An Improved Internet Voting Protocol

Mehmet Sabir Kiraz[1] , Süleyman Kardaş[1,2],
Muhammed Ali Bingöl[1,3], and Fatih Birinci[1]

[1] TÜBİTAK BİLGEM–National Research Institute of Electronics & Cryptology , Gebze, Kocaeli, Turkey

[2] Sabanci University, Istanbul, Turkey

[3] Istanbul Technical University, Institute of Science and Technology, Istanbul, Turkey

Abstract. Norway is going to experience an Internet voting scheme in September 2011 for local governmental elections, targeting a comprehensive Internet voting system in 2017 for national election. This protocol is strong from several aspects. First of all, it resists against malicious voter's computers. Namely, an honest voter will be aware of a malicious behavior caused by the computer during the entire voting procedure. However, the security of the protocol depends on the assumption that the players (organizations) are completely independent and reliable, and the receipt codes are sent to the voters securely.
In this work, we take a closer look at the Internet voting protocol and investigate the followings:

- The privacy of voters are compromised if there is a cooperation between the players Ballot Box (BB) and Receipt Generator (RG) since the private key of Decryption Service (DS) can be obtained by the two former players. To prevent this possible issue, we propose an improved protocol without adding additional players.
- To verify the correctness of the overall protocol two additional channels are used where receipt codes are sent to the voters over the pre-channel (e.g., postal service) and also sent over the post-channel (e.g., SMS). However, if a voter holds both SMS and the paper of receipt codes at the same time, he can reveal his/her vote even after the election. To overcome this issue, we propose a new method where the SMS is used only as a notification message, and an additional phone call is used for the complete verification of the vote.
- The reliability of the Norwegian scheme is based on the correctness of the receipt codes that are sent to the voters over a secure pre-channel. However, if the printed receipt codes are falsely generated (or falsely printed) or the pre-channel is not completely secure, a vote can be counted for different candidates without any detection. In order to prevent this problem, in our protocol, the voters also take a part in the verification of the receipt codes before the vote casting protocol.

Keywords: Internet voting, Voting privacy, Threshold cryptography, Homomorphic encryption

1 Introduction

Electronic voting is getting more and more popular in the world. There are many advantages over the traditional paper-based systems like security, reliability, convenience, mobility, tally speed, less cost, and flexibility if the system is carefully designed. Up to now, cryptographers have been trying to improve the feasibility of voting systems to satisfy the security properties like privacy, correctness, anonymity, verifiability and receipt-freeness.

Developing an electronic voting system is known as one of the most difficult problem for cryptographers since it involves several research areas like sociology, psychology, politics, laws, information technology and security. It is also rather interesting and unique problem in cryptology since malicious behavior can be both from insider and outsider threats: for example, the system can cheat voters, voters can cheat systems and coercers can affect voters. Furthermore, the proposed system must be understandable and usable by the entire voting population, regardless of age or disability. Voters in general do not have the computing power and expertise. Therefore, the proposed e-voting systems should also be user-friendly, understandable and scalable. If the expected security requirements are also satisfied then it can be a great improvement over the traditional paper-based systems.

Electronic voting can be described with two different methods. First, *Electronic Voting Machine* (EVM) which refers to the use of any electronic device [10]. This may be a specialized voting machine in a voting booth with touch screens or a stand-alone PC specially designed for this purpose in a voting kiosk. The second one is *Electronic distance voting*, goes a step further than EVM in the sense that it implies the electronic registration, casting and counting of votes from different locations [10]. It typically allows the voter to use a more generic technology such as interactive digital TV, telephone, *Short Message Service* (SMS) or the Internet, to cast their vote from any preferred place. Electronic distance voting via Internet is convenient, quick, easy and an accurate voting process.

Estonia is the first country that is currently using the Internet for general elections since 2005 [2]. Internet voting is particularly attractive to those voters who spend considerable time to reach the polling station. For example, about half of the Internet voters in Estonia indicated that they would have spent half an hour or more to reach the polling station. Many of these voters might not live in their official residence, either living in another place or abroad. It is good to know that Estonia already successfully managed the Internet election project.

Norway will be the second country which will use the Internet voting for the local governmental elections in September 2011 [1]. The cryptographic protocol to be used in Norway is designed by Scytl, a Spanish electronic voting company. Norway also benefits from the Estonian expertise and improves their system against a stronger adversarial model in which the voter's computer is assumed to be malicious. Unlike the Estonian protocol, the Norwegian protocol preferred to use two additional channels: pre- and post-channel which are postal service and SMS respectively. The aim of using different channels is to prevent a ma-

licious computer attack, under the assumption that a secure and authenticated pre-channel is unlikely to be controlled by the same attacker that controls the computer. Before the voting process, voters receive the printed integrity check codes by the postal service, which is a table with candidate names, identification numbers and receipt codes. During the voting days, the voter casts his vote to a computer, which encrypts the ballot and submits it to a *Ballot Box* (BB). To prevent coercions, the voter is able to cast multiple votes, where the final one will be counted. BB and a *Receipt Generator* (RG) together compute the receipt codes for the casted vote and sends it to the voter via SMS. The voter can verify his vote by checking SMS against the printed receipt codes. Once the election time is over, the casted ciphertexts are decrypted by a *Decryption Service* (DS) by using re-encryption mixnet for creating integrity and anonymity. An *Auditor* verifies the correctness of the entire process. The security of the system depends on the assumption that BB and RG cannot be both compromised and cooperate [8, 9].

