Practical Remote End-to-End Voting Scheme

Xun Yi[1] and Eiji Okamoto[2]

[1] School of Engineering and Science
Victoria University, Melbourne, VIC 8001, Australia
[2] Department of Risk Engineering
University of Tsukuba, Tsukuba, Ibaraki, 305-8573, Japan

Abstract. Recently, remote voting systems have gained popularity and have been used for government elections and referendums in the United Kingdom, Estonia and Switzerland as well as municipal elections in Canada and party primary elections in the United States and France. Current remote voting schemes assume either the voter's personal computer is trusted or the voter is not physically coerced. In this paper, we present a remote end-to-end voting scheme, in which the voter's choice remains secret even if the voter's personal computer is infected by malware or the voter is physically controlled by the adversary. In particular, our scheme can achieve absolute verifiability even if all election authorities are corrupt. Based on homomorphic encryption, the overhead for tallying in our scheme is linear in the number of candidates. Thus, our scheme is practical for elections at a large scale, such as general elections.

Keywords: Electronic voting, coercion-resistance.

1 Introduction

Essentially, an end-to-end voting system can be envisioned as a decryption network composed of a collection of election authorities. The network takes as input a collection of encrypted ballots (posted publicly by voters) in one end and outputs in another end the tally of votes (posted publicly by the authorities) with a mathematical proof that the encrypted ballots were decrypted properly and that the votes were unmodified. Informally, an end-to-end voting system achieves *integrity* if any voter can verify that her ballot is included unmodified in a collection of ballots, and the public can verify that the collection of ballots produces the correct final tally, and the system keeps *privacy* if no voter can demonstrate how he or she voted to any third party.

So far, there have been two main categories of end-to-end voting schemes - polling station voting schemes and remote voting schemes.

Polling station voting schemes, such as [1,11,17,20,22], build their security on an untappable channel implemented as a private voting booth at a polling place, where a voter can cast her ballot in private. Thus, risk of voter coercion and vote buying can be greatly mitigated. These schemes require a voter to vote in person at a polling station on election days. This may not be convenient for those voters

K.N. Andersen et al. (Eds.): EGOVIS 2011, LNCS 6866, pp. 386–400, 2011.

who have no access to any polling station on election days, e.g., overseas citizens and military voters.

Remote voting schemes, such as [4,7,10,15], allow people to cast their votes over the Internet, most likely through a Web browser, from home, or possibly any other location where they have Internet access. While voting of this kind is hoped to encourage higher voter turnout and makes accurate accounting for votes easier, it also carries the potential of making abuse easier to perform, and easier to perform at a large scale [15]. One challenge to remote voting is how to prevent voter coercion and vote buying because the behavior of a voter casting a ballot remotely can be physically controlled by an adversary. Another challenge is how to ensure the remote personal computer by which a voter casts her vote is trusted because malware can endanger integrity of the elections as well as privacy of the voter [16].

The first voting scheme was introduced by Chaum [4], based on a mix network, where a collection of tally authorities take as input a collection of encrypted votes and output a collection of plain votes according to a secret permutation. This scheme allows each voter to make sure her vote was counted, while preserving the privacy of the vote as long as at least one tally authority is honest. In order to improve efficiency in tallying, Cohen (Benaloh) and Fischer [8] proposed a voting scheme, based on a homomorphic encryption E, where $E(x)E(y) = E(x+y)$ for any x and y in its domain. The basic idea is for each voter to encrypt her vote using a public-key homomorphic encryption function. The encrypted votes are then summed using homomorphic property without decrypting them. Finally, a collection of tallying authorities cooperate to decrypt the final tally. This scheme also preserves the privacy of votes as long as at least one tally authority is honest. In order to provide with unconditional privacy of votes, Fujioka et al. [10] proposed a voting scheme, based on blind signature, where a signer can digitally sign a document without knowing what was signed. The basic idea is that the voter has her ballot blindly signed by the voting authority and later publishes the ballot using an anonymous channel. Current voting schemes are based on either mix network, or homomorphic encryption, or blind signature.

The notion of receipt-freeness was first introduced by Benaloh and Tuinstra [2] to model the security of a voting scheme against voter coercion and vote buying. A voting scheme is receipt-freeness if a voter cannot prove to an attacker that he or she voted in a particular manner, even if the voter wishes to do so. Receipt-freeness voting schemes, such as [2,13,21], assume the existence of a private voting booth to isolate the voter from the coercer at the moment she casts her vote. Remote voting schemes are required to be coercion-resistant where the voter can be physically controlled by the adversary during voting. A rigorous definition for coercion-resistance was given by Juels et al. [15]. This model considers a powerful adversary who can demand of coerced voters that they vote in a particular manner, abstain from voting, or even disclose their secret keys. A voting scheme is coercion-resistant if it is infeasible for the adversary to determine if a coerced voter compiles with the demands. Intuitively, coercion-resistance implies receipt-freeness which itself implies privacy.

A coercion-resistant remote voting scheme was demonstrated by Juels et al. [15]. The basic idea is that each voter casts her ballot together with a secret credential, both encrypted by the public keys of the tally authorities. After a collection of encrypted ballots are mixed with a mix network such as [12,14,18], the validity of ballots (i.e., the validity of credentials) are checked blindly against the voter roll and only valid ballots are decrypted and counted. This scheme does not require an untappable channel for a voter to cast her ballot, but instead assumes an untappable channel for a voter to obtain a secret credential from the registrars during registration (potentially using post mail).

Current coercion-resistant remote voting schemes, such as Juels et al.'s scheme [15] and its variants [7], require public key encryptions on the side of the voter. Thus, they require the voter to trust the personal computer actually casting the ballot on her behalf. Considering that the voter's personal computer can be infected by malware that may reveal the voter's preferences or even change the encrypted ballot cast by the voter, Kutylowski and Zagorski [16] recently proposed a remote voting scheme, a combination of paper-based voting schemes Punchscan [5] and ThreeBallot [20]. The basic idea is that a voter makes a complete ballot by laying a ballot and a coding card side by side. Each voter is issued exactly one ballot by the election authority and she can get a coding card from any Proxy. This scheme preserves privacy of votes if both authorities do not collude. Even if the voter's personal computer is infected by viruses, her choice remains secret. This scheme does not allow a voter to prove how he or she voted unless vote-casting is physically supervised by an adversary.

Current remote voting schemes assume either the voter's personal computer is trusted to cast a vote or the voter is not physically controlled by the adversary. In this paper, we propose a remote voting scheme, in which the voter's choice remains secret even if the voter's personal computer is infected by malware or the voter is physically controlled by the adversary.

Our work is motivated by the most efficient voting scheme [13] based on homomorphic encryption. The main difference between them and our solution is that they assume the availability of an untappable channel between the voter and the authorities during voting while we require the untappable channel during voter registration only.

