Norwegian Internet Voting Protocol Revisited: Ballot Box and Receipt Generator are Allowed to Collude

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

[1]Batman University, Faculty of Engineering and Architecture, Turkey
[2]TUBITAK BILGEM, Kocaeli, Turkey
[3]Sabanci University, Computer Science and Engineering, Istanbul, Turkey

ABSTRACT

Norway experienced internet voting in 2011 and 2013 for municipal and parliamentary elections respectively. Its security depends on the assumptions that the involving organizations are completely independent, reliable, and the receipt codes are securely sent to the voters. In this paper, we point out the following aspects:

- The vote privacy of Norwegian scheme is violated if Ballot Box and Receipt Generator cooperate since the private key of Decryption Service can be obtained by the two former players. We propose a solution to avoid this issue without adding new players.
- To assure the correctness, the receipt codes are sent to the voters over a pre-channel (postal service) and a post-channel (SMS). However, by holding both SMS and the postal receipt code, a voter can reveal his vote even after the elections. Albeit revoting is a fairly well solution for coercion or concealment, intentional vote revealing is still a problem. We suggest SMS only for notification of vote submission.
- In case the codes are falsely generated or the pre-channel is not secure, a vote can be counted for a different candidate without detection. We propose a solution in which voters verify the integrity of the postal receipt codes.

KEYWORDS
Internet voting; Vote privacy; Cryptographic protocols; Threshold cryptography; Homomorphic encryption

Received . . .

1. INTRODUCTION

Electronic voting has been already used by several countries (e.g. Norway, Estonia etc.). There can be many advantages over the traditional paper-based systems like security, reliability, convenience, mobility, tally speed, cost-effectiveness, and flexibility if the system is carefully designed. Up to now, cryptographers have been trying to improve the feasibility of voting systems by satisfying the required security properties such as vote privacy, correctness, verifiability and receipt-freeness.

Developing a reliable electronic voting system is known as one of the most difficult p roblem f or cryptographers since it involves several research fields such as psychology, sociology, laws, politics, information security and technology. It is also unique and intriguing research problem in cryptology because malicious behavior can come from both insider and outsider; for instance, voters may cheat the system, the system may deceive voters and coercers may influence on voters. Moreover, the voting system should be usable and understandable by the whole voting population, notwithstanding age or disability. Furthermore, voters are generally assumed to have neither computer literacy nor the computing power. For this reason, the e-voting systems should be simple and user-friendly. If the required security goals are satisfied, then it can be a great advancement over the classical paper-based voting systems.

Electronic voting can be achieved in two different methods. First one is *Electronic Voting Machine* (EVM) which refers to a combination of mechanical, electromechanical or electronic equipment that define ballots, record and count votes and display election results in a predetermined public area [39]. This machine can be a customized voting machine with touch screens or a stand-alone PC and placed in a voting booth. The second one is *electronic distance voting*, an improvement over EVM in that the

electronic registration, ballot casting, ballot counting are done in different physical place [39]. It paves the way for using a more common technology such as telephone, *Short Message Service* (SMS), interactive digital TV or the internet in order to cast a vote from any preferred distant location. The electronic voting with internet (i-voting) is the most appropriate and proper voting process for expatriates. The i-voting process is also easier and faster than other alternatives but also yields to open new threat.

Estonia is the first country that is currently using the internet for general elections since 2005 [3]. Internet voting is attractive to exclusively the voters who spend prominent amount of time in order to arrive the polling station. For instance, approximately half of the Estonian internet voters pointed out that it would take half an hour or more to get at the polling station. Most of the voters might not live in their registered place of residence. They can be either abroad or living in another place. It is encouraging to know that Estonia already successfully managed the internet election project.

The French Ministry of Foreign Affairs has successfully performed the internet voting for their expatriates residing abroad [5]. The French internet voting system is provided by Scytl [37]. During the 2012 French legislative elections, over 240,000 votes were cast electronically and this amount represents over 55% of the total overseas votes.

Norway is the second country which used the internet voting for the local governmental elections in September 2011, and targeting a comprehensive internet voting in 2017 for national election [2]. The cryptographic protocol used in Norway is implemented by Scytl [37] which is 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. Their internet voting protocol is strong from several aspects. The most important feature of this protocol is to resist against malicious voter's computers. Unlike the Estonian protocol, the Norwegian protocol prefers to use two additional channels: pre- and post-channel that are postal service and SMS respectively. The purpose of using different channels is to prevent malicious computer attacks, under the assumption that pre-channel and post-channel are unlikely to be controlled by the same attacker that controls the computer. Namely, an honest voter will be aware of a malicious behavior caused by the computer during the entire voting procedure. 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 taken into account during counting process. BB and a *Receipt Generator* (RG) together compute the receipt codes for the cast vote and sends it to the voter via SMS. The voter can verify his/her vote by checking SMS against the printed receipt codes. Once the election time is over, the cast ciphertexts are decrypted by a *Decryption Service* (DS) using re-encryption mix-net 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 [19, 22].