1.1 Our contributions

In this paper, we focus on the Internet based voting protocol which will be tried out in Norway in September 2011. We investigate the protocol from both theoretical and practical points of view. The security of the protocol is based on some strong assumptions that require trust to organizations. However, these assumptions could make the protocol less convenient for some countries where the societies are more septic about their vote privacy since the trust level of these organizations is low. We propose our solutions to mitigate those issues. The contributions of this paper could be summarized as follows.

- The Internet voting protocol that will be used in Norway preserves vote privacy even if only one of BB and RG is compromised. However, it assumes that both BB and RG cannot be compromised and cooperate, otherwise, the election privacy will be lost even if the key of DS is not revealed. This may leave question marks in people's minds that decreases the credibility of the system. Motivated by this problem, we study the simultaneous corruption of BB and RG. We enhance the protocol that removes the former assumption without adding a new organization. We show that the modified protocol provides the vote privacy against possible internal collaborations.
- Having both the printed receipt codes and SMS might be a potential threat for vote privacy. A voter can show his vote even after the elections. This is not believed to be a big issue since it is not definite that this vote is indeed the final one (because a voter can re-vote). However, this might be a significant issue in practice when it is still possible to show the voters' intention. In our protocol, we modify the verification part where we utilize the SMS only as a notification mechanism and propose to use a phone call for the complete verification of the votes. This prevents (intentional or unintentional) violation of vote privacy.

– We also show that either if the receipt codes are falsely generated (or falsely printed) or the pre-channel is not secure, then the voters can be fooled easily with no detection. In this case, the confidence of the society could rapidly disappear. In order to avoid this concern, we propose that in addition to the printed receipt codes, the trusted party also generates the non-interactive zero knowledge proof of knowledge of the ballots. This provides voters to ensure the correctness of the receipt codes before casting their votes, and the voter can detect any malicious behavior during the vote casting even if the pre-channel is not secure and his computer is malicious.

1.2 Organization

In Section 2, we give a brief description of the underlying cryptographic primitives of the proposed protocols. In Section 3, we first give a succinct description of the Norwegian cryptographic protocol and show its possible weaknesses in practice. We also provide our quick solutions to these weaknesses. In Section 4, we propose our new protocol which avoids the possible cooperation between BB and RG. In Section 5, we give an informal security analysis of our protocol. Section 6 concludes the paper.

2 Preliminaries

We now briefly describe the underlying cryptographic primitives of the protocols. Given a public key encryption scheme, let $\mathcal{M}$ denote its message or plaintext space, $\mathcal{C}$ the ciphertext space, and $\mathcal{R}$ its randomness. Let $c = E(m; r)$ depict an encryption of m under the public key pk where r is a random value. Let sk its corresponding private key, who allows to the holder retrieve a message from a ciphertext. The decryption is done with the private key sk as shown $m = D(c)$.

Threshold cryptosystem. In a (t, n)-threshold cryptosystem there are n participants in total. They want to distribute the decryption power up-to the agreement of any subset of (at least) t of them. Informally speaking, each participant holds a 'share' of the secret key, the overall secret key is somehow reconstructed to let them recover the message in a given ciphertext. To encrypt, there is a public key that is used as in a regular public key scheme.

More formally, let $P_1, \ldots, P_n$ be the participants. We define a (t, n)-threshold encryption scheme to be the public key with the following three phases:

– In the *key generation* phase, each participant P_i will receive a pair (pk_i, sk_i), where pk_i and sk_i are thought as *shares* of the public and secret key, respectively. Then the overall public key pk is constructed by *combining* the shares. Finally pk is broadcast to allow anyone to encrypt messages in $\mathcal{M}$. The shares of this public key are also broadcast which allow all parties to check the correctness of the decryption process.

– The *encryption* phase is done as in any public key encryption cryptosystem. If $m \in \mathcal{M}$ is the message, a (secret) random value r from $\mathcal{R}$ is chosen and $c = E(m; r)$ is broadcast.
– In the *threshold decryption* phase, given that t (or more) participants agree to decrypt a ciphertext c, they follow two steps. Firstly, each participant produces a decryption share by performing $S_{i_j} = D_{sk_{i_j}}(c)$, $j = 1, \ldots, t$. After broadcasting S_{i_j}, they all can apply a reconstruction function $\mathcal{R}$ on these shares so that they can recover the original message by performing $m = \mathcal{R}(S_{j_1}, \ldots, S_{j_t})$ where $P_{j_1}, \ldots, P_{j_t}$ represents the group of t participants willing to recover m.
  To withstand with the malicious adversaries, in the first step, parties have to prove that S_{i_j} was computed using the share of the secret key sk_{i_j} corresponding to the public value pk_{i_j}.
– In the case of a (t, n)-threshold scheme, the additional requirement is that if less than t parties gather their correct shares of the decryption of a given ciphertext, they will get no information whatsoever about the plaintext.