Consider an election where the candidates are $\{C_1, C_2, \cdots, C_{n_C}\}$ and the choice for each candidate is either "Yes" (denoted by 1) or "No" (denoted by -1), our basic idea can be described as follows. First of all, a voter $\mathcal{V}_i$ generates a public/private key pair for digital signature scheme on her own device. During registration, $\mathcal{V}_i$ presents herself to a registrar's office, where she is allowed privately to input n_C references $r_{i,j}$ ($\in \{1, -1\}$) on a trusted entry device (like setting PIN number in a bank branch), which, in turn, encrypts each $g^{r_{i,j}}$ with the public keys of tally authorities according to ElGamal encryption scheme [9] (where g is a generator of a cyclic group $\mathbb{G}$) and then posts on a public bulletin board the ciphertexts $\mathsf{E}(g^{r_{i,j}}) = (A_{i,j}, B_{i,j})$ (each corresponds to one candidate C_j) along with the voter's public key. During voting, $\mathcal{V}_i$ posts on the public bulletin board her ballot composed of $\beta_{i,j} \in \{1, -1\}$ ($j = 1, 2, \cdots, n_C$) and her signature on it, where

$\beta_{i,j} = 1$ if the choice of $\mathcal{V}_i$ is the same as her reference $r_{i,j}$ and $\beta_{i,j} = -1$ otherwise. During tallying, the tallying authorities sum $({A_{i,j}}^{\beta_{i,j}}, {B_{i,j}}^{\beta_{i,j}})$ $(= \mathsf{E}(g^{r_{i,j}\beta_{i,j}}))$ for each candidate C_j and then cooperate to decrypt the final tally. In case that the voter $\mathcal{V}_i$ is physically controlled by an adversary, the adversary can force the voter to choose $\beta_{i,j}$, but is uncertain of $g^{r_{i,j}\beta_{i,j}}$ because $g^{r_{i,j}}$ is encrypted, and therefore the voter's choice remains secret.

Compared to most of existing remote voting schemes, our scheme has three merits as follows: (1) No encryption is needed during voting and the ballot cast by a voter is "plain", thus any voter can verify that her ballot is included unmodified; (2) No mix network is needed during tallying and the tallying overhead is linear in the number of candidates, therefore it is practical for elections at a large scale; (3) Verifiability remains even if all election authorities are corrupt.

In addition, our scheme allows a voter repeatedly to refresh her references remotely after she registers and to use refresh references for a new election. Privacy is built on voter registration protected by a untappable channel.

The rest of our paper is organized as follows. Section 2 describes a basic voting scheme at first and then extends it to a general voting scheme. A rigorous proof of coercion-resistance and verifiability for our scheme is provided in Section 3. Conclusions are drawn in last section.

2 Remote End-to-End Voting

2.1 Participating Parties

Assume that there exists a publicly readable, insert-only bulletin board ($\mathcal{BB}$) on which public information (e.g., public keys, ballots and final tally) is posted. No one can overwrite or erase existing data on $\mathcal{BB}$. The public (including voters) can read the contents of $\mathcal{BB}$ anytime.

Normally, our remote voting scheme involves three types of participants as follows:

- Registrar ($\mathcal{R}$) authorizes voters for an election by posting each voter's identity and public information on $\mathcal{BB}$.
- Voters $(\mathcal{V}_1, \mathcal{V}_2, \cdots, \mathcal{V}_{n_V})$ are the entities participating in the election administrated by $\mathcal{R}$.
- Tallying authorities $(\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_{n_T})$ process ballots, jointly count votes and publish the final tally.

2.2 Basic Remote Voting Scheme

We now introduce our basic remote voting scheme, where there is only one candidate, and the choice of the election is either "Yes" or "No".

Setup: Our scheme is built on ElGamal (homomorphic and threshold) encryption scheme (ES) [9], the modified ElGamal signature scheme (SS) [19], the non-interactive zero-knowledge reencryption proof (ReencPf) [3,13], and the

non-interactive zero-knowledge equal discrete logarithm proof (EqDlog) [6], over a group $\mathbb{G}$ of a large prime order q with a generator g.

Let the choices of the election be $C = \{1, -1\}$, where $1, -1$ stand for "Yes" and "No", respectively.

Let the list of tallying authorities be $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_{n_T}\}$. Each $\mathcal{T}_i$ randomly chooses a private key $TSK_i = t_i$ from $\mathbb{Z}_q^*$ and computes the public key $TPK_i = g^{t_i}$. Let $TSK = \{t_1, t_2, \cdots, t_{n_T}\}$ and $TPK = \{g^{t_1}, g^{t_2}, \cdots, g^{t_{n_T}}\}$. Let h be chosen from a family of collision-resistant hash functions.

At last, the registrar posts $\Omega = \{\mathcal{R}, \mathsf{ES}, \mathsf{SS}, \mathsf{ReencPf}, \mathsf{EqDlog}, (\mathbb{G}, q, g), h, C, \mathcal{T}, TPK, \mathcal{V}\}$ on the public bulletin board $\mathcal{BB}$.

Registration: Before registration, each voter $\mathcal{V}_i$ generates a public/private key pair $(sk_i = x_i, pk_i = g^{x_i})$ for the signature scheme SS on her own device and prints out the public key pk_i and the hash value $h(pk_i)$ on paper. The purpose of using hash function is to facilitate human checking.

To vote, a voter $\mathcal{V}_i$ presents herself to a registrar's office, where $\mathcal{V}_i$ is allowed privately to press 1 or -1 button on a trusted entry device (e.g. PIN entry device), which, in turn, encrypts g or g^{-1} accordingly, and then prints out the hash value $h(R_i)$ on a slip, where $R_i = (A_i, B_i)$ is an encryption of either g or g^{-1}. Let $r_i = 1$, $A_i = g^{\gamma_i}$, $B_i = g(\prod_{t=1}^{n_T} TPK_t)^{\gamma_i}$ if press 1, and let $r_i = -1$, $A_i = g^{\gamma_i}$, $B_i = g^{-1}(\prod_{t=1}^{n_T} TPK_t)^{\gamma_i}$ if press -1, where γ_i is randomly chosen by the device from $\mathbb{Z}_q^*$. Therefore, R_i is an encryption of g^{r_i}. The voter $\mathcal{V}_i$ needs to remember her reference r_i. Having seen $h(R_i)$ on the slip, the voter $\mathcal{V}_i$ is allowed to confirm her choice by pressing "Confirm" or "Cancel" button on the device, like [1,16].

If $\mathcal{V}_i$ presses "Cancel", the device prints out r_i, γ_i, R_i on the slip for $\mathcal{V}_i$ to check if r_i is her choice. In this case, the staff in the registrar's office tears off the slip and provides a handwriting signature on it. $\mathcal{V}_i$ either keeps the slip for anyone later to check or inserts the slip into a locked box placed in the registrar's office for the election inspector with key later to check. Then the registration restarts. Note that anyone can check if R_i on the slip is computed correctly with r_i, γ_i, TPK without the knowledge of private keys of tallying authorities.

If the voter $\mathcal{V}_i$ presses "Confirm", the device scans her identity (denoted as $\mathcal{V}_i$ as well) from her identity card, her public key pk_i from her paper, and then computes the hash value $h(pk_i)$ and prints out $\mathcal{V}_i, h(pk_i)$ on the slip. The voter needs to check if the hash value $h(pk_i)$ on the slip is the same as that on her paper. At last, the device provides non-interactive zero-knowledge reencryption proof P_i (using ReencPf) that R_i is a re-encryption of either $(1, g)$ or $(1, g^{-1})$, posts $\mathcal{V}_i, pk_i, h(pk_i), R_i, h(R_i), P_i$ on $\mathcal{BB}$, and then erases r_i, γ_i from its memory. The staff tears off the slip with $h(R_i), \mathcal{V}_i, h(pk_i)$, provides a handwriting signature on it and then hands it to the voter.

Let the list of registered voters be $\mathcal{V} = \{\mathcal{V}_1, \mathcal{V}_2, \cdots, \mathcal{V}_{n_V}\}$. For each $\mathcal{V}_i$, there is a row $(\mathcal{V}_i, pk_i, h(pk_i), R_i, h(R_i), P_i)$ on $\mathcal{BB}$.

Voting: The registrar $\mathcal{R}$ announces the candidate on $\mathcal{BB}$. Each $\mathcal{V}_i$ chooses her vote v_i from $C = \{1, -1\}$ and determines β_i as follows: If $v_i = r_i$, then $\beta_i = 1$; If $v_i \neq r_i$, the $\beta_i = -1$. Note that $\mathcal{V}_i$ remembers her reference r_i.