In [20], Norway internet voting protocol of 2011 was enhanced in terms of computational efficiency. In this protocol a new novel method has been proposed for computing the return codes efficiently. The Norway government ran the trials of this protocol during the 2013 parliamentary elections. After that, [21] proposed another instantiation of the cryptographic protocol fulfilling Gjøsteen's requirements for security and functionality. In this protocol, the security proof avoids the technical obstruction that was encountered by Gjøsteen. Additionally, this protocol has a better security proof, at the expense of a small increase in computational cost.

1.1. Related Work

Several electronic voting schemes have been proposed in the literature. As this paper focuses on internet based voting schemes, below we only mention some of the well-known related works on i-voting.

The Rijnland Internet Election System (RIES) is an i-voting system which is used in public elections (next to the traditional voting by postal mail) in the Netherlands in 2004 and 2006. RIES is end-to-end verifiable which enables the voter to verify whether his/her vote has been counted as cast. After the source code was published at June 2008, it has been observed that RIES is vulnerable to even very simple attacks. For more details and analysis of RIES we refer to [15].

In [31], the authors highlight the problem of the tallying complexity in Juels et al.'s [26] first coercion resistant i-voting protocol. They also examine and determine the key concept behind the existing protocol which performs the tallying process in linear time. Then, they propose a new i-voting protocol which allows efficient vote authorization with less computation power on the voter's side. The protocol is based on adjustable anonymity sets. The size of the anonymity sets determines the degree of coercion resistance. They show that their protocol requires linear time complexity and is comparable with the existing ones.

The Estonian i-voting system and the analysis of the election results are presented in [38]. In 2011 Parliamentary elections, over 140.000 voters used i-voting which was about the 24,3% of all votes. Although the quarters of the voters used Estonian i-voting system it is not sufficient to fulfill the expected security requirements, i.e., it does not prevent attacks from voter's computers. An attacker can inject malicious code into voters' computers to listen to the data and interpret it without being aware of the voter. That's why, the authors in this paper conclude

that the level of trust in the voter's computer should be significantly reduced.

The Helios is the first web-based, open-audit voting system to provide solutions for low coercion elections [6, 7]. A user can set-up an election, invite voters to cast a private ballot using a modern web browser with the Helios. The user can also calculate a tally, and generate a validity proof for the whole process [24]. It has been used in real-life election by several institutions to elect e.g., (i) the board of the International Association for Cryptographic Research (IACR) of 2015 [25], (ii) the undergraduate student government by Princeton University [23], and (iii) the president of the Université Catholique de Louvain [7]. Helios 2.0 is based on the homomorphic tallying of the encrypted votes, and offers another approach Mix networks for the tallying of an election [35].Due to an attack by Estehghari and Desmedt [16] Helios 2.0 was upgraded to Helios 3.0. In [10], the authors propose a new generalized computational security model for ballot privacy and then analyze the security of Helios in this model. As Estonian Internet Voting system, Helios also does not protect against potential malware on the voter's computers.

In [8], the authors introduce a revised version of the e-valg 2011 project that is an i-voting pilot project for the municipal and regional elections of 2011 in Norway. In the remote voting system, the client software is responsible of encoding the voting choices selected by the voter. In order to avoid the need to trust the client software, cast as intended verification functionality is used for auditing this process. The paper also analyzes the security of the proposed functionality.

In [32], the authors debate the validation of transparency and the recommended measures meant to establish profound trust via analyzing Norwegian internet voting project. They also aim to reveal to which degree the voting project adheres to those recommendations. They show that considerable additional costs and complexity are required to achieve transparency and the technical measures. They also propose that E-valg project does additional effort in order to establish trust despite the majority of public trust is supposed to toward central election administration. The authors argue that the trust in different entities and the limitations of the verifiability are narrowly get into touch to public.

