End-2-End Verifiable Internet Voting Protocol Based on Homomorphic Encryption

Ahmet Sınak[1,3], Seçil Özkan[1], Hakan Yıldırım[1] and Mehmet Sabır Kiraz[1,2]

[1]Institute of Applied Mathematics, METU, Ankara, Turkey
[2]TUBITAK BILGEM UEKAE, Kocaeli, Turkey
[3]Department of Mathematics and Computer Sciences, Necmettin Erbakan University, Konya, Turkey
E-mails: ahmet.sinak@metu.edu.tr, {seilzkan, hakyildirim}@gmail.com, mehmet.kiraz@tubitak.gov.tr

Abstract—We have constructed a new End-2-End verifiable Internet voting protocol. Our motivation comes from the real experiments the Estonian and Norwegian schemes which are used both in local and general elections. Specifically, we are interested to avoid possibility of a malicious cooperation between Ballot Box and Receipt Generator in the Norwegian scheme. We also propose an alternative solution for coercion which is one of the most important issues for Internet voting. In our protocol, a voter is able to verify whether her vote reaches in the counting process.

Keywords—Electronic Voting, Internet Voting, Homomorphic Encryption

1. Introduction

In recent years, electronic voting solutions are used to elect political representatives by most of the countries besides the classical solution. Electronic voting (e-voting) can be mainly classified into two different systems: machine based systems and Internet voting (i-voting) systems. Machine based e-voting means that both casting a vote and tallying the votes are performed using dedicated electronic devices. The first e-voting machine, also the first e-voting system was proposed in 1800's in the United Kingdom [4]. After 1970's, the idea of e-voting have become increasingly common. Some significant inventors over the world have developed different electronic devices including Direct Recording Electronic (DRE) system, optical scan voting system, Voting By Mail (VBM) system and the others [13]. Some e-voting protocols based on these systems have been used by many countries around the world. On the other hand, i-voting system is the other type of system which is interesting to use for the countries where the voting participation is low and it is also useful for expatriates. I-voting is a voting method that transmits casted-votes via public Internet. Because of the convenience attainability, i-voting has been particularly attractive. The developed i-voting technology helps the providers to construct secure systems; indeed, with new technological developments, casting votes are more convenient by using personal computers, smart phones via Internet. Counting process in the i-voting system can also be classified into two different methods. One of them is based on mix-

nets [3] which is efficient for large-scale elections. The other one is homomorphic tallying which can be more suitable for small-scale elections. Our i-voting protocol is based on homomorphic tallying.

I-voting systems have been used since twentieth century. Although i-voting is quite useful, its usage is stopped in some countries due to security and the transparency concerns. Till 2012, eleven countries have used i-voting solutions in their elections [10]. Estonia, Norway, Switzerland, France, Germany, Spain, Paraguay, the Netherlands and the United Kingdom tried to apply i-voting systems in either local or general elections. Some of them are ongoing pilot voting, while the others have discontinued. Germany, Netherlands and Paraguay have decided to discontinue or restrict their use in the future. On the other hand, the others including Estonia, Norway and France have trusted to their systems and continued to the usage of i-voting. Estonia is the only country that uses i-voting systems on a nationwide level. Moreover, Estonia is the first country that has used i-voting for general elections since 2005. Estonian i-voting system and its analysis are presented in [19]. Norway is another country who used i-voting for both local and governmental elections in September 2011 and 2013, respectively. The original cryptographic protocol used in Norwegian election has been prepared by a Spanish company (Scytl) and later has been modified to suffice certain requirements. To create society's trust to new system, Norwegian government made all parts of the technical details of the solution public [12]. Moreover, Norwegian i-voting system has been analysed and some improvements has been accomplished in [16], [18].

In classical election method, the voters have to cast their vote in a specific place during the election. Thus, there can be rush in airports, terminals in which people are trying to go to specific places to cast a vote. Some people have to stop their holidays or change their places for a specific time interval (during the election) which lead to a loss of time and money. Moreover, the obligation to cast a vote in a certain place decreases the percentage of the voters. Therefore, i-voting is believed to be useful for the voters in the country and for the expatriates. I-voting scheme not only makes people's lives easier, but also increases the voter participation. On the other hand, the idea of i-voting is also equivalent to create new security concerns. The most important concerns in the i-voting protocols are security, privacy and coercion. There are many other issues to be solved, e.g., authentication, vote buying/selling, integrity, verifiability, vote privacy, tallying, receipt-freeness and usability. In our proposed i-voting protocol, we tried to take some precautions against those issues.

Contribution: Our aim is to present a secure, E2E i-voting system which slightly improves the Norwegian protocol. We try to highlight the weaknesses of Norwegian protocol and fix them in our protocol. There are four motivations of this paper. The first one is to simplify the *receipt code* which is more secure than the Norwegian's receipt code. Their receipt codes are constructed by two online players. If these players compromise, they can obtain the private key of the election. In this case, the privacy of the voter have been violated and also confidentiality of the system has been compromised. Our i-voting protocol is more secure than Norwegian protocol since these two players or the others cannot obtain the private key of the election. The second motivation is to suggest a new scheme to decrease the possibility of the coercion. Namely, we propose to use a *voter secret number* denoted by sn for each voter where the voter uses in case of coercion while casting a vote. When the entered secret number is right, the vote is tallied; otherwise, it is not counted. Therefore, the possibility of coercion is diminished.

The third motivation is to use a *Bulletin Board* BB in our protocol. The voter can verify from BB whether her vote reaches to the counting process, which increases the credibility of the system. Finally, our i-voting solution is *almost End-2-End Secure*, in which the system's security is considered from the initial point to the final point of the system. E2E secure system guarantees that casted votes are recorded, transferred and tallied correctly without the connection of the voter-vote.

Organization: The rest of this paper is organized as follows. Section II gives the previous i-voting protocols which are Estonian and Norwegian protocols. Section III describes the basic algorithms and definitions which are used in our protocol. In Section IV, we explain the main infrastructures and key generation process of our protocol. Moreover, this section presents the main i-voting protocol. Section V, VI evaluate security analysis and complexity analysis of this protocol, respectively. Finally, we present the main diagram of i-voting system in Appendix and conclude this paper.