In our voting protocol, we use (2,2)-threshold cryptosystem between BB and RG where both players must cooperate in order to decrypt.

Homomorphic cryptosystem. A public key encryption scheme is said to be multiplicatively homomorphic if given $c_1 = E(m_1; r_1)$ and $c_2 = E(m_2; r_2)$ it follows that $c_1 c_2 = E(m_1 m_2; r_1 + r_2)$.

As a consequence, it is also true that $E(m; r)^s$ is equal to $E(m^s; rs)$ for a known integer s. Another consequence of these properties is the re-randomization of encryptions, by observing that $E(m; r)E(1; r')$ is a new encryption whose plaintext is again m (and its randomness is $r + r'$). In the proposed protocols, ElGamal scheme is utilized as a multiplicatively homomorphic cryptosystem [5]. Also, re-randomization is being used to be able to shuffle the encrypted ballots.

Σ-protocols. A Σ-protocol for a relation $R = (v; w)$ is a three-move (commit-challenge-response) protocol between a prover and a verifier. Both parties know the common input v, and the prover has a witness w as private input, where $(v; w) \in R$. Informally a Σ-protocol is a zero knowledge proof of knowledge for relation R which satisfies special soundness and (special) honest-verifier zero-knowledge. Namely, the prover convinces the verifier not only of the validity of a statement, but also that it possesses the witness w for the statement. See [4] for further details. The Fiat-Shamir technique [7] is a famous trick for making such a protocol non-interactive zero knowledge in the random oracle model, while preserving the security of the protocol in a practical manner [6].

As in the Internet voting protocol that will be used in Norway, we will use the fact that for homomorphic ElGamal encryptions there are efficient Σ-protocols for the relation $R = \{(e; m, r) : e = E(m, r)\}$, proving knowledge of the message m and randomness r for a given encryption $e = E(m, r)$. In our protocols, since the voter computes two distinct encryptions of the same vote, we also need to ensure equality-composition of Σ-protocols for

the relation $R_{\mathsf{AND}} = \{(e_1, e_2; m, r_1, r_2, k) : e_1 = E(m, r_1) \bigwedge e_2 = E(m^k, r_2)\}$. We also note that OR-composition Σ-proofs is important to use in our protocol to prevent a malicious voter submitting a fake ballot which is not in the eligible candidate list. Therefore, the voter should also prove to BB that his vote $f(v_j)$ (where f is a public encoding function and v_j is the j-th vote) is one of the candidate in $\{f(v_1), \ldots, f(v_{k_{max}})\}$ where k_{max} is the number of candidates i.e., Σ-protocols for the relation $R_{\mathsf{OR}} = \left\{(e_j, e'_j, \{m_j\}_{j=1}^{k_{max}}; r_j, r'_j, k_j) : \bigvee_{j=1}^{k_{max}} \left(e_j = E(m_j, r_j) \bigwedge e'_j = E(m_j^{k_j}, r'_j)\right)\right\}$.

3 The Internet voting protocol and its potential weaknesses

In this section, we first briefly describe the Internet voting protocol which is going to be experienced in Norway. Next, we point out the practical issues of the protocol and propose our quick solutions. For more details of the Norwegian protocol we refer to [8].

3.1 Protocol description

There are three players (organizations) in this system which are assumed to be independent (i.e., *Ballot Box* (BB), *Receipt Generator* (RG) and *Decryption Service* (DS)). The protocol consists of three phases: key generation, vote casting and vote counting. Denote $\{a_j\}_{j=1}^{k_{max}}$ as $\{a_1, \ldots, a_{k_{max}}\}$.

Key generation. Key generation is assumed to be done by a trusted party. The trusted party chooses random values a_1 and a_2, and compute $a_3 = (a_1 + a_2) \bmod q$ for some order q of a group G with generator g. He then computes $y_1 = g^{a_1}, y_2 = g^{a_2}$ and $y_3 = g^{a_3}$. For each voter V_i, he chooses a random s_i and a pseudo-random function instance d_i. He computes $\gamma_i = g^{s_i}$ and generates the receipt code list $\mathcal{RC}_i = \{(v_j, f(v_j)^{s_i})\}_{j=1}^{k_{max}}$ where f is an encoding function and k_{max} is the number of candidates. Before the elections, $\mathcal{RC}_i$ is sent to the voter over the pre-channel (i.e., postal service), $(y_1, a_2, y_3, \{(V_i, s_i)\})$ is given to BB, and $(y_1, y_2, a_3, \{(V_i, \gamma_i, d_i)\})$ is given to RG.

