in Paillier Counting Process

The reason of decreasing efficiency is that Paillier counting method multiplies each vote to calculate encrypted result, it means that more number of vote is more elapsed time.

viii. Stage 7 - Audit Module

Anyone can monitor each step of election from Bulletin Board as universal verifiability requirement. Bulletin Board shows "how many registered voters / how many valid and invalid votes exist and how many votes are counted and results" etc. Additionally, for auditing purpose; after election result is announced, auditors can reach to all up-to-date election data from database, therefore verify the whole process step-by-step with system details and keys.

E. Construction of Design

We would like to give very brief information about our construction. Because all detailed documentation is accessible from GitHub [10] link.

We implemented software engineering principles to have modular design. System has three layered structure as database, object and user interface layers. Java programming language was used as implementation environment with oracle database connection.

IV. CONCLUSION AND FUTURE WORK

In this paper, we have provided the design and implementation of a secure and efficient local e-voting system based on the homomorphic encryption which guarantees the desirable security requirements like privacy, eligibility, verifiability, receipt-freeness etc. *Table 1* shows the detailed list of satisfied requirements and used cryptosystems for each module of our system design.

Table 1: An overview of modules in terms of requirement satisfaction and cryptosystems

	System Requirements												Cryptosystems						Stage
Modules	Inalterability	Non-reusability	Eligibility	Fairness	Individual verifiability	Universal verifiability	Privacy	Authentication	Integrity	Coercion-resistance	Receipt-freeness	Secrecy	Shamir' s secret sharing scheme	SHA3	Paillier homomorphic cryptosystem	ECDSA	ECDH message transfer protocol	Groth' s Shuffling protocol without commitment	
PKI							✓	✓						✓		✓			
Secret Sharing				✓								✓	✓			✓	✓		(1)
Registration			✓				✓	✓		✓				✓		✓			(2)
Election Preparation																			(3)
Vote Casting	✓	✓	✓		✓		✓	✓	✓	✓	✓	✓		✓	✓	✓			(4)
Vote Management							✓		✓	✓		✓		✓	✓			✓	(5)
Tallying				✓		✓						✓			✓				(6)
Audit						✓													(7)

Our system has been built with educational purposes by implementing cryptographic algorithms, taking into consideration requirements of e-voting system and basic security functions.

First as presents in Figure 2, e-voting implementation and design should be evaluated according to cost of Paillier. Different homomorphic solutions can be also tested with this modular design and measure their efficiencies.

The Kiosk that voters use to vote is not connected to a network. That's why, it is protected against a vast number of cyberattacks.

For the future scenarios, system modules can be spread to different locations or untrusted servers. System have to provide solutions to prevent network attacks. The secure channel has to be guaranteed among the communication of modules. Naturally the cost of this additional cryptographic implementations will be analyzed, and whole protocol verification and validation will be realized.

However, the system is already subjected to social engineering attacks to steal the registration code. For the future work, codes can be sent via e-mail, or SMS against codes the written down by the registered voter.

REFERENCES

[1]. Ben Adida. 2008. Helios: web-based open-audit voting. In Proceedings of the 17th conference on Security symposium (SS'08). USENIX Association, Berkeley, CA, USA, 335-348.

[2]. Neal Koblitz, *Elliptic Curve Cryptosystems*, Mathematics of Computation Volume 48, Number 177, January 1987, Pages 203-209

[3]. Federal Information Processing Standards Publication 186-4

[4]. Alfred Menezes, Elliptic Curve Public Key Cryptosystems, Kluwer Academic Publishers, 1993.

[5]. Craig Gentry. 2010. Computing arbitrary functions of encrypted data. Commun. ACM 53, 3 (March 2010), 97-105. DOI=http://dx.doi.org/10.1145/1666420.1666444

[6]. Stephanie Bayer and Jens Groth. 2012. Efficient zero-knowledge argument for correctness of a shuffle. In *Proceedings of the 31st Annual international conference on Theory and Applications of Cryptographic Techniques* (EUROCRYPT'12), David Pointcheval and Thomas Johansson (Eds.). Springer-Verlag, Berlin, Heidelberg, 263-280. DOI=http://dx.doi.org/10.1007/978-3-642-29011-4_17

[7] Adi Shamir. How to Share a Secret. Communications of the ACM, 22(11):612–613, 1979.

[8] Georgios Tsoukalas, Kostas Papadimitriou, Panos Louridas and Panayiotis Tsanakas. From Helios to Zeus. USENIX Journal of Election Technology and Systems(JETS), Volume 1, Number 1, 2013.

[9] Douglas W. Jones. Kazakhstan: The Sailau E-Voting System

[10] https://github.com/Edu-Voting

[11] Niels Ferguson, Bruce Schneier, Tadayoshi Kohno, Cryptography Engineering: Design Principles and Practical Applications, Wiley, 2010.

[12] Fang-Yu Ra, On the Security of a Variant of ElGamal Encryption Scheme