Next, $\mathcal{V}_i$ generates a signature on β_i (using $\mathcal{SS}$) as follows: $S_i = g^{\delta_i}, T_i = (H(\beta_i, S_i) - S_i x_i){\delta_i}^{-1}(mod\ q)$, where δ_i is randomly chosen from $\mathbb{Z}_q^*$, H is a hash function and x_i is the private key of $\mathcal{V}_i$. Note that a time stamp may be included in the message to be signed to prevent replaying attacks.

Then, $\mathcal{V}_i$ constructs a ballot $b_i = \{\beta_i, S_i, T_i\}$ and casts it to $\mathcal{R}$, which, in turn, posts b_i next to $\mathcal{V}_i$ on $\mathcal{BB}$ if $g^{H(\beta_i, S_i)} = pk_i^{S_i} S_i^{T_i}$. The voter $\mathcal{V}_i$ checks if b_i on $\mathcal{BB}$ is the same as that she casts.

Tallying: To tally all valid ballots posted on $\mathcal{BB}$, $\mathcal{T}$ performs the following steps:

1. **Combining:** Based on the homomorphic property of ElGamal encryption scheme, all valid ballots $\{b_i\}_{i=1}^{n_V}$ on $\mathcal{BB}$ can be combined as follows: $X_T = \prod_{i=1}^{n_V} {A_i}^{\beta_i}$, $Y_T = \prod_{i=1}^{n_V} {B_i}^{\beta_i}$.
2. **Decrypting:** Following the threshold ElGamal encryption scheme, each tally authority $\mathcal{T}_i$ computes $X_i = X_T^{t_i}$ and posts X_i on $\mathcal{BB}$. With $\{X_i\}_{i=1}^{n_V}$, one can compute $Y_T \prod_{i=1}^{n_V} X_i^{-1} = \prod_{i=1}^{n_V} g^{r_i \beta_i} = \prod_{i=1}^{n_V} g^{v_i} = g^{\mathbf{y}-\mathbf{n}}$ can be obtained, where $\mathbf{y}, \mathbf{n}$ are the numbers of "Yes" and "No" and $\mathbf{y} + \mathbf{n} = n_V$. Since n_V is a small number relatively to q, $\mathbf{y}$ can be determined from $g^{2\mathbf{y}-n_V} = g^{\mathbf{y}-\mathbf{n}}$ by exhaustively searching $\mathbf{y}$ from 1 to n_V. At last, $\mathcal{T}$ release a tally $\mathbf{X} = (\mathbf{y}, \mathbf{n})$ on $\mathcal{BB}$.
3. **Proving:** $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_{n_T}$ jointly provide a multi-party non-interactive zero-knowledge proof P (using EqDlog) that $\prod_{i=1}^{n_T} TPK_i = g^{\sum_{i=1}^{n_T} t_i}$ and $g^{\mathbf{n}-\mathbf{y}} Y_T = X_T^{\sum_{i=1}^{n_T} t_i}$ have the equal discrete logarithm, and then posts the proof P next to $\mathbf{X}$ on $\mathcal{BB}$.

Verifying: During registration, each voter $\mathcal{V}_i$ is able to check if her public key pk_i and ciphertext R_i are posted on $\mathcal{BB}$ correctly on the basis of hash values $h(pk_i)$ and $h(R_i)$ on her registration slip. In addition, $\mathcal{V}_i$ is able to detect if the entry device in the registrar's office cheats by pressing "Cancel" and checking if r_i on the test slip is her choice and if R_i on the test slip is computed correctly by herself or with the help of someone later. During voting, each voter $\mathcal{V}_i$ is able to check whether β_i (either 1 or -1) in the ballot $b_i = \{\beta_i, S_i, T_i\}$ posted on $\mathcal{BB}$ is her choice even if the computer of $\mathcal{V}_i$ is infected by malware.

During registration, the election inspector is able to detect if the entry device cheats voters by collecting all test slips with the handwriting signatures of the registrar from the test box and checking if all ciphertexts are computed correctly. During voting, the public (including the voters) is able to verify if each R_i is an encryption of either g or g^{-1} based on the non-interactive zero-knowledge proof P_i, and check if each ballot b_i is valid with the signature (S_i, T_i) of $\mathcal{V}_i$. During tallying, the public can check if all valid ballots are combined and decrypted correctly based on the non-interactive zero-knowledge proof P.

Remark. Our basic scheme can fit a two-candidate election trivially.

2.3 General Remote Voting Scheme

Our basic remote voting scheme can be used to build a general remote voting scheme, where there is a list of candidates $\mathbb{C} = \{C_1, C_2, \cdots, C_{n_C}\}$, and the choice for each candidate is either "Yes" or "No".

Setup: Same as our basic scheme, the registrar $\mathcal{R}$ posts $\Omega = \{\mathcal{R}, \mathsf{ES}, \mathsf{SS}, \mathsf{ReencPf}, \mathsf{EqDlog}, (\mathbb{G}, q, g), h, C, \mathcal{T}, TPK, \mathcal{V}\}$ on the public bulletin board $\mathcal{BB}$.

Registration: For registration, a voter $\mathcal{V}_i$ presents herself with her printed public key pk_i and hash value $h(pk_i)$ to the registrar's office, where $\mathcal{V}_i$ is allowed privately to enter an integer $\mathbf{r}_i$ $(= a_{i,1} + a_{i,2}2 + \cdots + a_{i,n_C}2^{n_C-1}$, where $a_{i,j}$ is either 0 or 1) into a trusted entry device, which, in turn, encrypts a series of g and g^{-1} according to $a_{i,j}$. The ciphertext $R_{i,j} = (A_{i,j}, B_{i,j})$ is either $(g^{\gamma_{i,j}}, g(\prod_{t=1}^{n_T} TPK_t)^{\gamma_{i,j}})$ if $a_{i,j} = 0$ or $(g^{\gamma_{i,j}}, g^{-1}(\prod_{t=1}^{n_T} TPK_t)^{\gamma_{i,j}})$ if $a_{i,j} = 1$, where $\gamma_{i,j}$ is randomly chosen by the device from $\mathbb{Z}_q^*$. Then the device prints out the hash value $h(\mathbb{R}_i)$ on a slip, where $\mathbb{R}_i = \{R_{i,j}\}_{j=1}^{n_C}$. The voter $\mathcal{V}_i$ needs to remember her reference $\mathbf{r}_i$ (like a PIN number). If the number of candidates is large, $\mathcal{V}_i$ may write down $\mathbf{r}_i$ on a note privately.

Having seen $h(\mathbb{R}_i)$ on the slip, $\mathcal{V}_i$ decides whether to confirm $\mathbf{r}_i$. If not, the device prints out $\mathbf{r}_i$, $\{\gamma_{i,j}\}_{j=1}^{n_C}$ and $\mathbb{R}_i$ on the slip. In this case, the staff in the registrar's office tears off the slip, provides a handwriting signature on it. $\mathcal{V}_i$ either keeps the slip for anyone later to check or inserts the slip into a locked box placed in the registrar's office for the election inspector with key later to check. Then the registration restarts. Otherwise, the device scans the identity $\mathcal{V}_i$ and the public key pk_i and prints out $\mathcal{V}_i, h(pk_i)$ on the slip for $\mathcal{V}_i$ to check. At last, the device provides a non-interactive zero-knowledge reencryption proofs $\mathbb{P}_i$ (using $\mathsf{ReencPf}$) that each ciphertext in $\mathbb{R}_i$ is a re-encryption of either $(1, g)$ or $(1, g^{-1})$, and erases $\mathbf{r}_i, a_{i,j}, \gamma_{i,j}$ from its memory, and posts $\mathcal{V}_i, pk_i, h(pk_i), \mathbb{R}_i, h(\mathbb{R}_i), \mathbb{P}_i$ on $\mathcal{BB}$. The staff tears off the slip with $h(\mathbb{R}_i), \mathcal{V}_i, h(pk_i)$, provides a handwriting signature on it and then hands it to the voter.