Kusters et al. [27] introduce a simple attack, which they call a *clash attack*, on the verification procedure of e-voting systems. The purpose of this attack is to force the voting machines to utilize a particular randomness r for the same candidate during the process of the semantically secure encryption, and therefore, the same voting receipt would be generated and distributed to different voters. Note that using the same randomness would lead to a deterministic encryption scheme rather than a probabilistic scheme because final encrypted votes (for the same candidate where the same randomness is used) would be identical. Hence, the malicious voting authority will keep only one receipt on the bulletin board, and in this way, can remove the remaining encryption without being detected. Namely, the tallying authorities can replace ballots by some new selected ballots, and by doing this, they can manipulate the election result without being detected. The authors show that this simple attack can be applied to several e-voting mechanisms such as [4, 6, 36]. Furthermore, in [27], Kusters et al. also propose countermeasures against this attack. Similarly, [34] applies this attack to voting mechanism of [33] and propose an enhanced voting scheme based on utilizing ring signature in the ballot distribution process where voters are exchanging required information with the authorities.

As described above, most of the i-voting systems have either lack of enough security or privacy issue of the voters. Also, we highlight that most of the proposed schemes (including the above-mentioned ones) do not protect against malicious voter's computers. Norwegian i-voting system is the only system that prevents the possible security attacks from malicious voter's computers.

1.2. Our contributions

In this paper, we focus on the internet based voting protocol which is tried out in Norway in September 2011 for municipal elections. Note that our contributions do not collude with other extensions of this protocol. 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 skeptic about their vote privacy since the trust level of these organizations is low. We propose our solutions to mitigate those issues.

The main contribution of this paper is as follows. The Norwegian internet voting protocol 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 vote privacy will be lost even if the key of DS is not revealed. The key of DS can be recovered by colluding BB and RG. 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 this collaboration.

Secondly, we remark that either if the receipt codes are falsely generated (or falsely printed) or the pre-channel is not secure, then a vote for candidate c_i can be counted for candidate c_j where $c_i \neq c_j$ without being detected. In order to avoid this concern, we propose that the trusted party also prepares and prints non-interactive zero knowledge proofs of the ballots on the receipt code form. More concretely, with the help of smart device using 2D Barcode (eg., QR code), the voter can verify the receipt

codes with these proofs whether each code is assigned to a candidate correctly. This provides voters to ensure the integrity 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 voter's PC is malicious.

Finally, another minor contribution of this paper is that having both the printed receipt codes and SMS might be a potential threat for vote privacy. A voter may disclose 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 automated phone call for the complete verification of the votes. This might prevent (intentional or unintentional) violation of vote privacy.

1.3. 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. In Section 4, we propose our new protocol which avoids the possible cooperation between BB and RG. We also provide our quick solutions to the weaknesses described in Section 3. 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 the message or the plaintext space, $\mathcal{C}$ the ciphertext space, and $\mathcal{R}$ the 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 multiplicative 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 $E(m;r) \times E(1;r')$ is a new encryption whose plaintext is again m (and its randomness is $r + r'$).

Elgamal cryptosystem is one of the most popular partial homomorphic cryptosystems. In Elgamal cryptosystem, let G be a cyclic group of order q with generator g. Then, $x \in \{1, \ldots, q-1\}$ is the secret key and $h = g^x \mod q$ is the public key. The encryption of message m is $Enc(m) = (c_1, c_2) = (g^r, mh^r))$ for some random $r \in \{0, \ldots, q-1\}$. The decryption process is done as follows. The secret owner simply calculates $s = c_1^x$ and computes $c_2 s^{-1}$ which recovers the message m. In the proposed protocols, ElGamal scheme is utilized as a multiplicative homomorphic cryptosystem [14]. 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 persuades the verifier not only of the correctness of a statement, but also that it possesses the witness w for the statement. See [12] for further details. The Fiat-Shamir technique [18] 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 [17].

As in the internet voting protocol that is used in Norway, we also use the fact that there are efficient Σ-protocols for the relation $R = \{(e; m, r) : e = E(m, r)\}$ for homomorphic ElGamal encryptions, proving knowledge of the message m and randomness r for a given encryption scheme $e = E(m, r)$. In our protocols, since the voter computes two distinct encryptions of the same vote, we also need to ensure equality-composition of Σ-protocols (EQ-ZK) for the relation $\mathsf{EQ\text{-}ZK} = \{(e_1, e_2; m, r_1, r_2, k) : e_1 = E(m, r_1) \bigwedge e_2 = E(m^k, r_2)\}$. We also note that OR-composition Σ-proofs (OR-ZK) 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-protocol for the relation is shown in Figure 1.

3. NORWEGIAN INTERNET VOTING PROTOCOL AND ITS POTENTIAL WEAKNESSES

In this section, we first briefly describe the internet voting protocol which was 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 [19].