2. Previous I-voting Protocols

The pioneer country of the nationwide i-voting usage in the world is Estonia. In 2005, Estonian government firstly used i-voting system in local election. In 2007 parliamentary election, Estonian government used i-voting that was the first usage in parliamentary elections in the world. From 2005 to the present, Estonia has used i-voting and each time the rate of the participant has been increased because of the system's reliability. Also on 11th of July 2013, Estonian Electronic Voting Committee revealed the source code of voting system, in order to open up technical analysis of the voting system to the public (https://github.com/vvk-ehk/evalimine). Technically, Estonian i-voting system consists of i-voting protocol, i-voting system and i-voting client application (IVCA). In the I-Voting System there are three servers; the Vote Forwarding Server (VFS) corresponds to the authentication, candidate lists distribution and i-vote collection. The Vote Storing Server (VSS) corresponds to the storing the i-votes and making them anonymous. The Vote Counting Server (VCS) corresponds to the tallying the votes. In this system, voting process is as follows. The voters use their identity cards to authenticate the system and also for digital signature. The voter is authenticated by the VFS. Then the chosen vote is encrypted with RSA encryption algorithm. The encrypted vote is produced as an anonymous ballot, which is later digitally signed with the signature algorithm on ID-card. All the encrypted and signed votes are forwarded to the VFS where the signature is verified and valid votes are transferred to the VSS. Until the end of the election, the votes are stored in VSS and then the encrypted votes are tallied in VCS and finally the result is announced [25]. We note that this system enables the voters to re-vote as a security precaution; however, the last one is valid [5].

Norway is the second country that used i-voting systems for local government election. The Norwegian i-voting system is an improvement of the Estonian i-voting system. The technical information about the system has been released by the government to provide the public trust [11]. Norwegian system is more complicated than the Estonian i-voting system. In this system, there are two improvements as pre- and post- channel information. Pre-channel information thought as receipt code paper which is prepared before the election and there are specific verification numbers on it for each voter. In the election time, the voter can control her vote by using this receipt code paper. Also, after the vote submission process, each voter takes a verification SMS. This SMS is used for the post-

channel information of the system. As Estonian system, the voter can submit her vote multiple times and the last one is valid. [14].

The protocol consists of mainly six components; the voter V, the voter's computer P, the ballot box B, the receipt code generator R, the decryption service D and the auditor A. Voting process is as follows. A vote is selected by the voter then it is encrypted in the system and digitally signed with the signature in the voter's ID-card by P. B and then R receive the encrypted vote, respectively. R sends to the voter the control SMS, and the voter can control her vote with the receipt code sent before by mail service. At the end of the election, the encrypted ballots are submitted to D, and then it decrypts the ballots and publishes the result [12].

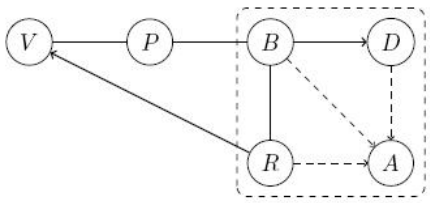


Fig. 1. Norwegian I-voting Protocol [12]

Norwegian system is more secure except for coercion and compromised computers. Also the more pronounced deficient is in the receipt code generator. This deficiency is as follows. In this system there exist three mathematically dependent secret parameters (a_1, a_2, a_3) shared among D, B, R, respectively, which are

$$a_1 + a_2 \equiv a_3 \mod q$$

where G is a group of order q and with generator g. System uses the public parameters, y_1, y_2 and y_3 generated by the secret parameters as;

$$y_1 = g^{a_1}, y_2 = g^{a_2} \; and \; y_3 = g^{a_3}.$$

R generates the receipt codes for each voter as a set $\{(v; r(v))\}$. Receipt codes are generated as the form of $r(v) = d((f(v)^s)$ where v is the vote, f is the encryption function of the voting system, s is selected for each voter $(s \in \{0, \cdots, q-1\})$, d is a pseudo-random function and r is the receipt generator function [12]. Any collusion of B and R gives them the decryption key $a_1 \equiv a_3 - a_2 \mod q$. Thus, the vote and voter's privacy are vanished.

This problem violates the security and privacy of the system. To eliminate this breach of confidentiality, we have tried to construct E2E secure i-voting protocol. Our protocol will be described in Section 4.

3. Cryptographic Background

In this section, we present fundamental definitions and cryptographic algorithms required for our i-voting protocol.

3.1. Discrete Log Assumption

Let $G =< g >$ be a multiplicative cyclic group of prime order p (p is a very large prime).

$$G = \langle g \rangle = \left\{ g^0, g^1, \ldots, g^{p-1} \right\}, \; \#G = p$$

Discrete Log Assumption states that it is hard to calculate x from the random group element $y = g^x$ by using the generator g.

3.2. ElGamal Encryption Scheme

The ElGamal scheme [24] is a public-key cryptographic system based on *the discrete logarithm problem*. It consists of both encryption and signature algorithms. Security of these algorithms depends on the difficulty of computing discrete logs in a large prime number. The encryption algorithm is similar in nature to the Diffie-Hellman key agreement protocol. The signature algorithm is used to sign the encrypted votes in our protocol. The encryption

algorithm is also chosen in our protocol to encrypt the votes because of four advantages. These are homomorphic properties, efficiency of the algorithm and it is sufficiently standard. The last but not least advantage is that the same plaintext gives a different ciphertext each time it is encrypted (because of the randomization). On the other hand, ElGamal encryption algorithm has a disadvantage that the ciphertext is twice as long as the plaintext.
The encryption scheme consists of three phases [6];

- Key generation:
  G is a multiplicative cyclic group of order p. For the private key $x \in \{1, \ldots, p-1\}$, $y = g^x$ is the public key.
- Encryption:
  The encryption of a message m is

$$E : M \times R \to G \times G$$
$$(m, r) \longmapsto E(m, r) =$$
$$(g^r \mod p, my^r \mod p) = (A, B)$$

  where $r \in_r R$.
- Decryption;
  $\frac{B}{A^x}$ simply output message $\frac{my^r}{(g^r)^x} = \frac{m(g^x)^r}{(g^r)^x} = m$.

3.3. Homomorphic Encryption

Homomorphic encryption is an encryption scheme with a special property that allows operations applied to ciphertext be preserved and carried to the plaintext. Let M be the set of plaintexts, C be the set of ciphertexts, and K be the set of keys. An encryption scheme is said to be homomorphic if for any given encryption key $pk \in K$, the encryption function E satisfies; $E_{pk}(m_1 \odot_M m_2) = E_{pk}(m_1) \odot_C E_{pk}(m_2), \forall m_1, m_2 \in M$.
There exist two types of homomorphic scheme:

- Multiplicative Homomorphic Scheme:

$$E(m_1) \odot_C E(m_2) = E(m_1 \cdot_M m_2).$$

- Additive Homomorphic Scheme:

$$E(m_1) \odot_C E(m_2) = E(m_1 +_M m_2).$$

Multiplicative Homomorphic Property of ElGamal Encryption: The encryption of a message m is

$$E(m; r) = (g^r \mod p, my^r \mod p)$$

where $r \in_r R$. For $m_1, m_2 \in M$ and $r_1, r_2 \in_r R$,

$$E(m_1, r_1) \odot_C E(m_2, r_2) = (g^{r_1}, m_1 y^{r_1})(g^{r_2}, m_2 y^{r_2})$$
$$= (g^{r_1+r_2}, m_1 m_2 y^{r_1+r_2}) = E(m_1 m_2, r_1 + r_2)$$

Additive Homomorphic Property of ElGamal Encryption: In our protocol, we use additive homomorphic ElGamal cryptosystem. If we modify ElGamal encryption algorithm then the encryption of m is

$$E(m; r) = (g^r \mod p, g^m y^r \mod p)$$

where $r \in_r R$. For $m_1, m_2 \in M$ and $r_1, r_2 \in_r R$,

$$E(m_1, r_1) \odot_C E(m_2, r_2) = (g^{r_1}, g^{m_1} y^{r_1})(g^{r_2}, g^{m_2} y^{r_2})$$
$$= (g^{r_1+r_2}, g^{m_1+m_2} y^{r_1+r_2}) = E(m_1 + m_2, r_1 + r_2)$$

3.4. Commitment Schemes

Commitment is the important part of the cryptographic protocols. Damgard explains the commitment schemes clearly in [8]. Informally speaking, a commitment scheme consists of commit and reveal phases between two parties, called the sender and the receiver. In many cases, the protocols commit and reveal can be done in terms of a single algorithm, requiring no interaction between the sender and receiver at all.