- **Voter and PC.** In order to cast a vote, V_i selects $(v_1, \ldots, v_k)$ from the candidate list $O = \{1, \ldots, k_{max}\}$. PC sets $v_{k+1} = v_{k+2} = \ldots = v_{k_{max}} = 0$. Then, PC encrypts the vote $f(v_j)$ as $(x_j, w_j) = (g_1^{t_j}, h_1^{t_j} f(v_j))$ $\forall j = 1, \ldots, k_{max}$ where $t_j \in_R \mathbb{Z}_p$. PC also proves the correctness of his computations by Σ-protocols $\Sigma_i = \{V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}; \{t_j\}_{j=1}^{k_{max}}, f(v_j)\}$ and signs as $\sigma_{V_i} = \mathsf{Sign}_{V_i}(V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i)$. PC sends $(V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i, \sigma_{V_i})$ to BB.
- **Ballot Box.** BB first verifies Σ_i and σ_{V_i} to check whether the computations are done correctly. BB then stores $counter_i^{++}, V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i$, and σ_{V_i}. For each V_i, it computes $(\breve{x}_i, \breve{w}_i) = (x_j^{s_i}, (w_j x_j^{-a_2})^{s_i})$ for $j = 1, \ldots, k_{max}$ and sends these values to RG.

– **Receipt Generator.** RG computes $\breve{r}_j = \breve{w}_j \breve{x}_j^{-c_2}$ $\forall j = 1, \ldots, k_{max}$. RG signs as $\sigma^i_{\mathsf{RG}} = \mathsf{Sign}_{\mathsf{RG}}(\mathsf{Hash}(V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}), \Sigma_i, \sigma_{V_i})$ and sends σ^i_{RG} back to PC. RG also generates SMS $= (\breve{r}_1, \ldots, \breve{r}_k)$ and sends it to V_i via SMS.

Once the SMS is received, V_i first verifies σ^i_{RG}. Next, V_i verifies $(v_1, \breve{r}_1), \ldots, (v_k, \breve{r}_k)$ which is received via SMS channel and $\{(v_j, f(v_j))^{s_i}\}$ which is received via postal service. If there is a mismatch, V_i observes that there is a problem with his vote. This means that either there exists a malware in the voter's computer or the data has been changed during the data transmission.

Audits and Vote counting protocol. The auditor must approve the input from BB before the decryption of the encrypted ballots. DS decrypts the encrypted ballots sent by BB by his private key a_1 and shuffles the result before output. DS also proves to the auditor that the output ballots are indeed the encrypted ballots.

3.2 Potential weaknesses of the protocol

A practical issue with the threshold. The protocol described in Section 3.1 seems to be a (2,3)-threshold cryptosystem among the players BB, RG and DS [8]. However, since DS can also decrypt by his private key a_1 this system is not really a (2,3)-threshold scheme. If BB and RG are compromised and collaborate then the private key of DS can be obtained, and therefore, vote privacy can be easily compromised. This part of the system might be weak since the threshold is very small.

One may come up with a questioning that there might be a cooperation between BB and RG, and argue that the anonymity of the election could be easily violated. Since it is not easy to refute those claims, the trust of the society may decrease dramatically. The reason is that there are only two players (BB and RG) in the system. Note that these players are required to be physically and organizationally separate (if they are not, they are not really two distinct players). Since in practice, the number of distinct and independent organizations available are very small, it is also not desirable to introduce new players.

In Section 4, we propose our improved protocol which is slightly modified version of the Norwegian protocol. This protocol prevents vote privacy even if there is a cooperation of BB and RG without new additional players.

A practical issue with holding both the SMS and the receipt codes. Once the election is over, there should be no way to reveal the voter's intention. However, having both the receipt codes and SMS gives a potential threat to vote privacy. That means, many people in fact carries their votes in their pocket. This part of the protocol might be very sensitive since any voter having mobile phones and the receipt paper can show his vote to anyone including his friends, colleagues and family members. Note that this issue is not the case with the conventional paper-based elections.

Indeed, one may argue that this problem can be avoided by re-voting. However, in general, re-voting is used either in the case of malicious computers or

coercions. Besides that, there are not expected to exist many coercions and malicious computers, and therefore, not many people are going to vote more than once. This might be a serious problem and the system might open the door to a new threat. Hence, this adversarial behavior might be realistic in many commercial, political, and social settings for especially eastern countries. Therefore, it is better to construct a mechanism where the votes can be verified by only a limited number of times (e.g., only once) by the voters.

In order to solve this problem, we propose changing the structure of SMS as follows: once the voter casts his vote, an SMS will be sent to his mobile phone except that it will not contain the receipt code. In our proposal, SMS will only include a notification message for each casted vote. This notification is indeed necessary in order to prevent a malicious computer casting a vote without the knowledge of the voter. In order to verify a vote, we propose to use a phone call as an additional channel. More precisely, once the voter casts his vote for a candidate, he directly calls a number for verification. The voter asks any 4 positions of the receipt codes (e.g., the 1st, 5th, 8th and 9th positions) from the operator. These 4 positions are randomly chosen to simplify the verification for the ordinary voters in order to avoid asking the complete codes. The operator tells the characters (e.g., alphanumeric) in those 4 positions. The voter verifies his vote correctly in case the codes match, otherwise he observes that the vote has not not been successfully sent to BB. In the latter case, after taking necessary precaution the voter should re-vote. At the end of the voting time, the verification process is also closed. As we said before, the voter should only a limited number of times for every casted ballot (e.g., only once). This of course, is not an absolute solution, however, in many practical cases it does go a long way towards reducing the severity of the intentional and unintentional privacy violation.