Voting: The registrar $\mathcal{R}$ announces the list of candidates $\mathbb{C} = \{C_1, C_2, \cdots, C_{n_C}\}$ on $\mathcal{BB}$.

For each candidate C_j $(j = 1, 2, \cdots, n_C)$, a voter $\mathcal{V}_i$ chooses her vote $v_{i,j}$ from $\{1, -1\}$ and determines $\beta_{i,j}$ as follows: If $v_{i,j} = (-1)^{a_{i,j}}$, then $\beta_{i,j} = 1$; If $v_{i,j} \neq (-1)^{a_{i,j}}$, then $\beta_{i,j} = -1$. Note that $\mathcal{V}_i$ remembers her reference $\mathbf{r}_i = a_{i,1} + a_{i,2}2 + \cdots + a_{i,n_C}2^{n_C-1}$.

Next, $\mathcal{V}_i$ generates a signature on $\{\beta_{i,1}, \beta_{i,2}, \cdots, \beta_{i,n_C}\}$ as follows: $S_i = g^{\delta_i}, T_i = (H(\beta_{i,1}, \beta_{i,2}, \cdots, \beta_{i,n_C}, S_i) - S_i x_i){\delta_i}^{-1}$ $(mod\ q)$, where δ_i is randomly chosen from $\mathbb{Z}_q^*$, and x_i is the private key of $\mathcal{V}_i$.

Then, $\mathcal{V}_i$ constructs a ballot $b_i = \{\{\beta_{i,j}\}_{j=1}^{n_C}, S_i, T_i\}$ and casts it to $\mathcal{R}$, which, in turn, posts b_i next to $\mathcal{V}_i$ on $\mathcal{BB}$ if $g^{H(\beta_{i,1}, \beta_{i,2}, \cdots, \beta_{i,n_C}, S_i)} = pk_i^{S_i} S_i^{T_i}$. The voter $\mathcal{V}_i$ checks if b_i on $\mathcal{BB}$ is the same as that she casts.

Tallying: To tally all valid ballots posted on $\mathcal{BB}$ for each candidate C_j $(j = 1, 2, \cdots, n_C)$, $\mathcal{T}$ performs the following steps:

1. **Combining**: $\mathcal{T}$ combines all valid ballots on $\mathcal{BB}$ for the candidate C_j as follows: $X_{T,j} = \prod_{i=1}^{n_V} {A_{i,j}}^{\beta_{i,j}}, Y_{T,j} = \prod_{i=1}^{n_V} {B_{i,j}}^{\beta_{i,j}}$.
2. **Decrypting**: Each tally authority $\mathcal{T}_i$ computes $X_{i,j} = {X_{T,j}}^{t_i}$ and posts $X_{i,j}$ on $\mathcal{BB}$. By $\{X_{i,j}\}_{i=1}^{n_V}$, one can compute $Y_{T,j} \cdot \prod_{i=1}^{n_V} {X_{i,j}}^{-1} = \prod_{i=1}^{n_V} g^{\beta_{i,j}(-1)^{a_{i,j}}} = \prod_{i=1}^{n_V} g^{v_{i,j}} = g^{\mathbf{y}_j - \mathbf{n}_j}$ can be obtained, where $\mathbf{y}_j, \mathbf{n}_j$ are the numbers of "Yes" and "No"for the candidate C_j and $\mathbf{y}_j + \mathbf{n}_j = n_V$. Then $\mathbf{y}_j$ can be determined from $g^{2\mathbf{y}_j - n_V} = g^{\mathbf{y}_j - \mathbf{n}_j}$ by exhaustively searching $\mathbf{y}_j$ from 1 to n_V. At last, $\mathcal{T}$ release a tally $\mathbf{X}_j = (\mathbf{y}_j, \mathbf{n}_j)$ for the candidate C_j on $\mathcal{BB}$.
3. **Proving**: Tallying authorities $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_{n_T}$ jointly provide a multi-party non-interactive zero-knowledge proof P_{C_j} (using EqDlog) that $\prod_{i=1}^{n_T} TPK_i = g^{\sum_{i=1}^{n_T} t_i}$ and $g^{\mathbf{n}_j - \mathbf{y}_j} Y_{T,j} = {X_{T,j}}^{\sum_{i=1}^{n_T} t_i}$ have the equal discrete logarithm, and then posts the proof P_{C_j} next to $\mathbf{X}_j$ on $\mathcal{BB}$.

Verifying: Same as our basic voting scheme.

Remark: Our general scheme can fit a m out of n selection election (where $m < n$), in which m candidates are elected from n candidates $C_1, C_2, \cdots, C_n$, as long as we rank $\mathbf{y}_i - \mathbf{n}_i$ $(i = 1, 2, \cdots, n_C)$ after tallying. In addition, our general scheme can be extended to a ranked election. For example, considering a ranked election with 4 candidates C_1, C_2, C_3, C_4, a voter can rank them by 4 preferences $(+,+), (+,-), (-,+)$ and $(-,-)$. To implement this, each voter pre-sets two ciphertexts $R_{i,j,1}, R_{i,j,2}$ on $\mathcal{BB}$ for each candidate C_j. After voting, two columns of ciphertexts for C_j are tallied, respectively, and the tallying result for C_j can be $2(\mathbf{y}_{j,1} - \mathbf{n}_{j,1}) + (\mathbf{y}_{j,2} - \mathbf{n}_{j,2})$, where $(\mathbf{y}_{j,k}, \mathbf{n}_{j,k})$ is the tallying result of the k-th column of the ciphertexts for C_j.

2.4 Voter Reference Refresh

In our basic and general remote voting schemes, the reference of a voter can be used for one election only. For a new election, the voter may go to the registrar's office to reset her reference as the registration described above or refresh her reference online as follows.

For our basic scheme, when the voter $\mathcal{V}_i$ refreshes her reference r_i $(\in \{1, -1\})$, whose ciphertext on $\mathcal{BB}$ is $R_i = (A_i, B_i)$, she randomly chooses μ_i from $\{1, -1\}$ while her computer randomly chooses ρ_i from $\mathbb{Z}_q^*$ and computes $R'_i = (A'_i, B'_i) = (g^{\rho_i} A_i^{\mu_i}, (\prod_{t=1}^{n_T} TPK_t)^{\rho_i} B_i^{\mu_i})$, where R'_i is an encryption of $g^{r_i \mu_i}$ and the refresh reference $r'_i = r_i \mu_i$. Next the computer of $\mathcal{V}_i$ provides a non-interactive zero-knowledge reencryption proof P'_i that R'_i is a reencryption of either (A_i, B_i) or (A_i^{-1}, B_i^{-1}) and generates a signature on R'_i as follows: $S'_i = g^{\delta'_i}, T'_i = (H(R'_i, S'_i) - S'_i x_i) {\delta'_i}^{-1} (mod\ q)$, where δ'_i is randomly chosen from $\mathbb{Z}_q^*$ and x_i is the private key of $\mathcal{V}_i$. At last, $\mathcal{V}_i$ posts (R'_i, S'_i, T'_i, P'_i) next to $\mathcal{V}_i$ on $\mathcal{BB}$.