Definition 3.1. *Let commit*

$$\{0,1\}^k \times \{0,1\}^* \to \{0,1\}^*$$

*be a deterministic polynomial time algorithm, where k is a security parameter. A **commitment scheme** consists of two phases between a sender and a receiver [23].*

- *Commit Phase: A protocol in which the sender commits to a value $x \in \{0,1\}^*$ by computing $C = commit(r, x)$ where $r \in_R \{0,1\}^k$, and sending C to the receiver. The receiver stores C for later use.*
- *Reveal Phase: A protocol in which the sender opens commitment $C = commit(r, x)$ by sending r and x to the receiver. The receiver computes $commit(r, x)$ and verifies that it is equal to the previously received commitment.*

Pedersen Commitment Scheme: Pedersen introduced a commitment scheme in 1992 [16]. Receiver chooses large primes p and q such that q divides $p^n - 1$, generator g of the order-q subgroup of $\mathbb{Z}^*_{p^n}$, he selects a random secret x from $\mathbb{Z}_q$, $y = g^x \mod p$. The parameters p, q, g, y are public and x is secret.

- *Commit:* To commit $m \in \mathbb{Z}_q$, sender chooses random $r \in_R \mathbb{Z}_q$ and sends $c = g^m y^r \mod p^n$ to the receiver.
- *Reveal:* To open the commitment, sender reveals m and r, receiver verifies that $c = g^m y^r \mod p^n$.

3.5. Threshold Cryptography

In cryptography, threshold is meant to distribute basic cryptographic schemes between a group of participant [20]. Each distributed part can be an algorithm or a key. Secret sharing is the main basis of threshold cryptography. In a *secret sharing scheme* there exist a dealer $\mathcal{D}$ and participants $\mathcal{P}_1, \ldots, \mathcal{P}_n$. Dealer and participants follow two protocols (distribution and reconstruction).

- *Distribution:* Dealer shares a secret s among the participants, each $\mathcal{P}_i$ obtains a share s_i for $1 \leq i \leq n$.
- *Reconstruction:* Secret is recovered by combining any qualified set of shares s_i. To reconstruct the secret, it is not necessary to obtain ***all*** participants' secrets (Shamir's (t, n)-threshold scheme). This means that a secret is distributed to n participants, in order to reconstruct the secret, only t of the participants are sufficient.

To decrypt an encrypted message, a number of parties exceeding a threshold is required to cooperate in the decryption protocol. The message is encrypted using a public key and the corresponding private key is shared among the participating parties. Let n be the number of parties in total. Such a system is called (t, n)-threshold, if at least t of these parties can efficiently decrypt the ciphertext, while less than t have no useful information.

3.6. Σ-Protocols

A Σ-protocol is a three round honest-verifier zero-knowledge proof of knowledge [9]. In this protocol, there are two players, a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. The aim is that $\mathcal{P}$ convinces an honest $\mathcal{V}$ of the truth of the argument without revealing any information about the statement. More formally, Σ-protocols satisfy the following properties:

- *Completeness:* If $\mathcal{P}$ and $\mathcal{V}$ are honest, i.e., they follow the protocol honestly, then protocol occurs with an overwhelming probability.
- *Special-soundness:* There exists a probabilistic polynomial time algorithm E (extractor) which given any $v \in \mathcal{V}$ and any pair of accepting conversations $(sn; \beta; \gamma)$ and $(sn; \beta'; \gamma')$ with $\beta \neq \beta'$ computes a secret/witness x satisfying $(v; x) \in R$.

  $$g^{\gamma} = sn \cdot y^{\beta}, g^{\gamma'} = sn \cdot y^{\beta'}$$

  When dividing with each other,

  $$g^{\gamma - \gamma'} = y^{\beta - \beta'} \Leftrightarrow y = g^{\frac{\gamma - \gamma'}{\beta - \beta'}}$$

Therefore, the secret value x can be obtained as $x = \frac{\gamma - \gamma'}{\beta - \beta'}$

- *Special honest-verifier zero-knowledge:* There exists a probabilistic polynomial time algorithm S (simulator) which given any $v \in \mathcal{L}_R$ and any challenge $c \in C$ produces conversations $(sn; \beta; \gamma)$ with the same probability distribution as conversations between honest $\mathcal{P}$ and $\mathcal{V}$ on common input v and challenge β where $\mathcal{P}$ uses any witness x satisfying $(v; x) \in R$. Furthermore, for $v \in \mathcal{V}/\mathcal{L}_R$, S is just required to produce arbitrary accepting conversations with challenge β;

  $\{(sn; \beta; \gamma) : r \in_R \mathbb{Z}_q; sn \leftarrow g^r; \gamma \leftarrow_q r + \beta x\}$
  $\{(sn; \beta; \gamma) : \gamma \in_R \mathbb{Z}_q; sn \leftarrow g^{\gamma} y^{-\beta}\}$.

  Given any arbitrary challenge β, the distributions are equal, then the simulated conversations are accepting.

4. Proposed i-voting protocol

In this section, we explain how the private key of the election and the other secret values are generated. Moreover, we describe the main infrastructures of the protocol. Finally, our main protocol will be described.

Our i-voting protocol consists of eight players. The voter's computer PC, Authentication Box AB, Control Box CB, Authority A, Bulletin Board BB are online players. The voter V, Counter C and Decryption Service DS are offline players in our protocol. The other components of system are the vote v, Thin Client TC, Terminal Server TS, Trusted Parties TP and Certification Authority CA.

4.1. Key Generation

Distributed key generation is the main component of the threshold cryptosystems. There are many solutions to the distributed generation of the private keys. Here in our solution, the private key is created by the shares of parties who may be a governmental institution, university and political parties which are independent from each other. We have t of them and any number of institutions less than t cooperating together cannot calculate the private key. The threshold scheme is perfect if knowledge of $t-1$ or fewer shares give no information regarding the private key. In our protocol, before the election period, the private and public key pair of the election are generated (one can use distributed key generation protocol where each party receives a share of the private key and all parties learn the public key of the election [2]). The private key x of the election is shared by trusted parties and the corresponding public key $y = g^x$ is public.