A problem of an insecure pre-channel. The problem could exist when the printed receipt codes are either falsely generated (or falsely printed) or falsely sent to the voters via a malicious pre-channel [9]. In this case, the verification will fail immediately in the case of honest voter's computer. This situation can dramatically decrease the reliability of system in the society.

On the other hand, if the wrong receipt code is received by the voter and the computer is malicious, the entire voting system can work subtly without any detection. To illustrate this issue we give the following scenario. For the sake of simplicity, we assume that there are only two candidates (v_1, v_2) (e.g., Yes/No referendum) and the following receipt codes $\mathcal{RC}_i = \{(v_1, f(v_2)^{s_i}), (v_2, f(v_1)^{s_i})\}$ is sent to voter V_i (which is changed during either printing the codes or the postal service). Assume that the voter chooses the candidate v_1. When the voter casts his vote, a malicious PC encrypts $f(v_2)$ instead of $f(v_1)$. PC sends the encrypted vote $(x_j, w_j) = (g^{t_j}, y_1^{t_j} f(v_2))$ together with its Σ-proofs to BB. BB checks whether the computations are done correctly and if so, computes $(\breve{x}_i, \breve{w}_i) = (x_j^{s_i}, w_j^{s_i a_2})$ sends it to the RG. Finally, RG computes the receipt $\breve{r}_j = \breve{w}_j \breve{x}_j^{-a_3} \rightarrow f(v_2)^{s_i}\} \; \forall j = 1, \ldots, k_{max}$ and sends them to V_i via SMS. The voter checks whether the receipt code sent via SMS is the expected one. It can be

easily seen that, the voter has chosen the candidate v_1 and PC encrypted v_2 but the voter could not detect any problem since the receipts are equal. Hence, DS decrypts it to v_2 instead of v_1 even though the voter had intention for candidate v_1, BB and RG could not also detect any problem. We give two different possible solutions to avoid this concern.

- The first solution could be that the trusted party (electoral board) generates the printed receipt codes together with non-interactive Σ-proofs during the key generation. Once the codes $(x_j, y_j) = (v_j, f(v_j)^{s_i})$ are generated, the proofs assure that the discrete logarithm to the base $f(v_j)$ of y_j is known to the voter. In this way, it is impossible to swap the codes without detection. Note that this verification need not to be done for each voter, instead, a sufficient number of voters will already imply that the codes are generated and sent to the voters correctly. This verification can be checked by independent organizations, universities or experts of political parties who have sufficient ability to check the correctness of the proofs. The requirement of the pre-channel is that the receipt-codes should not be known by the voter's computer. For example, the voter can obtain and easily check the correctness of the Σ-proofs from a different computer (or a smartphone) than the one used for vote casting so that the verification can be computed by an ordinary voter [9].
- Another possible solution could be suggested in such a way that the encryptions (together with Σ-proofs) can be prepared by the voter in other (simple) machine dedicated to this process that has no connection with his PC [9]. The voter puts this value into the PC. In this way, it is impossible for the malicious computer to learn and change the vote without any detection. PC sends the encryptions to BB which will forward to RG after the necessary calculations. RG will finally return the receipt code via SMS. The voter then checks whether this value is the same as the one received via post-channel. We highlight that the software on this machine should be open source so that anyone can check the correctness of the code which will be really simple, since it only does ElGamal encryption with non-interactive Σ-proofs. However, this solution can be difficult to perform for an ordinary voter. Still, it could be more convenient and easier than checking the correctness of the entire software of the voting system.

4 The enhanced protocol

In this section, we propose our protocol, which is a slight modified version of the Norwegian protocol. This protocol prevents a possible cooperation between BB and RG without introducing new players. Similar to the Norwegian protocol, we have the same three players in our protocol. We highlight that the ballots are encrypted by the public key $y_1 = g^{a_1}$. The encryptions are decrypted by BB and RG in such a way that the voter verifies the correctness. The same encryptions are also decrypted by DS. Similarly, we follow the the same form

of the underlying cryptosystem except that BB and RG share a private and a public key pair. Unlike Norwegian protocol, we give a new public and private key pair to DS. Hence, in our scheme we completely separate the keys of BB and RG from DS.