3.1. Protocol description

There are three players (organizations) in this system which are assumed to be independent and do not collude maliciously (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 $O = \{1, \ldots, k_{max}\}$ where $v_i \in O$. PC sets $v_{k+1} = v_{k+2} = \ldots = v_{k_{max}} = 0$. Then, PC encrypts the vote $f(v_j)$ as $(x_j, w_j) = (g_1^{t_j}, y_1^{t_j} f(v_j)) \quad \forall j = 1, \ldots, k_{max}$ where $t_j \in_R \mathbb{Z}_p$. PC also proves the correctness of his computations by Σ-protocols OR-ZK, EQ-ZK using $\Sigma_i = \{V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}; \{t_j\}_{j=1}^{k_{max}}, f(v_j)\}$ 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 the OR-ZK, EQ-ZK proofs using Σ_i and the signature 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 $\breve{r}_j = \breve{w}_j \breve{x}_j^{-a_3}$ $\forall j = 1, \ldots, k_{max}$. RG signs as $\sigma_{\mathsf{RG}}^i = \mathsf{Sign}_{\mathsf{RG}}(\mathsf{Hash}(V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}), \Sigma_i, \sigma_{V_i})$ and sends σ_{RG}^i 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_{RG}^i. 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

3.2.1. A practical issue with the threshold.

The protocol described in Section 3.1 is a (2,3)-threshold cryptosystem among the players BB, RG and DS [19]. 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

$$\mathsf{OR}-\mathsf{ZK}=\left\{(e_j,e'_j,\{m_j\}_{j=1}^{k_{max}};r_j,r'_j,k_j):\bigvee_{j=1}^{k_{max}}\left(e_j=E(m_j,r_j)\bigwedge e'_j=E(m_j^{k_j},r'_j)\right)\right\}.$$

Figure 1. Σ-protocols for *OR* Zero-knowledge (OR-ZK) relation

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. Our protocol preserves vote privacy even if there is a cooperation of BB and RG without new additional players.

3.2.2. A practical issue with holding both 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 in social settings. 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. The solution to this issue is described in our enhanced protocol in Section 4.5.1.

3.2.3. 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 [22]. In this case, the verification will fail immediately in the case of honest voter's computer. This situation can dramatically decrease the reliability of the system.

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 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. The solutions to fix this issue are given in our enhanced protocol in Section 4.5.2.

4. THE ENHANCED PROTOCOL

4.1. Overview of Our Protocol

In this section, we propose our protocol, which is a slight modified version of the Norwegian protocol. This protocol preserves 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 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.

Before the system setup, the key generation phase is executed by the Electoral Board as follows: BB and

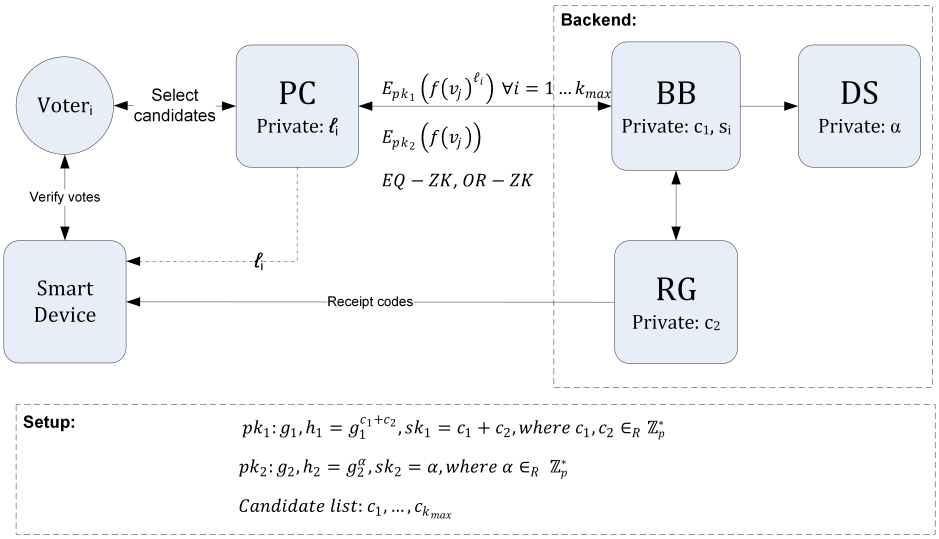

Figure 2. Key Distribution in our protocol. PC chooses random ℓ_i privately, encrypts $f(v_j)^{\ell_i}$ under the public key pk_1 and $f(v_j)$ under the public key pk_2 together with the EQ-ZK proof that the two encrypted messages are actually the same. Additionally, PC also proves with OR-ZK that the encrypted message is one of the candidate list $\{c_1, \cdots, c_{k_{max}}\}$.