Remark. If an adversary coerces a voter $\mathcal{V}_i$ to compute R'_i with μ_i and ρ_i chosen by himself, he is uncertain of the refresh reference $r'_i = r_i \mu_i$ because he is uncertain of the the original reference r_i. In case the registrar obtains r_i, γ_i during the registration of $\mathcal{V}_i$, the registrar is uncertain of the refresh reference $r'_i = r_i \mu_i$ because he is uncertain of μ_i.

For our general scheme, when the voter $\mathcal{V}_i$ refreshes her reference $\mathbf{r}_i$ ($= a_{i,1} + a_{i,2}2 + \cdots + \cdots a_{i,n_C}2^{n_C-1}$), where $a_{i,j} \in \{0,1\}$), she randomly chooses $\mu_{i,1}, \mu_{i,2}, \cdots, \mu_{i,n_C}$ from $\{1, -1\}$, while her computer chooses random numbers $\rho_{i,1}, \rho_{i,2}, \cdots, \rho_{i,n_C}$ from $\mathbb{Z}_q^*$ and computes $\mathbb{R}'_i = \{(A'_{i,j}, B'_{i,j})\}_{j=1}^{n_C} = \{(g^{\rho_{i,j}} A_i^{\mu_{i,j}}, (\prod_{t=1}^{n_T} TPK_t)^{\rho_{i,j}} B_i^{\mu_{i,j}})\}_{j=1}^{n_C}$, where $\mathbb{R}'_i$ is the set of the encryptions of $g^{\mu_{i,1}(-1)^{a_{i,1}}}$, $g^{\mu_{i,2}(-1)^{a_{i,2}}}, \cdots, g^{\mu_{i,n_C}(-1)^{a_{i,n_C}}}$) and the refresh reference $\mathbf{r}'_i = a'_{i,1} + a'_{i,2}2 + \cdots + a'_{i,n_C}2^{n_C-1}$, where $a'_{i,j} = \frac{1-\mu_{i,j}(-1)^{a_{i,j}}}{2}$. Next $\mathcal{V}_i$ provides a non-interactive zero-knowledge reencryption proof $\mathbb{P}'_i$ that each $(A'_{i,j}, B'_{i,j})$ in $\mathbb{R}'_i$ is a reencryption of either $(A_{i,j}, B_{i,j})$ or $(A_{i,j}^{-1}, B_{i,j}^{-1})$ and generates a signature on $\mathbb{R}'_i$ as follows: $S'_i = g^{\delta'_i}, T'_i = (H(\mathbb{R}'_i, S'_i) - S'_i x_i)\delta_i'^{-1} (mod\ q)$,where δ'_i is randomly chosen from $\mathbb{Z}_q^*$ and x_i is the private key of $\mathcal{V}_i$. At last, $\mathcal{V}_i$ posts $(\mathbb{R}'_i, S'_i, T'_i, \mathbb{P}_i)$ next to $\mathcal{V}_i$ on $\mathcal{BB}$.

Remark. As a voter $\mathcal{V}_i$ is able to test if the entry device in the registrar's office is cheating during the registration, $\mathcal{V}_i$ is able to test if her computer is cheating during voter reference refresh by sending test data to the election inspector by post.

3 Security Proofs

3.1 Definition of Coercion Resistance

Formally, the definition of coercion resistance is built on a game between an adversary and a voter targeted by the adversary for coercive attack, where a coin is flipped; the outcome is represented as a bit b; if $b = 0$, then the voter provides the adversary with a false reference; if $b = 1$, then the voter furnishes the adversary with the true reference. The task of the adversary is to guess the value of coin b.

In this game, we assume that there are n_V eligible voters for the election, and the number of vote choices is 2 only. The adversary has corrupted (i.e., completely controlled) a minority of tallying authorities and n_A voters. As convention, we consider the case where the adversary attempts to coerce a single voter in the game. Extension to coercion of multiple voters is straightforward. Therefore, there are $n_U = n_V - n_A - 1$ voters outside the control of the adversary. In other word, the n_U voters are not subject to coercion. We model the voting patterns of the n_U voters in terms of a probability distribution $D_{n_U,2}$. In the same way, we model the voting patterns of the n_A corrupted voters as a probability distribution $D^A_{n_A,2}$, which represents the strategy of the adversary.

We assume that the adversary is uncertain of $D_{n_U,2}$ in the game. In addition, we assume that the election tally $\mathbf{X}$ released in the tally phase is a pair $(\mathbf{y}, \mathbf{n})$, where $\mathbf{y}, \mathbf{n}$ stand for the numbers of "Yes" and "No", respectively. In our model, the abstention case, where a voter has registered the election, but does not cast her ballot, can be easily identified and excluded. In view of it, we do not consider the abstention case in our model.

We consider a static adversary, i.e., one that selects voters to be corrupt prior to protocol execution. We assume that the adversary has a list of "voter names", i.e., a roll of potential participating voters. Let $\leftarrow$ denote assignment and $\Leftarrow$ denote the append operation. The game can be illustrated in Fig. 1.

Experiment $\mathbf{Exp}^{c-resist}_{\mathsf{VS},\mathcal{A}}(k)$

$\mathcal{BB} \Leftarrow \Omega, (\Omega = \{\mathcal{R}, \mathsf{ES}, \mathsf{SS}, C, \mathcal{T}, TPK, \mathcal{V}\}, TSK) \leftarrow \mathsf{setup}(1^k)$

$\mathcal{V}_A \leftarrow \mathcal{A}(n_A$, voter names, "control voters")

$\{\mathcal{BB} \Leftarrow (\mathcal{V}_i, pk_i, R_i, P_i), (\mathcal{V}_i, pk_i, R_i, P_i, r_i, sk_i) \leftarrow \mathsf{registration}(\mathcal{R}, \mathcal{V}_i, TPK)\}_{i=1}^{n_V}$

$\mathcal{V}_j \leftarrow \mathcal{A}\{\mathcal{BB}$, "set target voter"$\}$

if $|\mathcal{V}_A| \neq n_A \vee V_j \notin \mathcal{V} - \mathcal{V}_A$, then output "0"

$b \in_R \{0, 1\}$

if $b = 0$ then

  $\tilde{r}_j \leftarrow \mathsf{fakeReference}(\mathcal{V}_j, r_j, R_j, TPK)$

else

  $\tilde{r}_j \leftarrow r_j$

$\mathcal{BB} \Leftarrow b_i \leftarrow \mathsf{vote}(\mathcal{V}_i, v_i, r_i, sk_i)_{\mathcal{V}_i \in \mathcal{V}_A}$,where v_i obeys $D^{\mathcal{A}}_{n_A,2}$

$\mathcal{BB} \Leftarrow b_i \leftarrow \mathsf{vote}(\mathcal{V}_i, v_i, r_i, sk_i)_{\mathcal{V}_i \in \mathcal{V} - \mathcal{V}_A \wedge \mathcal{V}_i \neq \mathcal{V}_j}$,where v_i obeys $D_{n_U,2}$

$\mathcal{BB} \Leftarrow b_j \leftarrow \mathsf{vote}(\mathcal{V}_j, v_j, \tilde{r_j}, sk_j)$

$(\mathbf{X}, P) \leftarrow \mathsf{tally}(\mathcal{T}, \Omega, \{\mathcal{V}_i, pk_i, R_i, P_i, b_i\}_{i=1}^{n_V}, TSK)$

$\mathcal{BB} \Leftarrow (\mathbf{X}, P)$

$b' \leftarrow \mathcal{A}(\mathcal{BB}$,"guess b")

if $b' = b$ then

  output "1"

else

  output "0"

halt

Fig. 1. The experiment for definition of coercion resistance

We consider a set of experiments as shown in Fig. 1. The adversary is assumed to retain states throughout the duration of an experiment. We say an adversary $\mathcal{A}$ succeeds if the experiment outputs "1". The advantage of an adversary $\mathcal{A}$ against the voting scheme VS is a function in the security parameter k, defined as $\mathsf{Adv}^{c-resist}_{\mathsf{VS},\mathcal{A}}(k) = \mathscr{P}[\mathsf{Exp}^{c-resist}_{\mathsf{VS},\mathcal{A}}(k) =$ "1"$]$, where the probability is taken over the random coins used by the adversary and the random coins used during the course of the experiment.