On the other hand, each voter needs the private and public key pair generated by CA to sign her vote. The voter's private key is identified to her ID-Card. The corresponding public key with owner ID are sent to AB module in the system to verify the signature of the voter. Moreover, in our protocol, the secret parameters a_1 belongs to AB, a_2 belongs to CB and a_3 belongs to C are generated by CA which satisfy $a_1 + a_2 = a_3^{-1} \mod p - 1$ (a_3 is invertible in $\mod p - 1$).

The aim of using parameters a_1, a_2 and a_3 is to increase the security of the system. We note that the parameter a_3 is not used for decryption as in the Norwegian protocol but these parameters bind the link of the overall process among the components AB, CB and C. These parameters are used to eliminate CB and/or A to produce any vote by itself. That is, they prevent the compromised case of CB and/or A. These masking operations also ensure that every vote passes over AB and CB.

Notice that these parameters are committed by using *Pedersen commitment scheme*. That is,

$u_1 = commit(r_1, a_1), u_2 = commit(r_2, a_2), u_3 = commit(r_3, a_3)$, where $r_1, r_2, r_3 \in_R \{0,1\}^k$.

4.2. Main infrastructures of our protocol

Before describing our protocol, we will highlight the main infrastructures of our protocol.

- **Voter's Secret Number** sn**:** As mentioned before, coercion is one of the most important problems for i-voting. To overcome this problem, we not only give opportunity to the voter cast her vote multiple times, but also protect the voter by giving to each voter a secret number sn which is specific for each voter. One of the most important findings of our work is to propose this secret number for coercion. Note that it should be certainly user-friendly (i.e., memorable number like for example four digits) since every voter have to use it during the vote submission. We note that the voter should not keep sn on a written note because of a possible coercion.
  Before the election, both secret number sn and ID number of each voter are given to C and this sn is also sent to the voter with SMS. During the vote submission, the system requires her sn and she enters it. The sn with the other parameters are transferred over the system till module C. We note that it is not a security concern to transport sn in an unencrypted form since coercer cannot learn whether this sn is real number or not. Next, C checks the received sn, and if sn is correct, the vote is tallied correctly; otherwise, C ignores this vote to be tallied without giving a warning to the voter. In each case, the votes casting attempt are given to BB. Therefore, the coercer sees all of them in BB and so not notice which one is the real vote.
- **Receipt Code:** In the Norwegian receipt code generation, if the ballot box B and the receipt generator R cooperate, they can obtain the private key of the election. We consider that, this assumption is strong and should be excluded from this protocol. To eliminate this deficiency, we use the hash value of the encrypted vote as a receipt code.
  In our receipt code, the privacy of the voters is still ensured even if any of our players cooperate. Moreover, our receipt code is generated with faster and lower cost. In our protocol, before the election, High Election Board (HEB) produces the receipt code paper more than the number of the voters. Receipt code paper consists of hash values of all possible encrypted votes and random number r used in ElGamal encryption algorithm. The sample of receipt code paper is presented in the Table 1. On the other hand, while casting a vote, PC give a receipt which includes the hash value of encrypted vote, h. If the voter's PC is enough secure, the voter can use this receipt instead of receipt code paper. Note that, in the verification process, the voter can use the first and last three digits of her hash value for the ease of understanding.
  Notice that the number of all possible encrypted votes is equal to the number of parties in the election since each voter has the specified unique random number used in ElGamal encryption algorithm. In this case, to satisfy privacy of the voter, the link between the voter and her random number r has to be unknown. For this purpose, the receipt code papers are safeguarded in a sealed envelope and should be anonymously sent to the voters. Before the election period, these papers are delivered to the election precincts in uniformly random order. During the election period, the voters get ran-

domly one of them from their election precinct or they are sent to the voters in independent safe channel at the same time. Thus, nobody knows the link between the voter and her random number. Notice that since the voters need to insert the random number r from her receipt code paper to PC, Quick Response (QR) code kind of a mechanism (e.g., from a smart phone) can be used to make this process more simple.

Candidates	$h = H(E(v,r)),\ r = 7B3FC48E9A86F9DE7AFD$
$PARTY_A$	$0CC175B9C0F1B6A831C399E269772661$
$PARTY_B$	$92EB5FFEE6AE2FEC3AD71C777531578F$
$PARTY_C$	$4A8A08F09D37B73795649038408B5F33$
$PARTY_D$	$8277E0910D750195B448797616E091AD$
$PARTY_E$	$7FC56270E7A70FA81A5935B72EACBE29$
$PARTY_F$	$8FA14CDD754F91CC6554C9E71929CCE7$
$PARTY_G$	$B2F5FF47436671B6E533D8DC3614845D$
$PARTY_H$	$2510C39011C5BE704182423E3A695E91$
$PARTY_I$	$DA564F38413A243E30E8C8C07FCCC5D8$
$PARTY_J$	$363B122C528F54DF4A0446B6BAB05515$
$PARTY_K$	$8CE4B16B22B58894AA86C421E8759DF3$

TABLE 1
Receipt code paper

- **Bulletin Board (BB):** BB is the online verification mechanism. After the vote is arrived to the tallying process, the hash value of the vote is sent to the voter's mobile with SMS by BB. We note that, to protect coercion, the voters are informed about their votes even if it is not counted. Moreover, the voters can verify from BB whether casted votes, even if not counted, are passed the encryption and masking processes over the system without any alteration. Thus, the voters become sure that their votes are transferred and counted correctly. The query is done with the hash value and their ID number. When the voter makes a query and if any record is found, the record includes h, sn and ID number. Wrong information in BB guarantees that at least one of the system components is compromised.

4.3. Main Protocol

Our i-voting protocol mainly consists of four steps: the vote submission, the vote transporting, the counting and announcement processes. We are now ready to describe these processes step by step for each casted vote. During these procedures, you can benefit from the diagram of our protocol in Figure 2 in Appendix.

Vote Submission: During the vote submission process, V and PC are active players. The main task of the PC is to encrypt the selected votes and sign encrypted votes.

- *Online process:*

1 V authenticates herself to the voting system using her ID card. She selects her candidate and inputs her secret number sn.

2 PC first encrypts her vote v with

$$E(v,r) = (g^r \mod p, g^v y^r \mod p) = c,$$

where r is the random number, $y = g^x$ is the public key of the election and x is the private key of the election. In addition, it calculates hash of encrypted vote, $h = H(c)$, and signs this hash value using the private key of the voter, $Sign = S(h)$. At this point, PC gives a receipt to V which includes the receipt code value h but neither voter's ID nor her voting choices.