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 [19], the decryption server DS will have different key pairs for a homomorphic cryptosystem. Let $\alpha \in Z_p$ denote the private key of DS. Then, the corresponding public key is $h_2 = g_2^{\alpha}$ (see Figure 2).

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 printed form, the pre-receipt codes would be also encoded as visual 2D Matrix such as Data Matrix [1]. This visual matrix will be used during the vote verification by the voter via smart phone. Considering the vote casting, the main difference between our protocol with the Norway voting protocol is that in our protocol PC blinds the vote against BB and RG. In the first step of the vote casting, the voter chooses the candidates $(v_1, v_2, \dots, v_{k_{max}})$ on PC. Then, PC prepares two set of encryptions. For the first set of encryptions, PC chooses a random value ℓ_i in order to blind (mask) his votes $(f(v_j)^{\ell_i} \forall j = 1 \dots k)$ and then encrypts the blinded votes with the public key of BB and RG. Even if BB and RG cooperate they cannot learn any information about the votes since they are already blinded. Next, PC computes the second encryption of the same votes (but this time without mask) with the public key of DS. The PC also proves with equality zero-knowledge (EQ-ZK) that the encrypted votes are the same. BB and RG decrypt the first encryption and send the *post-receipt code* to the voter via SMS the voter's smart phone. Note that, blinding votes changes the result of the receipts $(\breve{r}_j)$ and RG would generate $\breve{r}_j^{\ell_i}$ instead of $\breve{r}_j$ and sends $\breve{r}_j^{\ell_i}$. Now, the voter actually has the pre-receipt code as $v_j, f(v_j)^{s_i}$ and post-receipt code $(\breve{r}_j^{\ell_i})$. For the verification, the smart phone reads the ℓ_i from the PC's screen and the voter's selected receipt codes $(f(v_j))$ via 2D barcode scanner (e.g, QR code) . The application will compute $(f(v_j)^{s_i})^{\ell_i}$ and compare these results with the post-receipt coded received by SMS. If the verification is passed by the voter with the help of the smart phone, the second encryption which used the public key of DS are sent for decryption and counting.

4.2. Our Main Protocol

The details of the protocol are as follows. For the illustration of the protocol see Figure 3.

4.3. Vote casting protocol

1. **Vote casting**

   - Voter V_i chooses $(v_1, \dots, v_k)$ from the candidate list $O = \{1, \dots, k_{max}\}$.
   - PC then computes $v_{k+1} = v_{k+2} = \dots = v_{k_{max}} = 0$. For each candidate, PC chooses a masking value $\ell_j \in_R \mathbb{Z}_p$ and encrypts the vote as $(x_j, w_j) = (g_1^{u_{j1}}, h_1^{u_{j1}} f(v_j)^{\ell_j})$ for $u_{j1}, \forall j = 1, \dots, k_{max}$.
   - PC also encrypts $(a_j, b_j) = (g_2^{u_{j2}}, h_2^{u_{j2}} f(v_j))$ for $u_{j2} \in_R \mathbb{Z}_p$.
   - PC also proves the correctness by Σ-protocols $\Sigma_i = \{V_i, g_1, g_2, h_1, h_2, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}; \{u_{j1}, u_{j2}\}_{j=1}^{k_{max}}, f(v_i), \ell_j\}$ and signs as $\sigma_{V_i} = \mathsf{Sign}_{V_i}(V_i, \{(a_j, b_j), (x_j, w_j)\}_{j=1}^{k_{max}}, \Sigma_i)$.

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 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:** $\gamma_i\ (= g_1^{s_i})$

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:

$$\left(\begin{array}{c} u_{j1}, u_{j2} \in_R \mathbb{Z}_p \\ \ell_j \in_R \mathbb{Z}_p \\ (x_j, w_j) = (g_1^{u_{j1}}, h_1^{u_{j1}} f(v_j)^{\ell_j}) \\ (a_j, b_j) = (g_2^{u_{j2}}, h_2^{u_{j2}} f(v_j)) \end{array} \right)_{j=1}^{k_{max}}$$

PC computes OR-ZK, EQ-ZK using $\Sigma_i = \{V_i, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}; \{(u_{j1}, u_{j2})\}_{j=1}^{k_{max}}, f(v_i), \ell_j\}$
PC signs: $\sigma_{V_i} = \mathsf{Sign}_{V_i}(V_i, \{(x_j, w_j), (a_j, b_j)\}_{j=1}^{k_{max}}, \Sigma_i)$

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

2. Phase: Verification and Signature

BB verifies the OR-ZK, EQ-ZK proofs using Σ_i and the signature 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 $(v_1, r'_1), \ldots, (v_k, r'_k) \in \{(v_j, f(v_j)^{s_i \ell_j})\}_{j=1}^{k_{max}}$

Figure 3. Protocol diagram of our vote casting protocol.