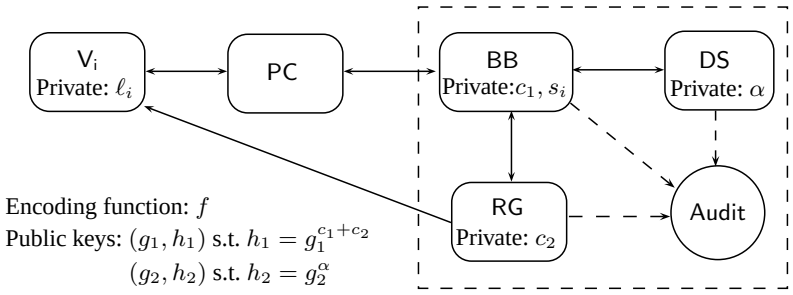

Fig. 1. Key distribution in our protocol

Our proposed scheme is divided into two main procedures: (i) vote casting/verification (ii) vote counting. Before the elections, the voters receive the printed *pre-receipt codes* (e.g., by postal service). In the first step of the vote casting, the voter prepares two encryptions: For the first encryption, he chooses a random value to mask his vote and then encrypts the masked vote with the public key of BB and RG. Next, he computes the second encryption of the same vote (but this time without mask) with the public key of DS. The voter proves with Σ-protocols that the votes are the same. BB and RG decrypt the first encryption and send the *post-receipt code* to the voter via SMS. The voter uses the mask value and the pre-receipt codes to verify whether the vote has been casted correctly. So that, even if BB and RG cooperate they can not learn any information about the votes since they are already masked. If the verification is passed by the voter, the second encryption which used the public key of DS are sent for decryption and counting.

Before the system setup, the key generation phase is executed by the Electoral Board as follows: BB and RG share private key pairs c_1 and c_2 respectively. The corresponding public key is $h_1 = g_1^{c_1+c_2}$. Unlike Norwegian Protocol [8], the decryption server DS will have different key pairs for a homomorphic cryptosystem. Let's say the private key of DS is $\alpha \in Z_p$, and its public key is $h_2 = g_2^{\alpha}$ (see Figure 1). The details of the protocol are as follows.

4.1 Vote casting protocol

Protocol steps:

1. **Vote casting**
   - Voter V_i chooses $(v_1, \ldots, v_k)$ from the candidate list $O = \{1, \ldots, k_{max}\}$.

– PC then computes $v_{k+1} = v_{k+2} = \ldots = v_{k_{max}} = 0$. For each candidate, PC chooses a masking value $\ell_j \in_R \mathbb{Z}_p$ and encrypts the vote as $(x_j, w_j) = (g_1^{u_j^1}, h_1^{u_j^1} f(v_j)^{\ell_j})$ for u_j^1, $\forall j = 1, \ldots, k_{max}$.

– PC also encrypts $(a_j, b_j) = (g_2^{u_j^2}, h_2^{u_j^2} f(v_j))$ for $u_j^2 \in_R \mathbb{Z}_p$.

– PC also proves the correctness by Σ-protocols $\Sigma_i = \{V_i, g_1, g_2, h_1, h_2, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}; \{u_j^1, u_j^2\}_{j=1}^{k_{max}}, f(v_i), \ell_j\}$ and signs as $\sigma_{V_i} = \mathsf{Sign}_{V_i}(V_i, \{(a_j, b_j), (x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i)$.

– PC sends $V_i, \{(a_j, b_j), (x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i, \sigma_{V_i}$ to BB.

2. **Verification and Signature by BB and RG**

– BB verifies Σ_i and σ_{V_i}.

– BB stores $counter_i^{++}, V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i, \sigma_{V_i}$.

– BB computes $(\breve{x}_i, \breve{w}_i) = \{x_j^{s_i}, (w_j x_j^{-c_1})^{s_i}\}_{j=1}^{k_{max}}$ and sends it to RG.

– RG computes the post-receipt codes as $\breve{r}_j = \breve{w}_j \breve{x}_j^{-c_2}$ $\forall j = 1, \ldots, k_{max}$. Note that unlike Norwegian protocol, pseudo-random function d_i is not required in our protocol. The reason is that BB and RG do not learn any information about voter's intention because of his masking technique.

– RG signs as $\sigma_{\mathsf{RG}}^i = \mathsf{Sign}_{\mathsf{RG}}(\mathsf{Hash}(V, \{(x_j, w_j)\}_{j=1}^{k_{max}}), \Sigma_i, \sigma_{V_i})$ and sends σ_{RG}^i back to PC.

– RG sends $\breve{r}_j$ $\forall j = 1, \ldots, k$ to the voter via SMS.

3. **Verification by the voter**

– V_i verifies σ_{RG}^i.

– V_i also verifies the post-receipt codes $(v_1, \breve{r}_1), \ldots, (v_k, \breve{r}_k)$ which is received via SMS and the pre-receipt codes $\{(v_j, f(v_j))^{s_i}\}$ which is received via postal service as follows: V_i computes $r'_j = f(v_j)^{s_i \ell_j}$ $\forall j = 1, \ldots, k$ and checks whether r'_j is equal to $\breve{r}_j$.

– If there is a mismatch, V_i observes that there is a problem with his vote. This means that either there is a malware in the voter's computer or the data has been changed during the data transmission.