Transporting Process: During the vote transporting process, PC, AB, CB and A are active online players. The main task of the AB is to authenticate the voter. The main task of the CB is to generate the receipt code and send it to the V and PC. The main task of the A is to multiply the vote pair coming from AB and CB. The other task of A is to export the stored data to C as offline periodically.

1 PC sends encrypted vote, its signature, users ID number, zero knowledge proof of its own

computation (ZK_{PC}) and secret number sn to AB.
$PC \xrightarrow{c,Sign,ID,ZK_{PC},sn} AB.$

2 AB verifies $Sign \rightarrow h$ using the public key of the voter and checks the proof ZK_{PC}. AB gives a unique number u to each vote so that the next module A can match the vote pairs coming from AB and CB. It also adds a sequence number s to each voter's vote in order to determine the final vote in C. (Actually, this sequence number shows how many times the voter casts a vote).

3 AB sends the zero knowledge of its own computations (ZK_{AB}), u, s and all the data received from PC except $Sign$ to CB.
$AB \xrightarrow{c,ID,ZK_{PC},ZK_{AB},sn,s,u} CB.$

4 CB checks the proofs ZK_{PC}, ZK_{AB}. Next, it computes $h = H(c)$ and sends it to V by using a safe channel (like SMS). CB also sends this hash data to PC during the submission process.
$CB \xrightarrow{h} V$ and PC.

5 The received h is better to be checked by PC. Moreover, the voter has opportunity to check the received hash from her receipt code paper. By doing those controls, the voter learns whether her vote is received by CB without any alteration or not. If the received h is true, she becomes sure that the vote is transmitted by AB truly and the vote is valid; otherwise, she realises whether her PC is compromised or any alteration is occurred in the system and so the vote is canceled.

6 If the vote is valid, AB and CB are masking the encrypted vote c with a_1, c^{a_1} and a_2, c^{a_2} respectively.

7 AB sends c^{a_1}, u, s, ZK_{AB} and the received data from PC except $Sign$ and c to A.
$AB \xrightarrow{c^{a_1},ID,ZK_{PC},ZK_{AB},sn,s,u} A$

8 CB also sends c^{a_2}, zero knowledge proof of its own computation (ZK_{CB}) and the received data from AB except c to A.
$CB \xrightarrow{c^{a_2},ID,ZK_{PC},ZK_{AB},ZK_{CB},sn,s,u} A.$

9 A verifies the proofs ZK_{PC}, ZK_{AB} and ZK_{CB}. Next, A matches the vote pairs by using the unique number u and multiplies the pair of masked encrypted votes coming from AB and CB. A also stores all these data in a database.

$$\begin{aligned} c^{a_1} \cdot c^{a_2} &= E(v^{a_1}, ra_1) \cdot E(v^{a_2}, ra_2) \\ &= E(v^{a_1+a_2}, r(a_1+a_2)) \\ &= (g^r, g^v y^r)^{a_1+a_2} = c^{a_1+a_2} \end{aligned}$$

- *Offline process:*

10 The masked data $c^{a_1+a_2}$, zero knowledge proof of its own computation (ZK_A) and received data except c^{a_1}, c^{a_2} and u are periodically exported to C by A (offline).
$A \xrightarrow{c^{a_1+a_2},ID,ZK_{PC},ZK_{AB},ZK_{CB},ZK_A,sn,s} C$

11 C verifies the proofs ZK_{PC}, ZK_{AB}, ZK_{CB}, ZK_A and decrypts $c^{a_1+a_2}$ by using a_3.

$$(c^{a_1+a_2})^{a_3} \mod p \rightarrow (c^{a_3^{-1}})^{a_3} = c \mod p.$$

12 Next, C checks the sn; If it is correct, the vote is valid; otherwise, invalid.

13 For each vote, the value h is calculated by C and sent to BB with voter's ID and sn. The data transfer from C to BB is done offline by using an external disc.
$C \xrightarrow{h,ID,sn} BB$

- *Online process:*

14 BB sends h and sn to the voter by SMS to guarantee her vote comes to the counter process. Moreover, V verifies whether h and sn are true.
$BB \xrightarrow{h,sn} V.$

15 Each voter can also check her vote whether it reaches in BB or not. BB is listing hash of all

casted votes and sn (it is not matter whether if the voter inputs right or wrong secret number to the system). Each voter expects to see every h and sn in BB. If they do not see it, they should inform the election authority.

Counting and Announcement Processes: During these processes, C and DS are active offline players. When the vote submission period ends, if there does not exist any problem, counting process can be started.

- *Offline process:*

1 When the election is over, C permutes the list of valid final votes (votes received which have the greatest sequence number s and correct secret number sn). By using the homomorphic property of ElGamal encryption, it gets l of them randomly (This limit is done because of the *discrete logarithm problem*) and it multiplies them:

$$E(v_1, r_1) \cdot E(v_2, r_2) \cdot \cdots \cdot E(v_l, r_l)$$
$$= E(v_1 + v_2 + \cdots + v_l, r_1 + r_2 + \cdots + r_l)$$

2 The bunch of votes are sent to DS which is far away from C by using a one way filter which does not let any traffic from DS to C. Thus, the private key of the election is kept away from the encrypted votes.

$$C \xrightarrow{E(v_1+v_2+\cdots+v_l, r_1+r_2+\cdots+r_l)} DS$$

3 After at least t TP in DS gather and they construct the private key x, they decrypt the encrypted results:

$$D_x(E(v_1 + v_2 + \cdots + v_l, r_1 + r_2 + \cdots + r_l))$$
$$= g^{v_1+v_2+\cdots+v_l}.$$

4 Finally, the election result is announced.

4.4. Solving output of the additive ElGamal decryption

We now explain how the output of the additive Elgamal decryption ($g^v = g^{v_1+v_2+v_3+\cdots+v_l}$) can be solved. In general, it is hard to compute v from g^v where g is generator of multiplicative cyclic group G. Taking into account this restriction, we can construct reasonable vote format in order to obtain suitable v value. In our vote format, v consists of 40-bit which is less than $2^{40} - 1$. Therefore, this discrete logarithm problem can be easily solved with *Index calculus algorithm* or such other algorithms. We note that *lookup table* can also be used instead of compute v from g^v repeatedly. During the solving process, parallel computation has to be done in our protocol to announce the result of the election immediately.

5. Security Analysis

5.1. Security Requirements of e-voting system

In this section, we define the required security issues related to privacy and accuracy of voting in an e-voting protocol. We also show that our protocol satisfies these requirements.

- *Coercion-Resistance:* It means that the voter cannot cooperate with a coercer to prove that she casts her vote in a certain party. It is a fundamental and strong property of e-voting systems. It can be done not only by force or threat, but also with personal relationships. It can manipulate the election results and harm the voters freewill. Thus, the effect of coercion on the election results cannot be ignored and some precautions should be taken. In order to prevent coercion, the voters, in the previous works, are allowed to cast their votes more than once but only the final vote is counted. In addition, we propose to use the secret number sn while casting a vote. Note that if there is a coercer, the voter can easily enter a fake four digit number different from the real number sn.

