An Anonymous Voting Scheme based on Confirmation Numbers

Paper

An Anonymous Voting Scheme based on Confirmation Numbers

Kazi Md. Rokibul Alam*, Non-member
Shinsuke Tamura*, Member
Shuji Taniguchi*, Non-member
Tatsuro Yanase*, Non-member

This paper proposes a new electronic voting (e-voting) scheme that fulfills all the security requirements of e-voting *i.e.* privacy, accuracy, universal verifiability, fairness, receipt-freeness, incoercibility, dispute-freeness, robustness, practicality and scalability; usually some of which are found to be traded. When compared with other existing schemes, this scheme requires much more simple computations and weaker assumptions about trustworthiness of individual election authorities. The key mechanism is the one that uses confirmation numbers involved in individual votes to make votes verifiable while disabling all entities including voters themselves to know the linkages between voters and their votes. Many existing e-voting schemes extensively deploy zero-knowledge proof (ZKP) to achieve verifiability. However, ZKP is expensive and complicated. The confirmation numbers attain the verifiability requirement in a much more simple and intuitive way, then the scheme becomes scalable and practical.

Keywords : Encryption/Decryption Shuffles, Incoercibility, Universal Verifiability, Confirmation Numbers, Signature Pairs.

1. Introduction

Unlike paper-based systems, electronic voting (e-voting) systems based on computers, computer networks and cryptographic protocols enable efficient, accurate, secure, and convenient elections. Also resources of e-voting schemes are reusable, therefore e-voting based elections become inexpensive. Moreover, they do not require any geographical proximity of voters and they provide better scalability for large elections.

Ideal e-voting schemes must satisfy privacy, accuracy, universal verifiability, fairness, receipt-freeness, incoercibility, dispute-freeness, robustness, scalability and practicality [(1), (2), (9)]. However, there are tradeoffs among them and satisfying all requirements at the same time is really difficult. For examples, verifiability requires voters to be linked to their votes, and hence is in contradiction to privacy. Also, achieving incoercibility leads to sacrificing universal verifiability and hence accuracy [(1)], and satisfying dispute-freeness makes schemes complicated and consequently schemes become impractical or unscalable. Therefore many existing e-voting schemes can satisfy only part of the above requirements.

To overcome these difficulties, this paper proposes a new e-voting scheme [(18)]. The scheme satisfies all the security requirements of e-voting systems listed above. Also this scheme is based on weaker assumptions about trustworthiness of election authorities, *i.e.* nothing can make the scheme unreliable if at least one authority is honest among multiple authorities, and the way of candidate selections is flexible; it accepts freely chosen write-in ballots, votes for a pre-specified or *t* out of *l* choices as well as yes/no votes.

* Graduate School of Engineering, University of Fukui
3-9-1, Bunkyo, Fukui, 910-8507

2. Related Works

Among various security requirements, receipt-freeness and incoercibility are especically difficult to satisfy although they are essential for voting, where receipt-freeness disables voters to prove their votes to any entity including themselves in order to achieve incoercibility. Several mixnet and homomorphic encryption based voting schemes [(2), (3), (4)] achieve receipt-freeness by attaching secret random numbers to votes while proving the correctness of votes by using interactive-zero-knowledge proof (IZKP) or non-interactive-ZKP (NIZKP). However ZKP that requires non negligable computations makes the schemes impractical. Also untappable channels, physically secure but unobservable communication channels, used in them make them unrealistic [(3), (4)]. A worse thing is that these schemes cannot achieve the complete receipt-freeness. Namely, authorities can know the random numbers and use them to link voters to their votes. Although tamper-resistant randomizer (TRR) [(2)], a hardware device to generate random numbers for voters, achieves the complete anonymity of voters, TRR further worsens its practicality.