Voter (V_i + PC)	**Server (BB + RG)**
Public key of BB and RG: h_1 $(= g_1^{c_1+c_2})$	**Private shares of BB:** c_1, s_i
Public key of DS: h_2 $(= g_2^{\alpha})$	**Private share of RG:** c_2
Codes received via pre-channel: $\{(v_j, f(v_j)^{s_i})\}_{j=1}^{k_{max}}$	**Private key of DS:** α
# of candidates: k_{max}	**Public key of RG:** γ_i $(= g_1^{s_i})$

1. Phase: Vote casting

V_i Select candidates $(v_1, \ldots, v_k)$
PC computes $v_{k+1} = v_{k+2} \ldots = v_{k_{max}} = 0$
PC computes:

$$\left(\begin{array}{c} u_j^1, u_j^2 \in_R \mathbb{Z}_p \\ \\ \ell_j \in_R \mathbb{Z}_p \\ \\ (x_j, w_j) = (g_1^{u_j^1}, h_1^{u_j^1} f(v_j)^{\ell_j}) \\ \\ (a_j, b_j) = (g_2^{u_j^2}, h_2^{u_j^2} f(v_j)) \end{array} \right)_{j=1}^{k_{max}}$$

PC computes: $\Sigma_i = \{V_i, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}; \{(u_j^1, u_j^2)\}_{j=1}^{k_{max}}, f(v_i), \ell_j\}$
PC signs: $\sigma_{V_i} = \mathsf{Sign}_{V_i}(V_i, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}, \Sigma_i)$

$$\xrightarrow{V_i, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}, \Sigma_i, \sigma_{V_i}}$$

2. Phase: Verification and Signature

BB verifies Σ_i and σ_{V_i}
BB stores: $counter_i^{++}, V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i, \sigma_{V_i}$
BB computes: $(\breve{x}_i, \breve{w}_i) = \{(x_j^{s_i}, (w_j x_j^{-c_1})^{s_i})\}_{j=1}^{k_{max}}$
BB sends $(\breve{x}_i, \breve{w}_i)$ to RG.
RG computes: $\breve{r}_j = (\breve{w}_j \breve{x}_j^{-c_2})\ \forall j = 1, \ldots, k_{max}$
RG signs:
$\sigma_{\mathsf{RG}}^i = \mathsf{Sign}_{\mathsf{RG}}(\mathsf{Hash}(V_i, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}), \Sigma_i, \sigma_{V_i})$

$$\xleftarrow{\sigma_{\mathsf{RG}}^i}$$
$$\xleftarrow{\mathsf{SMS}}$$

3. Phase: Verification

V_i verifies σ_{RG}^i
V_i computes $r'_j = (f(v_j)^{s_i})^{\ell_j}\ \forall j = 1, \ldots, k$
V_i verifies $(v_1, r'_1), \ldots, (v_k, r'_k) \in \{(v_j, f(v_j)^{s_i \ell_j})\}_{j=1}^{k_{max}}$

Protocol diagram of our vote casting protocol

Note that like the Norwegian protocol we also have used SMS in our protocol. We can of course adapt the phone call procedure to prevent the practical issue with the SMS described in Section 3.2. Namely, RG sends SMS in the second phase of the protocol which contain only the received-notification of the vote, saying *"Dear Alice, You have voted successfully on 15.06.2011 at 23:59:59. Please call 999 0 999 for the verification of your vote."*.

4.2 Vote counting protocol

Our vote decryption and counting protocol is exactly the same with the Norwegian protocol except that DS will use his own private key in order to decrypt (a_j, b_j). Namely, for each voter V_i, BB sorts all the recorded list $\{counter_i, \{(a_j, b_j)\}_{j=1}^{k_{max}}, \Sigma_i, \sigma_{V_i}\}$, finds the largest sequence number $counter_i$ and adds to a list $L = \{(a_j, b_j)\}$. BB finally sends L to DS which then mixes the encrypted vote list and proves the correctness to the auditors. Auditors involves in verifying in every step of the computations. Finally, votes are decrypted and counted.

5 Security analysis

In this section, we informally analyse the security of our protocol against active attacks. We first highlight that the differences between our protocol and the Norwegian protocol are adding a masking technique by the voter and separating distinct keys between BB, RG and DS. In the Norwegian protocol, the scheme use (2,3)-threshold cryptosystem (with the condition that $a_1 + a_2 = a_3 \mod q$) whereas in our protocol, BB and RG use a (2,2)-threshold cryptosystem which are only responsible for generating a pre-receipt code, verifying the encrypted votes and returning a post-receipt code to the voter. If everything goes correctly, the encrypted votes are sent to DS for the decryption and the counting. We note that in each step of the protocol the parties must prove with zero knowledge that they have executed the protocol correctly.