It remains to define what we mean by a coercion-resistant voting scheme. Note that the final election tally $\mathbf{X}$ is released to the public after tallying. Based on $\mathbf{X}$, the adversary can infer $\mathbf{X} - \mathbf{X}_A = (\mathbf{y} - \mathbf{y}_A, \mathbf{n} - \mathbf{n}_A)$, where $\mathbf{y}_A, \mathbf{n}_A$ are the numbers of "Yes" and "No" cast by $\mathcal{V}_A$. Although r_j is uncertain to the adversary, the probability of the adversary in guessing b on the basis of $\mathbf{X} - \mathbf{X}_A$ is usually higher than $1/2$. A voting scheme is coercion-resistant if the adversary cannot succeed in guessing b better than an adversarial algorithm whose only input is $\mathbf{X} - \mathbf{X}_A$.

Definition 1. A voting scheme VS is coercion-resistant if, for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ and any probability distributions $D_{n_U,2}$

and $D^A_{n_A,2}$, there exists a negligible function $\varepsilon(\cdot)$ such that $\mathsf{Adv}^{c-resist}_{\mathsf{VS},\mathcal{A}}(k) \leq \mathscr{P}_{\mathbf{X}-\mathbf{X}_A}(k)+\varepsilon(k)$, where $\mathscr{P}_{\mathbf{X}-\mathbf{X}_A}(k)$ stands for the probability of the adversary in guessing b on the basis of $\mathbf{X}-\mathbf{X}_A$ only.

The above definition ensures that the adversary can (essentially) do no better than guess b on the basis of $\mathbf{X}-\mathbf{X}_A$ only. This means that the adversary learns nothing from the private keys and references she acquires from corrupted voters and from the coerced voter.

3.2 Security Proof of Coercion Resistance

Given a probabilistic polynomial time-bounded (PPT) adversary $\mathcal{A}$, we imagine a simulator $\mathcal{S}$ that runs the game as shown in Fig. 1 for $\mathcal{A}$. More preciously, the simulator begins by generating a multiplicative cyclic group $(\mathbb{G}, q, g)$, and the public and private keys (TPK_i, TSK_i) $(i = 1, 2, \cdots, n_T)$ for all tallying authorities $\mathcal{T}$, and posting Ω on $\mathcal{BB}$.

During registration, the simulator $\mathcal{S}$, playing the role of the registrar $\mathcal{R}$, generates public and private key pairs (pk_i, sk_i) for $n_C - n_A$ honest voters, while the adversary $\mathcal{A}$ generates public and private key pairs (pk_i, sk_i) for n_A controlled voters. When a (controlled or honest) voter $\mathcal{V}_i$ registers the election, she randomly chooses r_i from $\{1, -1\}$ and sends r_i, pk_i to $\mathcal{S}$, which, in turn, encrypts g^{r_i} and posts pk_i and the ciphertext R_i with a non-interactive zero-knowledge proof P_i that each R_i is an encryption of either g or g^{-1} on $\mathcal{BB}$.

During voting, $\mathcal{A}$ constructs n_A ballots for controlled voters in $\mathcal{V}_A$ and posts them on $\mathcal{BB}$, and sets a target voter $\mathcal{V}_j \in \mathcal{V} - \mathcal{V}_A$ to coerce. On coercing, $\mathcal{S}$ randomly chooses a bit b. If $b = 0$, $\mathcal{S}$ gives r_j and sk_j to $\mathcal{A}$. Otherwise, $\mathcal{S}$ gives $-r_j$ and sk_j to $\mathcal{A}$. Then $\mathcal{A}$ constructs a ballot for $\mathcal{V}_j$ and posts it on $\mathcal{BB}$. In addition, $\mathcal{S}$ constructs n_U ballots for voters in $\mathcal{V} - \mathcal{V}_A - \{\mathcal{V}_j\}$ and posts them on $\mathcal{BB}$.

During tallying, $\mathcal{S}$, playing the role of tallying authorities, combines all valid ballots, decrypts the resulted ciphertext (with the private keys of tally authorities) and extracts a final tally $\mathbf{X} = (\mathbf{y}, \mathbf{n})$, and provides a non-interactive zero-knowledge proof P that the decryption is correct. Then $\mathcal{S}$ releases $(\mathbf{X}, P)$ on $\mathcal{BB}$.

At last, $\mathcal{A}$ submits a guess b' to $\mathcal{S}$. If $b' = b$, $\mathcal{A}$ wins the game. Otherwise, $\mathcal{A}$ fails. We denote this original experiment as Exp_0.

In the view of $\mathcal{A}$, there are three types of encrypted votes on $\mathcal{BB}$, i.e., (1) those cast by $\mathcal{V}_A$; (2) one cast by $\mathcal{V}_j$; (3) those cast by $\mathcal{V} - \mathcal{V}_A - \{\mathcal{V}_j\}$. $\mathcal{A}$ is uncertain the votes corresponding to the encrypted votes in (2) and (3), but $\mathcal{A}$ can deduce their tally $\mathbf{X} - \mathbf{X}_A = (\mathbf{y} - \mathbf{y}_A, \mathbf{n} - \mathbf{n}_A)$, where $\mathbf{X}_A = (\mathbf{y}_A, \mathbf{n}_A)$ is the tally of votes cast by $\mathcal{V}_A$.

If we can replace the encrypted votes in (2) and (3) with the encryptions of a random permutation of these votes and yet $\mathcal{A}$ cannot distinguish this modified experiment from the original experiment Exp_0 with a non-negligible probability, we can conclude that $\mathcal{A}$ can (essentially) do no better than guess b on the basis of $\mathbf{X} - \mathbf{X}_A$ only, and therefore our voting scheme is coercion-resistance on the basis of Definition 1.

Notice that any permutation can be expressed as the composition (product) of transpositions. We only need to consider a modified experiment where two

honest voters $\mathcal{V}_k$ and $\mathcal{V}_\ell$ in $\mathcal{V} - \mathcal{V}_A$ exchange their votes v_k and v_ℓ ($v_k \neq v_\ell$). We denote this experiment as Exp_1.

Theorem 1. The difference between the advantages of the adversary $\mathcal{A}$ in the original experiment Exp_0 and the modified experiment Exp_1 is negligible, if at least one tally authority is honest and ElGamal encryption scheme is semantically secure under chosen plaintext attack.

Proof. If the difference between the advantages of the adversary $\mathcal{A}$ in the original experiment Exp_0 and the modified experiment Exp_1 is non-negligible, a simulator $\mathcal{S}$ can make use $\mathcal{A}$ as a subroutine to break the semantic security of ElGamal encryption scheme as follows.

According to the game described in Section 3.1, the challenger of ElGamal encryption scheme runs the key generation algorithm, gives the public key y to the simulator $\mathcal{S}$, but keeps the private key x to itself.