About the incoercibility, in existing schemes [(5), (6), (7)], each voter constructs its encrypted vote while attaching the encrypted token assigned to it to submit its multiple votes without being traced by others. As a consequence, coercers cannot identify the exact vote of the voter. However, ZKP to confirm the equivalence of tokens corresponded to multiple votes of same voters sacrifices practicality and scalability.

Although e-voting schemes based on the blind signature [(12), (13)] do not exploit ZKP, usually these schemes cannot satisfy universal verifiability nor receipt-freeness because voters' blinding factors can be used as receipts of their votes, and therefore voters can prove their votes to buyers. Besides, these schemes assume the existence of anonymous channels which are impractical.

Regarding paper based voting schemes, visual cryptography based schemes had been proposed [8], [9], [11]. However, in these systems, voters must delegate their vote computations to the voting booth, therefore the voting booth can know the votes of the voters, by which the privacy of voters may be breached. Paper ballots prepared in advance do not guarantee privacy against the ballot creators either [9]. Although solutions exist for these problems [9], they require NIZKP and assume the existence of recordable private channels which are impractical.

3. Contributions of the Proposed Scheme

To enable the proposed scheme to satisfy all the requirements, 3 security components had been newly developed; they are confirmation numbers (*CNs*), signature pairs on encrypted votes, and those on blinded tokens. Here *CNs* are unique and registered numbers that are publicly disclosed in their encrypted forms, and they are assigned to individual voters to make votes verifiable. Because all *CNs* are publicly disclosed, anyone can convince itself that votes attached by *CNs* are the ones submitted in the authorized way. Also by examining the used *CNs*, anyone can confirm that all submitted votes are counted. A signature pair on v is a pair of signatures on v generated by different signing keys and ensures authenticity of even meaningless v when it has a consistent signature pair. Figure 1 shows the roles of voter V_j, its vote v_j, and pairs of signatures on v_j, confirmation number C_{Cj} and token T_{tj} assigned to V_j.

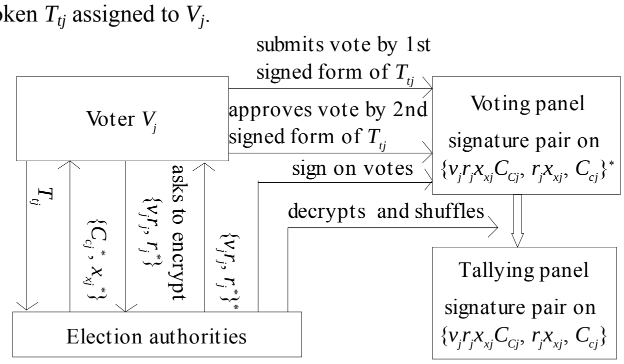

Fig. 1 Roles of V_j, v_j, pairs of signatures on v_j, C_{Cj} and T_{tj}

Firstly, V_j that anonymously obtained signed token T_{tj} generates its vote v_j, secret random number r_j and its encrypted form r_j^* to ask election authorities to repeatedly encrypt pair $\{v_j r_j, r_j^*\}$ to $\{v_j r_j, r_j^*\}^*$, and attaches x_{xj}^* and C_{Cj}^*, encrypted unknown random number x_{xj} and encrypted confirmation number C_{Cj} given by the authorities, to form encrypted triple $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}^*$. Then, after V_j's verification and approval of the correct encryption of $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}$ in voting panel, authorities repeatedly sign on $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}^*$ and decrypt it to 2 signed forms of $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}$ while shuffling it with other votes, and finally disclose it in tallying panel. Therefore, V_j can conceal v_j from others. Because r_j is the secret of V_j, only V_j can extract v_j from $\{v_j r_j, r_j^*\}$. Moreover, anyone including V_j and authorities cannot know the correspondence between V_j and publicly disclosed v_j in tallying panel *i.e.* any single entity cannot decrypt $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}^*$ without conspiring with all other authorities. Here in the vote approval process, V_j verifies whether C_{Cj}^* is registered or not also while examining publicly disclosed encrypted *CNs*.