- *Integrity:* Integrity means that votes are correctly casted, transported over the players and tallied at the end of the election. That is, election results must be correctly announced from casted votes. In our protocol, each player prove the correctness of their computations by using *Zero-knowledge proof algorithm*. Moreover, the voter can verify the integrity of her vote in the transporting and counting phases in our system by controlling the receipt code with SMSs which are sent by both CB and BB. If the received receipt value is the same on the receipt code paper, this means that the integrity is preserved. If some votes are being transferred and tallied dishonestly, then the voter's confidence to the ruling authority diminishes. In this case, the voter should inform the election authority.
- *Vote privacy:* It is related with the voters freewill; without privacy, the voters can be divided into groups or maybe some of them can be alienated from the community. Privacy is satisfied by additive homomorphic encryption method. In our system the votes are encrypted by PC and transferred in encrypted form and in any part of the system the decryption key is not released. After votes have been tallied, in order to announce the results, only the bunch votes (homomorphically added version of l votes) are decrypted. Therefore, at each step there is no authority who can read the voter's will.
- *Receipt freeness:* Receipt freeness roughly means the voter cannot prove to an adversary that she casts her vote in a particular manner. In other words, vote-buying is not possible. In our system, the voters do not have any chance to prove vote buyers (or coercers) that she casts her vote to a specific candidate. Because the voter may cast her vote using random number given by the system instead of using random number on her receipt code paper. In that case, the vote buyer or coercer cannot be sure whether the voter uses receipt code paper or not. Therefore, our system satisfies receipt freeness.
- *Individual and Universally Verifiability:* Our system provides individual and universally verifiability. Each voter can check that her encrypted vote is transported, counted and tabulated correctly in final tally. Moreover, someone can check whether any encrypted vote reaches in BB by using voter's ID and hash.

5.2. Security of the main infrastructures

We now describe the security issues of the main infrastructures of our system.

- *Security of Receipt Code:* Assume that coercer and/or the vote buyer wants to know the link between the ballot and hash of its encrypted form. If the voter trusts the system and her PC, she does not have to use receipt code paper. If she trusts her PC but not the system, then she can verify correctness of the system using the receipt on her PC instead of her receipt code paper. Therefore, since the voter casts her vote using the random number given by the system, it is not possible to find the link between the ballot and hash of its encrypted form. If she does not trust her PC, then she should use the receipt code paper to verify whether her vote has been successfully received. In order to resist to coercion, the voter may casts a vote multiple times using her receipt code paper. In this case, it is important to get a new receipt code paper from the election center to prevent the correlation between the ballot and hash of its encrypted form. Therefore, in all case, any malicious attacker cannot obtain any relation between the ballot and hash of its encrypted form.

  On the other hand, we assume that HEB wants

to know the link between the voter and her receipt code. To do this, HEB sends the predetermined random number in receipt code paper to the specific voter. Suppose that there exist about 100 voters at specific region and therefore more than 100 receipt code papers exist. The voter can select randomly one of them among the bunch of papers. The probability of identifying the link between specific voter and specific random number is about $\frac{1}{100}$ which is ignorable. We also highlight that the verification of the correctness of the system using the receipt code is not compulsory, about 5% of the voters is sufficient to guarantee the correctness of the system. If the voter does not trust i-voting system and/or her PC then she should use receipt code paper to prevent a possible malicious action. On the other hand, each encrypted vote is different from the other since different random numbers are used in ElGamal scheme for each voter. That is, each hash value is different from the other. Therefore, since two voter casting the same candidate have different receipt values, none of them can guess the other's vote.

- *Security of Secret number sn:* As mentioned before, each voter's secret number is sent to her mobile phone via SMS before the election. Let's assume that coercer has voter's mobile phone for a short time. In this case, there exists a solution to satisfy secrecy of her sn. This number is sent to the voter at an unpredictable time (e.g., any time in a month) before the election. Coercer has to keep the mobile phones for a long time which is not realistic. On the other hand, the coercer may want to see the voter's secret number from her phone. We note that the voter can delete first real SMS and can always request a new SMS for a dummy sn to show a fake secret number to the coercer (not to show the real one to a coercer). Thus, the system sends a new fake secret number. Without legal recourse, always the first secret number is valid and used in the tallying process; however, fake ones ignore the votes from tallying.
  As stated before, the secret number sn is transported over the system in an unencrypted form which is not a security concern. However, in order to avoid network kind of attacks (like cyber attacks), sn can be hidden by using $(2, 2)$-threshold scheme between the voter V and module C. Namely, the user enters sn and the system encrypts it using a $(2, 2)$-threshold public key cryptosystem. The encrypted value together with partial decryption by V is sent to C. C also partially decrypts and finds sn. Therefore, sn is transferred securely to C and anyone except C cannot see/modify it.
- *Security of BB:* Let's assume that there exists a coercer who can see casted-votes in BB. Here, there might be a possible attack scenario that any coercer can check the receipt codes of the voter from BB one by one. To solve this problem, the voter may cast more than one candidate from the list but only the final one is valid. Therefore, if coercer wants to check from BB, he sees more than one casted votes, but he can never learn real choice of the voter.

5.3. Compromised Players

We now show that our protocol is secure by considering the correctness of our system and privacy of the voters in each compromised module.

Theorem 5.1. *Our protocol is correct in the case that PC is compromised.*

Proof: If PC is compromised, it may change the vote or the secret number sn. If the vote is changed, hash value of the encrypted vote is also

changed. However, the receipt code sent by CB can help the voter to detect whether her vote has been changed by PC. Furthermore, if sn is changed, the voter can also detect this case during the search from BB. We note that, if the voter does not trust her PC, she can use the TC at election center during the election time. □

Theorem 5.2. *Our protocol is correct in the case that AB is compromised.*

Proof: The compromised AB can generate a vote on behalf of a voter. In that case, this fake vote is sent to CB. CB then sends a notification (e.g., SMS) to the voter. If the voter receives an SMS although she does not cast a vote, the voter notices that there is an attack to the system and a fake vote is casted. Therefore, the notification sent by CB prevents a possible attack by AB. □

Theorem 5.3. *Our protocol is correct in the case that CB is compromised.*

Proof: Assume that a compromised CB generates a fake vote and masks it then send to A. On the other hand, AB does not send the corresponding masked value to A since AB is a honest model. A checks the unique number u of the vote coming from CB. The generated vote is canceled since there is no match. Masking process is performed to protect the system from a compromised CB. □

Theorem 5.4. *Our protocol is also correct in the case that AB and CB are compromised and cooperate.*

Proof: If AB and CB are compromised and cooperate, they can generate a fake vote. Notice that their fake generated votes are listed in BB. However, the voter notices this malicious cooperation since BB also sends h via SMS to the voter. If the voter receives an SMS from BB although she did not cast a vote, the voter is immediately aware of a malicious case. We highlight that the privacy of voters is violated in the Norwegian protocol in the case that AB and CB are malicious, but in our system any malicious player cannot violate the privacy of the voter. Notice that masking operations also ensure that every votes pass over AB and CB. □

6. Complexity Analysis

Our i-voting protocol contains sub-protocols between the components. To analyse the complexity of the system, all the sub-protocol steps should be examined separately. Assume that the order of cyclic group G is p, the number of submitted votes is n and the number of votes in a bunch is l. Also, assume that all submitted votes are valid.