The simulator $\mathcal{S}$ runs an experiment as shown in Exp_0 except that:

- Because at least one tally authority is honest, we assume that the tally authority $\mathcal{T}_{n_T}$ is honest without loss of generality. During setup, $\mathcal{S}$ generates a public and private key pair (TPK_i, TSK_i) for the tally authority $\mathcal{T}_i$ $(i = 1, 2, \cdots, n_T - 1)$. In particular, $\mathcal{S}$ computes the public key $TPK_{n_T} = y / \prod_{i=1}^{n_T - 1} TPK_i$ for the tally authority $\mathcal{T}_{n_T}$. Then $\mathcal{S}$ reveals (TPK_i, TSK_i) $(i = 1, 2, \cdots, n_T - 1)$ to the adversary $\mathcal{A}$.Note that at least one tally authority is honest and thus $\mathcal{A}$ cannot query the private key of $\mathcal{T}_{n_T}$.
- During registration, for voters $\mathcal{V}_k$ and $\mathcal{V}_\ell$, $\mathcal{S}$ challenges two plaintexts $M_0 = g$ and $M_1 = g^{-1}$ and the challenger picks a random bit $\hat{b} \in \{0, 1\}$ and sends back $R_k = \mathsf{E}(M_{\hat{b}}, y) = (A_k, B_k)$ and $R_\ell = \mathsf{E}(M_{1-\hat{b}}, y) = (A_\ell, B_\ell)$ as the challenge along with non-interactive zero-knowledge reencryption proofs P_k and P_ℓ that R_k and R_ℓ are reencryptions of either $(1, g)$ or $(1, g^{-1})$. The simulator $\mathcal{S}$ posts them on $\mathcal{BB}$ for $\mathcal{V}_k$ and $\mathcal{V}_\ell$, respectively.
- During voting, the simulator $\mathcal{S}$ ensures $\beta_k = \beta_\ell$ even if either $\mathcal{V}_k$ or $\mathcal{V}_\ell$ is coerced (i.e., even if either $\mathcal{V}_k = \mathcal{V}_j$ or $\mathcal{V}_\ell = \mathcal{V}_j$). This can be always achieved because the adversary can coerce one voter from $\mathcal{V} - \mathcal{V}_A$ only.
- During tallying, $\mathcal{S}$ can determine $g^{\mathbf{y-n}}$ without any decryption because $\mathcal{S}$ knows $g^{r_i \beta_i}$ for all i except k and ℓ. Since $g^{\mathbf{y-n}} = \prod_{i=1}^{n} {B_i}^{\beta_i} / (\prod_{i=1}^{n_V} {A_i}^{\beta_i})^x$, the simulator $\mathcal{S}$ can commit X_{n_T} in the decrypting stage. In addition, the simulator $\mathcal{S}$ sends β_k $(= \beta_\ell)$ and $(\gamma_i, g^{r_i}, (A_i, B_i), \beta_i)$ for all i except from k and ℓ to the challenger and asks it to provide with a non-interactive zero-knowledge proof P that $y = g^x$ and $g^{\mathbf{n-y}} \prod_{i=1}^{n_V} {B_i}^{\beta_i} = (\prod_{i=1}^{n_V} {A_i}^{\beta_i})^x$ have the same discrete logarithm. Based on $g^{\mathbf{y-n}}$, P and information provided by $\mathcal{A}$, the simulator $\mathcal{S}$ can determine the part of the proof P for $\mathcal{T}_{n_T}$.

Depending on the value of b, the above experiment is either Exp_0 or Exp_1. If the difference between the advantages of the adversary $\mathcal{A}$ in Exp_0 and Exp_1 is non-negligible, the simulator can guess b correctly with probability significantly greater than $1/2$. This is in contradiction with the assumption that ElGamal encryption scheme is semantically secure against chosen plaintext attack. Therefore, the theorem is proved.

Experiment $\mathbf{Exp}^{ver}_{\mathsf{VS},\mathcal{A}}(k)$

$\mathcal{BB} \Leftarrow \Omega, (\Omega, TSK) \leftarrow \mathsf{setup}(1^k)$

$\mathcal{V}_A \leftarrow \mathcal{A}(n_A$, voter names, "control voters")

$\{\mathcal{BB} \Leftarrow (\mathcal{V}_i, pk_i, R_i, P_i), (\mathcal{V}_i, pk_i, R_i, P_i, r_i, sk_i) \leftarrow \mathsf{registration}(\mathcal{R}, \mathcal{V}_i, TPK)\}_{i=1}^{n_V}$

$\mathcal{V}_j \leftarrow \mathcal{A}\{(\mathcal{BB}, TSK)$, "set target voter"$\}$

if $|\mathcal{V}_A| \neq n_A \vee V_j \notin \mathcal{V} - \mathcal{V}_A$, then output "0"

$\mathcal{BB} \Leftarrow b_i \leftarrow \mathsf{vote}(\mathcal{V}_i, v_i, r_i, sk_i)_{\mathcal{V}_i \in \mathcal{V}_A}$, where v_i obeys $D^A_{n_A,2}$

$\mathcal{BB} \Leftarrow b_i \leftarrow \mathsf{vote}(\mathcal{V}_i, v_i, r_i, sk_i)_{\mathcal{V}_i \in \mathcal{V} - \mathcal{V}_A \wedge \mathcal{V}_i \neq \mathcal{V}_j}$, where v_i obeys $D_{n_U,2}$

$\mathcal{A} \Leftarrow b_j \leftarrow \mathsf{vote}(\mathcal{V}_j, v_j, r_j, sk_j)$

$\mathcal{BB} \Leftarrow b^A_j \leftarrow \mathcal{A}$("forge ballot", $\mathcal{V}_j, b_j, pk_j, TSK)$

$\mathcal{A} \Leftarrow (\mathbf{X}, P) \leftarrow \mathsf{tally}(\mathcal{BB}, TSK)$

$\mathcal{BB} \Leftarrow (\mathbf{X}', P') \leftarrow \mathcal{A}$("forge tally", $(\mathbf{X}, P), TSK)$

if $\mathsf{verify}(\mathcal{BB}) = 1 \wedge [(\exists \mathcal{V}_i : \mathsf{D}(R_i, TSK) \notin \{g, g^{-1}\}) \vee (\beta^A_j \neq \beta_j) \vee$
$(\exists \mathcal{V}_i : \mathsf{D}(R_i, TSK) = g^{-r_i}) \vee (\mathbf{X}' \neq \mathbf{X})]$

  output "1"

else

  output "0"

halt

Fig. 2. The experiment for definition of verifiability

3.3 Definition of Verifiability

Informally, a voting scheme is verifiable if any voter can verify that her ballot is included unmodified in a collection of ballots, and the public can verify that the collection of ballots produces the correct final tally. The verifiability is for the public (including voters) to be able to detect any misbehavior by the registrar and tally authorities.

Formally, the definition of verifiability is built on a game between an adversary $\mathcal{A}$ and the public (including voters), where $\mathcal{A}$ has corrupted the registrar, all tallying authorities and n_A voters. The task of $\mathcal{A}$ is to cause an incorrect tally to be accepted by the public. The game can be illustrated in Fig. 2. Note that the contents of $\mathcal{BB}$ changes depending on different phases.

We say an adversary $\mathcal{A}$ succeeds if the experiment outputs "1". The advantage of an adversary $\mathcal{A}$ against the voting scheme VS is a function in the security parameter k, defined as $\mathsf{Adv}^{ver}_{\mathsf{VS},\mathcal{A}}(k) = \mathscr{P}[\mathsf{Exp}^{ver}_{\mathsf{VS},\mathcal{A}}(k) =$ "1"$]$, where the probability is taken over the random coins used by adversary $\mathcal{A}$ and the random coins used during the course of the experiment.

Definition 2. A voting scheme VS is verifiable if, for any probabilistic polynomial-time (PPT) adversaries $\mathcal{A}$, any probability distributions $D_{n_V - n_A,2}$ and $D^A_{n_A,2}$, there exists a negligible function $\varepsilon(\cdot)$ such that $\mathsf{Adv}^{ver}_{\mathsf{VS},\mathcal{A}}(k) \leq \varepsilon(k)$.