- 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 the OR-ZK, EQ-ZK proofs using Σ_i and the signature σ_{V_i}.
   - BB stores $counter_i^{++}, V_i, \{(x_j, w_j)\}_{j=1}^{k_{max}}$, Σ_i, and σ_{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 $\sigma^i_{\mathsf{RG}} = \mathsf{Sign}_{\mathsf{RG}}(\mathsf{Hash}(V, \{(x_j, w_j)\}_{j=1}^{k_{max}}), \Sigma_i, \sigma_{V_i})$ and sends σ^i_{RG} 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^i_{RG}.
   - V_i also verifies the post-receipt codes $(v_1, \breve{r}_1), \ldots, (v_k, \breve{r}_k)$ which are 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.

Note that similar to the Norwegian protocol we have also 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 Voter, You have voted successfully on 15.09.2011 at 23:59:59. Please call 999 0 999 for the verification of your vote."*.

4.4. 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 involve in verifying in every step of the computations. Finally, votes are decrypted and counted.

4.5. Other Improved Extensions

4.5.1. SMS for only notification purpose.

In our protocol, 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 cast 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 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 cast ballot (e.g., only once). In fact, this may not be 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.

4.5.2. Solution to insecure pre-channel.

For the solution of the problem described in 3.2.3, we propose two different possible solutions to avoid this concern.

- The first possible solution is 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 the voter that the discrete logarithm to the base $f(v_j)$ of y_j is known. 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 smart phone) than the one used for vote casting so that the verification can be computed by an ordinary voter [22].
- Another possible solution could be suggested in such a way that the encryptions (together with Σ-proofs) can be prepared by the voter in a (simple) secure machine dedicated to this process that has no connection with his PC [22]. This proof should be prepared in such a way that the user is able to verify easily. The voter can put this value into a computing device by means of for example a barcode or RFID system. 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.

4.6. Computational Analysis

The original Norwegian Internet voting has been performed in practice. Regarding the security and privacy of voter, the computational complexity of the protocol is not much high for both client and server side. On the other hand, with our new instantiation the computational complexity increases, but also has increased security of the system. The PC computes extra the number of k_{max} modular exponentiation and EQ-ZK proofs. Smart device application also computes the number of k_{max} modular exponentiation for verification of vote via the receipt codes and the secret ℓ. Since the verification is not compulsory for every voter and the correctness of the scheme will be ensured with high probability if about % 5 of the voters perform the verification process. These computations are reasonable for a personal computer and a smart device. Therefore, we believe that the system will still be practical and quite effective with our extension.

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 uses (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. In the usual case the encrypted votes are sent to DS for decryption and counting processes. We note that in each step of the protocol the parties must prove with zero knowledge that they have executed the protocol correctly. Informally, vote privacy is achieved if (1) a ballot cannot be linked to the voter who cast it, and (2) no voter can prove that he/she voted in a particular way [9]. In [13], Delaune, Kremer and Ryan states that *"to ensure privacy we need to hide the link between the voter and the vote and not the voter or the vote itself"*. In this context, disclosing only the identity of the voter or the vote itself does not compromise vote privacy since the only requirement in terms of privacy is to break the link between the voter and the vote. In this respect, instead of user anonymity that defined in [30, 28, 29], we aim to achieve vote privacy against BB, RG and DS.

Theorem 5.1
Our protocol achieves vote privacy even if the Ballot Box and the Receipt Generator are corrupted simultaneously and maliciously collude.

Proof sketch.
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 of DS. Besides, if the corrupted 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. □

Theorem 5.2
Our protocol disallows a corrupted PC (or a malicious voter) to submit a valid vote v_j encrypted by the public key of BB and RG and a fake vote $v \notin \{v_1, \ldots, v_{k_{max}}\}$ encrypted by the public key of DS without being detected.

Proof sketch.
Following 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 the knowledge of the ciphertexts (x_j, w_j) and (a_j, b_j) with the proof of knowledge [12]. EQ-ZK proofs using Σ_i 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's PC 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 $f(v_j)^{\ell_j s_i}$. With the help of a smart device, the voter uses 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 cannot send a different valid code and learn the vote without bypassing the voter unless they solve the discrete logarithm problem. □

The Decryption Service. DS is a reasonably standard system consisting of a mix-net followed by verifiable decryption [11]. This part of the system is exactly the same as the Norwegian protocol except that it decrypts (a_j, b_j) using a key α.