The complexity of each component: The computational cost of our protocol can be described as follows. Note that we only count the expensive asymmetric operations since symmetric encryptions and hash functions have ignorable complexity.

- In PC; The vote is encrypted by using ElGamal encryption which costs 3 modular exponentiations and 1 modular multiplication for each submission. Each encrypted vote is hashed by a hash function which is a linear function. Hash value of the encrypted vote is signed with ElGamal which costs 1 modular exponentiation and 2 modular multiplications for each submission.
- In AB; Signature is verified, which costs 3 modular exponentiations. The encrypted vote is masked with a_1, which costs 1 modular exponentiation.
- In CB; The encrypted vote is hashed and masked with a_2, which costs 1 modular expo-

nentiation.

- In A; For each vote pair, 1 multiplication occurs since they are multiplied each other.
- In C; The masking process is repeated with a_3, which costs 1 modular exponentiation.
  Until this point, every multiplication and exponentiation occurs for each vote submission. After this point, l votes are grouped and multiplied which costs $(l-1)$ modular multiplications. This process is repeated $\frac{n}{l}$ times. It costs approximately n modular multiplications.
- In DS; The vote bunches are decrypted. In this part, there exists $\frac{n}{l}$ ElGamal decryption process. Each decryption process costs 1 modular exponentiation, 1 modular inversion and 1 modular multiplication.

As a result, for n submitted votes, $10n$ modular exponentiations and $4n$ modular multiplications are used in the encryption process. In counter process, n modular multiplications occur. Additionally, in decryption process, $\frac{n}{l}$ modular exponentiations, $\frac{n}{l}$ modular inversions and $\frac{n}{l}$ modular multiplications occur. The required number of operations for n submitted votes are presented in the Table 2.

	# multiplications	# exponentiations	# inversions
Encryption	$4n$	$10n$	$-$
Counter	n	$-$	$-$
Decryption	$\frac{n}{l}$	$\frac{n}{l}$	$\frac{n}{l}$

TABLE 2
Computational complexity

Notice that the complexities of the modular multiplications, inversions and exponentiations in G of order p are $O(log^2 p)$, $O(log^3 p)$ and $O(log^3 p)$, respectively. Therefore, computational complexity of the system is approximately;

$$O((10n + \frac{2n}{l})log^3 p + (5n + \frac{n}{l})log^2 p) \approx O(nlog^3 p)$$

Also for each vote in our system, 4 zero-knowledge proofs and 15 zero-knowledge proof verifications are used.

Communication Complexity: In our protocol, the voters who cast a vote over their PC need their web browsers to connect the vote casting system. For them to use i-voting system, 4 Mbps connection capacity is enough [22] since vote file does not have any difference from an e-mail of a simple text file. On the other hand, in our protocol, each voter has also chance to submit their ballots using a TC located in election centers. In that case, communication between TC and TS should be taken into account. This communication can be done using remote desktop services protocol by accessing TS using SSL on port 443 [7], this communication protocol is used to encapsulate Remote Desktop Protocol (RDP) within HTTP over an SSL connection. For this kind of communication, 56 Kbps connection capacity is adequate at the client side.

Database Issues: Our voting system is an example of a distributed system composed of modules which are integrated to work together. Some of those modules such as CB or BB have their own database and they communicate to each other in order to accomplish voting process. For a database to operate efficiently, normalization is important [15]. Normalization decreases redundancies and anomalies moreover it is important to increase efficiencies. When implementing the infrastructure and creating databases, all the databases should meet the requirements of third normal form.

7. Conclusion

I-voting is a growing trend for some countries which are interested in increased voter participation both in the country and overseas. Estonia, Norway and France are a few practical examples that used

i-voting system in real elections. In this paper, we try to construct an efficient E2E secure i-voting system. In our protocol, by using the homomorphic property of ElGamal, all casted votes are gathered, transported and tallied in an encrypted form so that no attacker has a chance to see the wish of the voters. Moreover, we try to prevent the voters from coercion with the help of secret number sn. It is crucial to construct a voting system in which security, transparency and privacy are satisfied. As a result, we present a new i-voting system satisfying the required security issues.

Acknowledgments

Preliminary version of this study has been presented as a poster at CryptoDays conference, in June 2013, Tubitak, TURKEY. Moreover, the second version of this work [1] has been presented at conference ISCTURKEY 2013, ANKARA, TURKEY.

Appendix

Our protocol is depicted in Figure 2.

References

[1] A. Sınak, M. Sabır Kiraz, S. Özkan and H. Yıldırım, *A Secure Internet Voting Protocol Based on Homomorphic Encryption*, ISCTURKEY 2013, Proceedings of 6th International Conference on Information Security and Cryptology, pp.142-148, Sep 20-21, 2013.

[2] B. Schoenmakers, *Lecture Notes over Cryptographic Protocols*, February 2013, http://www.win.tue.nl/ berry/2WC13/LectureNotes.pdf

[3] D. Chaum, *Untraceable electronic mail, return addresses, and digital pseudonyms*, Commun. ACM, 24(2), pp.84-88, 1981.

[4] Douglas W. Jones, *Early Requirements for Mechanical Voting Systems*, Atlanta, 2009.

[5] E. Maaten, *Towards Remote e-voting: Estonian Case*, In Electronic Voting in Europe-Technology, Law, Politics and Society, volume 47 of LNI, Bregenz, Austria, July 7th-9th, pp.83-100, 2004.

[6] F. Brandt, *Efficient Cryptographic Protocol Design Based on Distributed ElGamal Encryption*, Springer-Verlag, LNCS, Vol.3935, pp.32-47, 2006.

[7] http://sharepointgeorge.com/2009/remote-desktop-services-windows-2008-r2-part-2-gateway/

[8] I. Damgard, *Commitment Schemes and Zero-Knowledge Protocols*, Springer-Verlag, LNCS, Vol.1561, pp.63-86, 1999.

[9] N. Smart, *Cryptography: An Introduction*, Mcgraw-Hill Publication, Dec. 2004, isbn: 0077099877.

[10] J. Barrat i Esteve, B. Goldsmith and J. Turner, *International Experience with E-Voting*, International Foundation for Electoral Systems, 2012.

[11] J. Barrat i Esteve, B. Goldsmith and J. Turner, *Speed and Efficiency of the Vote Counting*, Process, 2012.