In the above process, V_j can show its qualification for the election without disclosing its identity by the 1st signatures of authorities on T_{tj}, because forging the signatures on T_{tj} is impossible. V_j can approve its submitting v_j also anonymously by the 2nd signatures on T_{tj}. Even after the 1st signatures are disclosed, the 2nd signatures cannot be generated illegitimately because different signing keys are used, therefore anyone can convince itself that $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}^*$ approved by the 2nd signatures is legitimate, in other words, if illegitimate votes are detected, they had been generated after the approvals, or voters cannot complain even if their privacies are revealed in processes for finding liable entities for inconsistent votes.

Here uniqueness of registered confirmation number C_{Cj} assigned to V_j and the signature pair on it enable any entity to verify the validity of finally disclosed $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}$, *i.e.* when $\{v_j r_j x_{xj} C_{Cj}, r_j x_{xj}, C_{Cj}\}$ includes registered C_{Cj} and has the valid signatures of multiple authorities, anyone can convince itself that v_j is the valid one. Because no one knows the signing keys of all authorities, the only ways to illegitimately generate a vote with consistent signatures are to copy it from other vote, or to disrupt a vote. However about the copying, both the original and the copy have the same *CN*, and this contradicts the uniqueness of *CNs*. Also anyone cannot disrupt signature pairs, so that both signatures reveal same values. Moreover *CNs* ensure that tallying panel includes only and all approved votes. Namely, firstly anyone can easily check that C_{Cj}^* in the voting panel is the registered one, secondly, the signature pair on C_{Cj} disclosed in tallying panel ensures that C_{Cj} is the correct decryption of *CNs* disclosed in voting panel, and thirdly, the numbers of *CNs* disclosed in voting and tallying panels ensure that all *CNs* in voting panel are decrypted. Of course it is possible for authorities to encrypt C_{Cj} to C_{Cj}^* dishonestly at the time when they generate *CNs*. However, dishonest encryptions are finally detected as duplicated *CNs* or disrupted *CNs* as mentioned before, and also liable authorities can be detected easily by forcing authorities to repeatedly encrypt inconsistent and unused *CNs* again.

In the above, if V_j knows the decrypted values of r_j or C_{Cj}, coercers can know V_j's vote by asking them and finding the vote in tallying panel attached by r_j or C_{Cj}. In order to protect voters from these threats, r_j is multiplied by x_{xj}, and x_{xj} and C_{Cj} are repeatedly encrypted to x_{xj}^* and C_{Cj}^*, so that no one including voters and authorities can know the corresponding x_{xj} and C_{Cj}.

3.1 Confirmation Numbers (*CNs*)

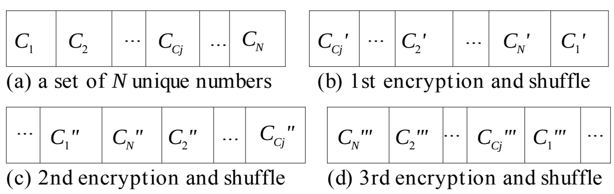

Fig. 2 Encryption steps of confirmation numbers

CNs attached to votes are encrypted repeatedly, so that no one can know their decrypted forms, and anyone including voters themselves cannot link voters to their votes attached by *CNs*. Firstly N unique numbers C_1, C_2, ---, C_N (N is the number of voters) are generated as shown in Fig. 2 (a). Then P (at least 2) mutually independent authorities TM_1, ---, TM_P repeatedly perform encryptions and shuffles of all *CNs* by using their encryption keys, *i.e.* TM_1 encrypts C_{Cj} to C_{Cj}' to be placed in random positions as shown in Fig. 2 (b). Then TM_2, TM_3, --- execute the same operations repeatedly, *i.e.* C_{Cj}' is further converted to C_{Cj}'', C_{Cj}''', --- as shown in Fig. 2 (c) and (d). Therefore, no entity can identify the link between C_{Cj} and its encrypted form C_{Cj}^* unless all *TMs* conspire *i.e.* no one including V_j itself can identify V_j from C_{Cj}.