The Auditors. Our scheme (as well as the Norwegian scheme) provides cast-as-intended property where the voters are able to verify that their ballot was cast as intended to the BB. Furthermore, we would like to clarify that the security of DS is also equivalent to the Norwegian voting system. The goal of the Norwegian protocol (and our protocol as well) is to transmit the encrypted votes to the DS securely. The counting process utilizing DS is in fact performed via a ceremony. Namely, the decryption key of DS is distributed to independent authorities in a threshold fashion by utilizing mix-nets (via semantically secure threshold encryption scheme). The link between the votes and the voters are broken via mix-nets (i.e., mix-net roughly works as follows: each independent authority first randomly mixes the votes, randomizes the encryptions, and then proves in zero-knowledge that the new fresh encryptions are equivalent to the given encrypted votes). Moreover, auditors are participated into vote tallying process in order to verify and guarantee the fair of counting. The auditors check the validity of the content of the ballot box (i.e., the signatures and the proofs) that no ballot has been inserted, removed or lost according to the receipt generator list. The auditors compare this list to the ciphertexts input to the mix-net and verify the proofs. Because of auditors and the keys are distributed to independent parties, the counting process is completely secured if at least one of them is honest.

Note that all of the existing i-voting systems are susceptible to Denial of Service (DoS) attacks. In case of DoS attacks voters would have to use the traditional paper voting in the polling station. Also, mitigating to a man in the middle is not possible for the Norwegian i-voting system since RG sends the voter a message over post-channel verifying his/her vote. This message would be difficult to spoof because of the sufficiently long random codes on the paper which is sent over the pre-channel.

6. CONCLUSION

The Norwegian internet voting protocol which was experienced in 2011 and 2013 is rather strong compared to other similar kind of known protocols. However, apart from its strengths, the protocol still has some potential weaknesses regarding its 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. More concretely, we have shown that our enhanced protocol ensures vote privacy even if there is a simultaneous corruption and cooperation between BB and RG.

Moreover, considering the Norwegian protocol, a voter can reveal his vote if he carries both SMS and the receipt code in his pocket, even elections has ended. This may cause a quite serious threat that opens the door for selling vote or intentional revealing. Therefore, we suggested to use SMS only as a notification message. In order to verify the vote, we use a automated phone call channel. We believe that this mechanism would provide an interesting alternative approach to receiving SMS that prohibits the voter to reveal his/her vote.

REFERENCES

1. ISO/IEC 16022:2006—Data Matrix bar code symbology specification, 2006.
2. Final results from the e-voting in the trial municipalities in Norway, September 2011.
3. Internet voting in Estonia. June 2011. Website: `http://www.vvk.ee/voting-methods-in-estonia/engindex`.
4. Wombat Voting system, 2011. Website: `http://www.wombat-voting.com/`.
5. FRENCH MINISTRY OF FOREIGN AFFAIRS: French expats vote online in 2012 legislative elections, 2012.
6. Ben Adida. Helios: web-based open-audit voting. In *Proceedings of the 17th conference on Security symposium*, pages 335–348. USENIX Association, 2008.
7. Ben Adida, Olivier De Marneffe, Olivier Pereira, and Jean-Jacques Quisquater. Electing a university president using open-audit voting: analysis of real-world use of helios. In *Proceedings of the 2009 conference on Electronic voting technology/workshop on trustworthy elections*, EVT/WOTE'09, pages 1–15. USENIX Association, 2009.
8. Jordi Puiggali? Allepuz and Sandra Guasch Castelli. Internet Voting System with Cast as Intended

Verification. In *3rd International Conference on E-Voting and Identity, Tallin, Estonia*, VoteID'11. Springer-Verlag, 2012.