[12] K. Gjosteen, *Analysis of an Internet Voting Protocol*, March 9, 2010, http://eprint.iacr.org/2010/380.pdf

[13] K. Peng, *Theory and Practice of Secure E-Voting Systems*, Theory and Practice of Cryptography Solutions for Secure Information Systems, pp.428-459, 2013.

[14] M. J. Morshed Chowdhury, *Comparison of e-voting schemes: Estonian and Norwegian solutions*, 2010.

[15] M. Tamer Ozsu, P. Valduriez, *Principles of Distributed Database Systems*, Third Edition, Springer, 2011.

[16] M. Wiik Oberg, *Improving the Norvegian Internet Voting Protocol*, June 2011, http://daim.idi.ntnu.no/masteroppgaver/005/5823/

[17] M. Hirt and K. Sako, *Efficient Receipt-Free Voting Based on Homomorphic Encryption*, Springer-Verlag, LNCS, Vol.1807, pp.539-556, 2000.

[18] M. Ali Bingol, F. Birinci, S. Kardas and M. Sabır Kiraz, *Norwegian Internet Voting Protocol Revisited: Security and Privacy Enhancements*, International Conference BulCrypt, Sofia, Bulgaria, September 2012.

[19] P. L. Sven Heiberg and J. Willemson, *On Applying i-voting for Estonian Parliamentary elections*, Springer-Verlag, LNCS, Vol.7187, pp.208-223, 2012.

[20] R. Canetti, S. Goldwasser, *An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack*, Springer-Verlag, LNCS, Vol.1592, pp.90-106, 1999.

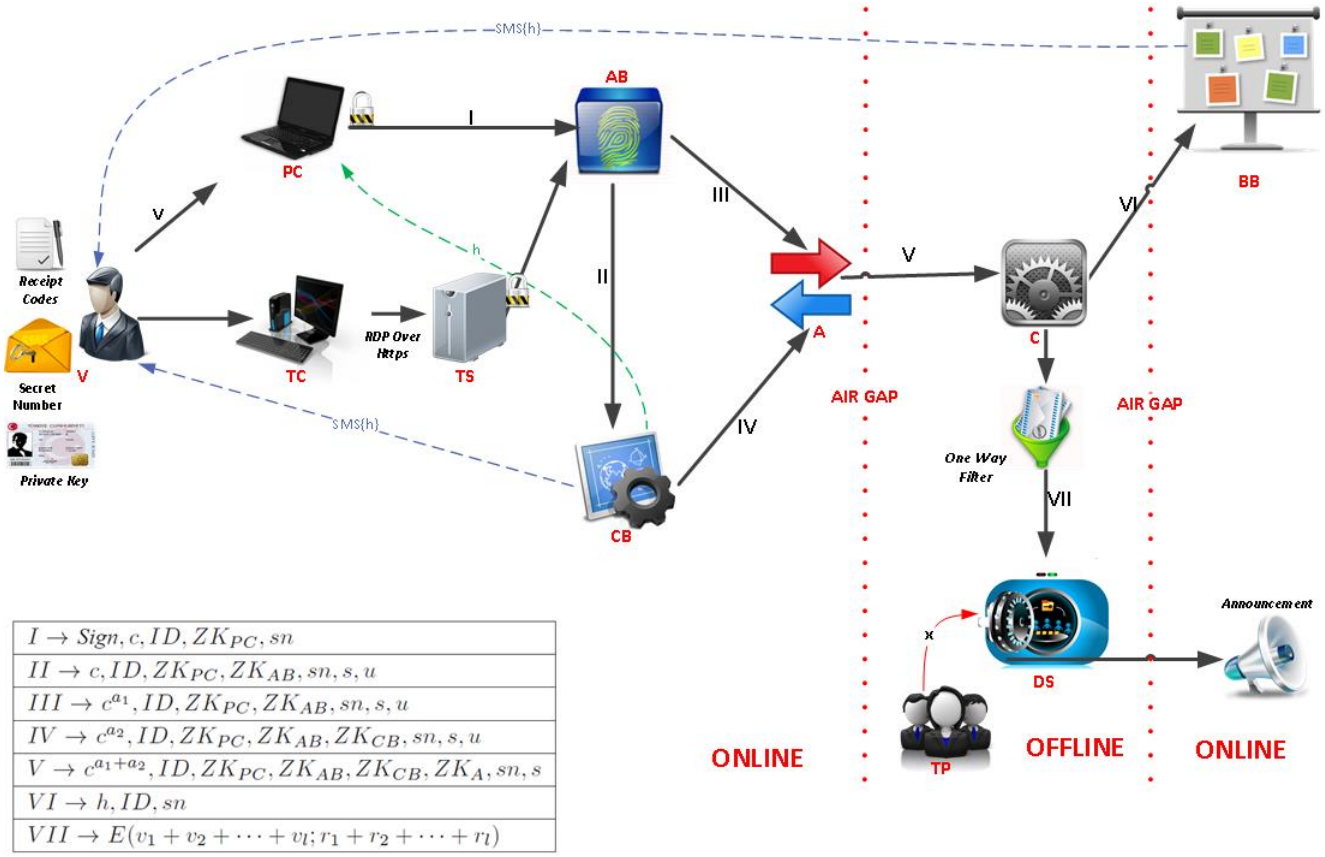
[21] S. Goldwasser, S. Micali and C. Rackoff, *The Knowledge Complexity of Interactive Proof Systems*, SIAM Journal on Computing, Vol.18, No.1, pp.186 - 208, 1989.

[22] J. D. Saunders, C. R. McClure, L. H. Mandel, *Broadband applications: Categories, requirements, and future frameworks*, 2012.

[23] S. Halevi, *Efficient Commitment Schemes with Bounded Sender and Unbounded Receiver*, Springer-Verlag, LNCS, Vol.12, No.2, pp.77-89, 1999.

[24] T. ElGamal, *A Public Key Cryptosystem and A Signature Scheme Based on Discrete Logarithms*, IEEE Transactions on Information Theory, Vol.31, No.4, pp.469–472, 1985.

[25] The Estonian National Electoral Committee, *General Description of the E-Voting System*, 2010.



$I \rightarrow Sign, c, ID, ZK_{PC}, sn$
$II \rightarrow c, ID, ZK_{PC}, ZK_{AB}, sn, s, u$
$III \rightarrow c^{a_1}, ID, ZK_{PC}, ZK_{AB}, sn, s, u$
$IV \rightarrow c^{a_2}, ID, ZK_{PC}, ZK_{AB}, ZK_{CB}, sn, s, u$
$V \rightarrow c^{a_1+a_2}, ID, ZK_{PC}, ZK_{AB}, ZK_{CB}, ZK_A, sn, s$
$VI \rightarrow h, ID, sn$
$VII \rightarrow E(v_1 + v_2 + \cdots + v_l; r_1 + r_2 + \cdots + r_l)$

Fig. 2. Our Internet Voting Protocol.