Here $C_{Cj}' = E(K_1, C_{Cj})$, $C_{Cj}'' = E(K_2, C_{Cj}')$, $C_{Cj}''' = E(K_3, C_{Cj}'')$, ---, provided that x is encrypted to $E(K_i, x)$ by the encryption key K_i of TM_i. In the following repeatedly encrypted form C_{Cj}^* is denoted as $E(K_*, C_{Cj})$, *i.e.* $E(K_*, C_{Cj}) = E(K_P, E(K_{P\text{-}1}, \text{---}\ E(K_1, C_{Cj})$ ---)). This multiple encryption is carried out based on the probabilistic and commutative re-encryption scheme described in Sec 3.4.

3.2 Signature Pairs on Encrypted Votes

To protect repeatedly encrypted vote $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*$ from modifications during the process where it is decrypted to $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}$ while being shuffled with other votes, multiple authorities TM_1, ---, TM_P repeatedly sign on $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*$ by their signing keys $\{M_1, M_2, \text{---}, M_P\}$. In the followings $S(M_*, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*) = S(M_P, S(M_{P\text{-}1}, \text{---}, S(M_1, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*)$ ---)) represents repeatedly signed form of $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*$, where $S(M_i, x)$ is the signature of TM_i on x generated by its signing key M_i. Then, when encrypted signed form $S(M_*, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*)$ is successfully decrypted to signed form $S(M_*, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\})$, anyone can convince itself that *TMs* had honestly decrypted $S(M_*, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*)$. However, this scheme is effective only when all voters put meaningful votes. When decryption result is meaningless, entities cannot determine whether *TMs* are dishonest or $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}$ is meaningless from the beginning. A signature pair on $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*$ solves this problem. When each TM_i signs on $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*$ by its 2 different signing keys $M_{(1)i}$ and $M_{(2)i}$, it is impossible for any entity to consistently generate 2 different signed forms $S(M_{(1)*}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}) = S(M_{(1)P}, S(M_{(1)P\text{-}1}, \text{---}, S(M_{(1)1}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\})$ ---)) and $S(M_{(2)*}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}) = S(M_{(2)P}, S(M_{(2)P\text{-}1}, \text{---}, S(M_{(2)1}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\})$ ---)) in unauthorized ways because each TM_i knows only its signing keys. Namely, anyone can convince itself that *TMs* had decrypted $S(M_{(1)*}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*)$ to $S(M_{(1)*}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\})$ honestly, when 2 forms $S(M_{(1)*}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\})$ and $S(M_{(2)*}, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\})$ reveal same $\{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}$. These signatures are also generated based on the probabilistic and commutative re-encryption scheme.

3.3 Signature Pairs on Blinded Tokens

To show its eligibility anonymously, voter V_j encrypts its token T_{tj} to $E(a_j, T_{tj})$ by using its secret key a_j, and while confirming the identity of V_j by usual means *e.g.* through an *ID* and a password of V_j, authorities TM_1, ---, TM_P blindly sign on $E(a_j, T_{tj})$ to generate 2 different sets *i.e.* $\{S(X_{(1)1}, E(a_j, T_{tj})), \text{---}, S(X_{(1)P}, E(a_j, T_{tj}))\} = S(X_{(1)*}, E(a_j, T_{tj}))$ and $\{S(X_{(2)1}, E(a_j, T_{tj})), \text{---}, S(X_{(2)P}, E(a_j, T_{tj}))\} = S(X_{(2)*}, E(a_j, T_{tj}))$ by using their signing keys $\{X_{(1)1}, X_{(1)2}, \text{---}, X_{(1)P}\}$ and $\{X_{(2)1}, X_{(2)2}, \text{---}, X_{(2)P}\}$, and V_j decrypts them into 2 unblinded sets of signed tokens *i.e.* $\{S(X_{(1)1}, T_{tj}), \text{---}, S(X_{(1)P}, T_{tj})\} = S(X_{(1)*}, T_{tj})$ and $\{S(X_{(2)1}, T_{tj}), \text{---}, S(X_{(2)P}, T_{tj})\} = S(X_{(2)*}, T_{tj})$. Then, because *TMs* had signed without knowing T_{tj}, anyone except V_j cannot know V_j from $S(X_{(1)*}, T_{tj})$ and $S(X_{(2)*}, T_{tj})$.