- **The Voter and PC.** With the same reasoning of the Norwegian protocol, digital signatures are used to prevent BB ballot from inserting forged ballots and a malicious voter claiming that a given ballot belongs to someone else. It also guarantees that at most one ballot is counted per voter. Besides that, PC proves that he knows the content of the ciphertexts (x_j, w_j) and (a_j, b_j) with the proof of knowledge [4]. Equality-composition of Σ-proofs also assure that the votes inside the encryptions (x_j, w_j) and (a_j, b_j) are equal to each other. Intuitively, these proofs are required in order to prevent a corrupt computer (or voter) submitting a valid vote v_j which is encrypted by the public key of BB and RG and a fake vote $v_j \notin \{v_1, \ldots, v_{k_{max}}\}$ (e.g., a random message) which is encrypted by the public key of DS. OR-proofs also prevent a corrupt voter submitting a fake candidate instead of encrypting a valid candidate.
  The voter uses the exponent ℓ_j to randomize his vote $f(v_j)$ and encrypts with the public key of BB and RG. In return, he receives $f(v_j)^{\ell_j s_i}$. The voter computes the exponent ℓ_j of the printed receipt codes to verify whether his vote was stored correctly. Even if there is a cooperation between BB and RG they can not send a different valid code and learn the vote without bypassing the voter unless they solve the discrete logarithm problem.
- **The Ballot Box and the Receipt Generator.** BB proves the correctness of its computations to RG. Namely, BB must send the signed ballot to RG, and also prove that $(\breve{x}_i, \breve{w}_i)$ is computed correctly. This proof prevents a

malicious BB cooperating with a malicious voter from misusing the receipt generator's decryption capability. In order to verify the computation of BB, RG must receive the entire ballot, including the voter's signature and the computer's proofs of knowledge. RG verifies the voter's signature and the proofs, and then computes a hash of the ballot which is then given to the voter as a second receipt to verify whether the vote has been received correctly. RG also returns the message to BB which forwards it to the voter's computer. Without this signature, the voter's computer will not inform the user that the ballot has been accepted.
If BB and RG are corrupt and cooperate, they can learn only the encrypted value (x_j, w_j). However, since this value has been masked by the voter's randomness ℓ_j the privacy of the voter will still be maintained. Besides that, the other encryption (a_j, b_j) does not give any information to BB and RG since it is encrypted by the public key DS. Besides, if the corrupt BB listens the communication between RG and the voter via SMS channel it does not get any information since the vote has been already masked. In the Norwegian scheme a pseudo-random function d_i is used to ensure this.

- **The Decryption Service.** DS is a reasonably standard system consisting of a mix net followed by verifiable decryption [3]. This part of the system is exactly the same with the Norwegian protocol except that it decrypts (a_j, b_j) using the key α.
- **The Auditor.** The auditor verifies the content of the ballot box (signatures and proofs), that no ballots have been inserted, removed or lost compared to the receipt generator list. The auditor compares this list to the ciphertexts input to the mix net, then verifies the proofs ordered by the mix net and the decryption service. The auditor finally publishes hash of every ballot to ensure voters that their ballots were included in the result of the election.

6 Conclusion

The Internet voting protocol which will be experienced in Norwegian elections is rather strong. However, apart from its strengths, the protocol still has some potential weaknesses regarding to the following assumptions: (i) BB and RG cannot be compromised and cooperate at the same time (ii) keeping both the printed receipt codes and SMS does not violate the vote privacy (iii) the receipt codes are printed and sent to the voters securely. Although the overall protocol is secure under these strong assumptions, we modified the protocol to improve its reliability and verifiability to make it more robust without these assumptions. We have shown that our enhanced protocol ensures vote privacy even if there is a simultaneous corruption and cooperation of BB and RG.

Moreover, in the Norwegian protocol, a voter can reveal his vote if he carries SMS and the receipt code in his pocket, even after the election has ended. This may cause a big threat that opens the door for selling vote or coercers, therefore we suggested to use SMS only as a notification message. In order to verify the vote, we use a phone call channel. We believe that this mechanism would provide

an interesting alternative approach to receiving SMSs that prohibits the voter to reveal his vote.

References

1. E-voting in trial municipalities starts in the autumn. 2011. http://www.regjeringen.no/en/dep/krd/prosjekter/e-vote-2011-project/news-about-the-e-vote-2011-project/year/2010/e-voting-in-trial-municipalities-starts-.html?id=616314.
2. Internet voting in estonia. 2011. http://www.vvk.ee/voting-methods-in-estonia/engindex.
3. D. Boneh and P. Golle. Almost entirely correct mixing with applications to voting. In *Proceedings of the 9th ACM conference on Computer and communications security*, CCS '02, pages 68–77. ACM, 2002.
4. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '94, pages 174–187. Springer-Verlag, 1994.
5. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc.
6. U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, STOC '90, pages 416–426. ACM, 1990.
7. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Proceedings on Advances in cryptology—CRYPTO '86*, pages 186–194. Springer-Verlag, 1987.
8. K. Gjøsteen. Analysis of an internet voting protocol. 2010. http://eprint.iacr.org/2010/380.
9. S. Heiberg, H. Lipmaa, and F. Van Laenen. On e-vote integrity in the case of malicious voter computers. In *Proceedings of the 15th European conference on Research in computer security*, ESORICS'10, pages 373–388. Springer-Verlag, 2010.
10. J. Svensson and R. Leenes. E-voting in europe: Divergent democratic practice. volume 8, pages 3–15. IOS Press, April 2003.