3.4 Security Proof of Verifiability

As described in Section 3.3, we assume that the adversary $\mathcal{A}$ has corrupted the registrar and all tallying authorities $\mathcal{T}$ plus n_A voters and intends to cause a fake election tally $\mathbf{X}'$ to be accepted by the public. $\mathcal{A}$ can only win the game in one of the following four events.

- During registration, $\mathcal{A}$ causes $\mathsf{D}(R_i, TSK) \notin \{g, g^{-1}\}$ for a voter $\mathcal{V}_i$ when $\mathsf{verify}(\mathcal{BB}) = 1$. We denote the event as Succ_1.
- During registration, $\mathcal{A}$ causes $\mathsf{D}(R_i, TSK) \neq g^{-r_i}$ for a (honest) voter $\mathcal{V}_i$ when $\mathsf{verify}(\mathcal{BB}) = 1$. We denote the event as Succ_2.
- During voting, $\mathcal{A}$ causes $\beta_j^A \neq \beta_j$ for a target voter $\mathcal{V}_j$ when $\mathsf{verify}(\mathcal{BB}) = 1$. We denote the event as Succ_3.
- During tallying, $\mathcal{A}$ causes $\mathbf{X}' \neq \mathbf{X}$ where $(\mathbf{X}, P) = \mathsf{tally}(\mathcal{BB}, TSK)$ when $\mathsf{verify}(\mathcal{BB}, \mathbf{X}', P') = 1$. We denote the event as Succ_4.

Obviously, $\mathsf{Adv}^{ver}_{\mathsf{VS},\mathcal{A}}(k) \leq \mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_1] + \mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_2] + \mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_3] + \mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_4]$.

If $\mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_1]$ is non-negligible, a simulator $\mathcal{S}$ can make use $\mathcal{A}$ as a subroutine to break the security of the non-interactive zero-knowledge reencryption proof scheme ($\mathsf{ReencPf}$) [13] in the similar way as we prove Theorem 1.

Next, let us analyze $\mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_2]$. During registration, a (honest) voter $\mathcal{V}_i$ sends her reference $r_i \in \{1, -1\}$ to the adversary $\mathcal{A}$ who is playing the role of the entry device. To facilitate analysis, we assume the entry device prints out the ciphertext at first instead of its hash value. Our scheme still works without the hash function h. Because $\mathcal{A}$ cannot predict when $\mathcal{V}_i$ confirms her reference, $\mathcal{V}$ has to print out $R_i = (g^{\gamma_i}, g^{-r_i}(\prod_{t=1}^{n_T} TPK_t)^{\gamma_i})$ at first to win the game. In case $\mathcal{V}_i$ presses "Cancel" button, $\mathcal{A}$ has to print out r_i, γ_i' such that $R_i = (g^{\gamma_i'}, g^{r_i}(\prod_{t=1}^{n_T} TPK_t)^{\gamma_i'})$ to avoid being detected. In fact, $g^{\gamma_i'} = g^{\gamma_i}$ implies $\gamma_i' = \gamma_i (mod\ q)$. Therefore, $g^{-r_i}(\prod_{t=1}^{n_T} TPK_t)^{\gamma_i} = g^{r_i}(\prod_{t=1}^{n_T} TPK_t)^{\gamma_i}$ must hold. However it cannot hold even if the adversary $\mathcal{A}$ has known all private keys of tallying authorities.

In addition, if $\mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_3]$ is non-negligible, a simulator $\mathcal{S}$ can make use $\mathcal{A}$ as a subroutine to break the security of the modified ElGamal signature scheme (SS) [19]. If $\mathsf{Pr}_{\mathcal{A}}[\mathsf{Succ}_4]$ is non-negligible, a simulator $\mathcal{S}$ can make use $\mathcal{A}$ as a subroutine to break the security of the non-interactive zero-knowledge equal discrete logarithm proof scheme (EqDlog) [6]. Based on Definition 2, we have

Theorem 2. Assume that (1) the modified ElGamal signature scheme (SS) [19] is existentially unforgeable under the chosen message attack, over a group $\mathbb{G}$ of a large prime order q with a generator g; (2) the non-interactive zero-knowledge reencryption proof ($\mathsf{ReencPf}$) [13] is secure; (3) the non-interactive zero-knowledge equal discrete logarithm proof (EqDlog) [6] is secure, our basic voting scheme is verifiable even if all election authorities are corrupt.

4 Conclusion

In this paper, we present a remote voting scheme based on homomorphic encryption and provide a proof of coercion-resistance and verifiability for our scheme.

References

1. Benaloh, J.: Ballot casting assurance via voter-initiated poll station auditing. In: Proc. Electronic Voting Technology Workshop, EVT 2007 (2007)
2. Benaloh, J., Tuinstra, D.: Receipt-free secret-ballot elections (extended abstract). In: Proc. 26th ACM STOC 1994, pp. 544–553 (1994)
3. Blum, M., Santis, A.D., Micali, S., Persiano, G.: Non-interactive zero-knowledge. SIAM Journal on Computing 6, 1084–1118 (1991)
4. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 24(2), 84–88 (1981)
5. Chaum, D.: Punchscan (2005), `http://www.punchscan.org`
6. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer, Heidelberg (1993)
7. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: A secure remote voting system. In: Proc. IEEE Symposium on Security and Privacy, pp. 354–368 (2008)
8. Cohen, J. D.(Benaloh), Fischer, M.J.: A robust and verifiable cryptographically secure election scheme. In: Proc. FOCS 1985, pp. 372–382 (1985)
9. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31, 469–472 (1985)
10. Fujioka, A., Okamoto, T., Ohta, K.: A practical secret voting scheme for large scale elections. In: Zheng, Y., Seberry, J. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 244–251. Springer, Heidelberg (1993)
11. Gardner, R.W., Garera, S., Rubin, A.D.: Coercion resistant end-to-end voting. In: Dingledine, R., Golle, P. (eds.) FC 2009. LNCS, vol. 5628, pp. 344–361. Springer, Heidelberg (2009)
12. Golle, P., Jakobsson, M., Juels, A., Syverson, P.: Universal re-encryption for mixnets. In: Okamoto, T. (ed.) CT-RSA 2004. LNCS, vol. 2964, pp. 163–178. Springer, Heidelberg (2004)
13. Hirt, M., Sako, K.: Efficient receipt-free voting based on homomorphic encryption. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 539–556. Springer, Heidelberg (2000)
14. Jakobsson, M., Juels, A., Rivest, R.: Making mix nets robust for electronic voting by randomized partial checking. In: Proc. USENIX 2002, pp. 339–353 (2002)
15. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic election. In: Proc. WPES 2005, pp. 61–70 (2005)
16. Kutylowski, M., Zagorski, F.: Scratch, click & vote: E2E voting over the Internet. In: NIST End-to-End Voting System Workshop (2009)
17. Moran, T., Naor, M.: Split-ballot voting: everlasting privacy with distributed trust. In: Proc. CCS 2007, pp. 246–255 (2007)
18. Neff, A.: A verifiable secret shuffle and its application to e-voting. In: Proc. ACM CCS 2001, pp. 116–125 (2001)
19. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 387–398. Springer, Heidelberg (1996)
20. Rivest, R.L., Smith, W.D.: Three voting protocols: Threeballot, VAV, and twin. In: Proc. Electronic Voting Technology Workshop, EVT 2007 (2007)
21. Sako, K., Kilian, J.: Receipt-free mix-type voting scheme. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 393–403. Springer, Heidelberg (1995)
22. Teague, V., Ramchen, K., Naish, L.: Coercion-resistant tallying for STV voting. In: Proc. Electronic Voting Technology Workshop, EVT 2008 (2008)