3.4 Probabilistic and Commutative Re-encryptions

A multiple encryption and signing scheme for votes and *CNs* described in Secs. 3.1 and 3.2 can be implemented based on the probabilistic and commutative encryption algorithm with homomorphic property, proposed in (15). In the election, different voters may choose the same candidates, therefore the encryption function must be probabilistic; if not probabilistic, same candidates are encrypted into same forms, and a voter can know votes of other voters who had chosen the same candidate even they are encrypted. Also to ensure the authenticity of votes, the encryption and signing algorithms must be commutative. When they are not commutative, the signed form of encrypted vote $S(M_*, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\}^*)$ cannot be decrypted to $S(M_*, \{v_jr_jx_{xj}C_{Cj}, r_jx_{xj}, C_{Cj}\})$. In Sec. 3.1 *CNs* are encrypted without being mixed with random numbers. Because all *CNs* are unique and all of their encrypted forms are different, probabilistic encryption is not necessary.

To use re-encryption scheme proposed in (15), each authority TM_i defines its encryption and decryption key pairs $\{K_i, F_i\}$ and $\{H_i, G_i\}$, while selecting 2 large appropriate integers p_1 and p_2, where for any integer u and w, $u^{KiFi} \pmod{p_1} = u \pmod{p_1}$ and $w^{HiGi} \pmod{p_2} = w \pmod{p_2}$. Then TM_i encrypts x to $E(\{K_i, H_i\}, \{x, r\}) = \{E(K_i, xr) = (xr)^{Ki}, E(H_i, r) = r^{Hi}\}$ while mixing x with random secret number r as shown in Fig. 3 *i.e.* the encrypted form consists of a pair of data part $E(K_i, xr)$ and a randomixation part $E(H_i, r)$. Here, the key pairs are kept as TM_i's secrets, in order to enable each TM_i to securely use its key pairs under the environment where multiple authorities share the same modulo arithmetic. When key K_i is disclosed, it is easy for TM_j to calculate TM_i's decryption key F_i from the relation $K_iF_i \pmod{\varphi(p_1)} = K_jF_j \pmod{\varphi(p_1)}$ where $\varphi(p_1) = p_1 - 1$ when p_1 is a prime number, for example. In the following u^{Ki}, w^{Hi}, $u^{K1\text{---}KP}$ and $w^{H1\text{---}HP}$ are denoted as $E(K_i, u)$, $E(H_i, w)$, $E(K_*, u)$ and $E(H_*, w)$ respectively.

data part	randomization part
$E(K_i, xr) = (xr)^{Ki} \pmod{p_1}$	$E(H_i, r) = r^{Hi} \pmod{p_2}$