9. Emmanue Benoist, Bernhard Anrig, and David-Olivier Jaquet-Chiffelle. *Internet-Voting: Opportunity or Threat for Democracy?*, pages 29–37. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
10. David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. Adapting helios for provable ballot privacy. In *Computer Security – ESORICS 2011*, volume 6879 of *Lecture Notes in Computer Science*, pages 335–354. Springer Berlin Heidelberg, 2011.
11. Dan Boneh and Philippe 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.
12. Ronald Cramer, Ivan Damgård, and Berry 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.
13. Stéphanie Delaune, Steve Kremer, and Mark Ryan. Verifying Privacy-type Properties of Electronic Voting Protocols. *Journal of Computer Security*, 17(4):435–487, December 2009.
14. Taher 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.
15. Berry Schoenmakers Engelbert Hubbers, Bart Jacobs and Henk van Tilborg. Description and Analysis of the RIES Internet Voting System. EIPSI, 2008. Website: `http://www.win.tue.nl/eipsi/images/RIES_descr_anal_v1.0_June_24.pdf`.
16. Saghar Estehghari and Yvo Desmedt. Exploiting the client vulnerabilities in internet e-voting systems: hacking helios 2.0 as an example. In *Proceedings of the 2010 international conference on Electronic voting technology/workshop on trustworthy elections*, EVT/WOTE'10, pages 1–9. USENIX Association, 2010.
17. 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.
18. Amos Fiat and Adi 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.
19. K. Gjøsteen. Analysis of an internet voting protocol. 2010. Website: `http://eprint.iacr.org/2010/380`.
20. Kristian Gjøsteen. The norwegian internet voting protocol. *IACR Cryptology ePrint Archive*, 2013:473, 2013.
21. Kristian Gjøsteen and Anders Smedstuen Lund. The norwegian internet voting protocol: A new instantiation. *IACR Cryptology ePrint Archive*, 2015:503, 2015.
22. Sven Heiberg, Helger Lipmaa, and Filip 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.
23. Helios Headquarters, Princeton University Undergraduate Student Government, 2016. Website: `https://princeton.heliosvoting.org/`.
24. Helios Voting, 2016. Website: `http://heliosvoting.org`.
25. International association for cryptologic research, 2015. Election page at: `http://www.iacr.org/elections/2015`.
26. Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, WPES '05, pages 61–70, New York, NY, USA, 2005. ACM.
27. R. Kusters, T. Truderung, and A. Vogt. Clash attacks on the verifiability of e-voting systems. In *2012 IEEE Symposium on Security and Privacy*, pages 395–409, May 2012.
28. Yanrong Lu, Lixiang Li, Haipeng Peng, and Yixian Yang. An anonymous two-factor authenticated key agreement scheme for session initiation protocol using elliptic curve cryptography. *Multimedia Tools and Applications*, pages 1–15, 2015.
29. Yanrong Lu, Lixiang Li, Haipeng Peng, and Yixian Yang. An energy efficient mutual authentication and key agreement scheme preserving anonymity for wireless sensor networks. *Sensors*, 16(6):837, 2016.
30. Yanrong Lu, Lixiang Li, Haipeng Peng, and Yixian Yang. Robust anonymous two-factor authenticated key exchange scheme for mobile client-server environment. *Security and Communication Networks*, 9(11):1331–1339, 2016.
31. Reto Koenig Michael Schläpfer, Rolf Haenni and Oliver Spycher. Efficient Vote Authorization in Coercion-Resistant Internet Voting. In *3rd International Conference on E-Voting and Identity, Tallin, Estonia*, VoteID'11. Springer-Verlag, 2012.
32. Melanie Volkamer Oliver Spycher and Reto Koenig. Transparency and Technical Measures to Establish Trust in Norwegian Internet Voting Authors. In *3rd International Conference on E-Voting and Identity,*

Tallin, Estonia, VoteID'11. Springer-Verlag, 2012.
33. H. Pan, E. Hou, and N. Ansari. E-note: An e-voting system that ensures voter confidentiality and voting accuracy. In *2012 IEEE International Conference on Communications (ICC)*, pages 825–829, June 2012.
34. H. Pan, E. Hou, and N. Ansari. Re-note: An e-voting scheme based on ring signature and clash attack protection. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 867–871, Dec 2013.
35. Damien Giry Philippe Bulens and Olivier Pereira. Running mixnet-based elections with helios. In *Proceedings of the 2011 conference on Electronic voting technology/workshop on trustworthy elections*, EVT/WOTE'11. USENIX Association, 2011.
36. Ronald L. Rivest and Warren D. Smith. Three voting protocols: Threeballot, vav, and twin. In *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, EVT'07, pages 16–16, Berkeley, CA, USA, 2007. USENIX Association.
37. Scytl. A Secure Electronic Voting Company. 2016. `http://www.scytl.com/`.
38. Peeter Laud Sven Heiberg and Jan Willemson. On applying i-voting for Estonian Parliamentary elections in 2011. In *3rd International Conference on E-Voting and Identity, Tallin, Estonia*, VoteID'11. Springer-Verlag, 2012.
39. Jörgen Svensson and Ronald Leenes. E-voting in Europe: Divergent democratic practice. volume 8, pages 3–15. IOS Press, April 2003.