Fig. 3 $E(\{K_i, H_i\}, \{x, r\})$ encrypted form of x

Based on the above scheme voter V_j encrypts its vote v_j to $E(\{K_i, H_i\}, \{v_j, r_j\})$, while generating its secret random number r_j and asking TM_1, ---, TM_P to encrypt v_jr_j and r_j^{Lj}; where $\{L_j, Z_j\}$ is a secret encryption and decryption key pair of V_j, and *TMs* cannot calculate v_j from v_jr_j and r_j^{Lj}, because r_j is secret of V_j and the calculation of r_j from r_j^{Lj} is a discrete logarithm problem. Then TM_1, ---, TM_P repeatedly encrypt the pair $\{v_jr_j, r_j^{Lj}\}$, *i.e.* calculate $E(K_*, v_jr_j)$ and $E(H_*, r_j^{Lj})$ by their encryption keys K_1, ---, K_P and H_1, ---, H_P, and finally V_j calculates $E(H_*, r_j^{Lj})^{Zj} = E(H_*, r_j)$ to construct its repeatedly encrypted vote as $E(\{K_*, H_*\}, \{v_j, r_j\}) = \{E(K_*, v_jr_j), E(H_*, r_j)\}$. $E(\{K_*, H_*\}, \{v_j, r_j\})$ can be decrypted into v_j by calculating $E(K_*, v_jr_j)^{F1\text{---}FP} = (v_jr_j)^{(K1\text{---}KP)(F1\text{---}FP)} = v_jr_j$ and $E(H_*, r_j)^{(G1\text{---}GP)} = r_j^{(H1\text{---}HP)(G1\text{---}GP)} = r_j$ by decryption keys F_1, ---, F_P and G_1, ---, G_P, and by dividing v_jr_j by r_j.

For the confirmation of correct encryptions of *TMs*, V_j asks TM_1, ---, TM_P to decrypt $E(K_*, (v_jr_j)^{Aj})$ and $E(H_*, r_j^{Bj})$, where $\{A_j, B_j\}$ are secret random numbers of V_j. Here, TM_1, ---, TM_P cannot decrypt $E(K_*, (v_jr_j)^{Aj})$ and $E(H_*, r_j^{Bj})$ into $(v_jr_j)^{Aj}$ and r_j^{Bj} when they calculate $E(K_*, v_jr_j)$ and $E(H_*, r_j^{Lj})$ dishonestly, because they do not know A_j, B_j, v_jr_j or r_j. Therefore although K_i and H_i of each TM_i are secret, V_j can confirm the correctness of encryptions as same as it is using public keys. It is apparent that this encryption scheme is probabilistic and commutative. Fortunately, it is also homomorphic, *e.g.* $E(K_*, x_1)E(K_*, x_2) = x_1^{K1\text{---}KP}x_2^{K1\text{---}KP} = E(K_*,$

x_1x_2) and $E(H_*, y_1)E(H_*, y_2) = y_1^{H1---HP}y_2^{H1---HP} = E(H_*, y_1y_2)$.

In the above, as V_j knows r_j, coercers can use this r_j to identify V_j's vote. To disable vote identification, V_j also multiplies its vote v_j by random number x_{xj} that is not known to anyone, where $x_{xj} = x_{xj1}x_{xj2}---x_{xjP}$. Namely, each TM_i generates its secret random number x_{xji}, and encrypts x_{xji} by its encryption keys K_i and H_i *i.e.* calculates $\{E(K_i, x_{xji}), E(H_i, x_{xji})\}$ and asks other authorities to calculate $\{E(K_*, x_{xji}), E(H_*, x_{xji})\}$. Then by using the homomorphic property, $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$ is generated by multiplying P different $\{E(K_*, x_{xji}), E(H_*, x_{xji})\}$. Because each TM_i knows only x_{xji}, no one can know the decrypted form of $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$. For the confirmation of correct encryptions of x_{xj}, V_j calculates $\{E(K_*, x_{xji}), E(H_*, x_{xji})^{Bj}\}$ to ask TM_1, ---, TM_P to decrypt them *i.e.* to calculate $E(K_*, x_{xji})^{F1---FP} = x_{xji}$ and $E(H_*, x_{xji})^{BjG1---GP} = x_{xji}^{Bj}$, for randomly selected i, and V_j checks the consistency between x_{xji} and x_{xji}^{Bj}. When $E(K_*, x_{xji})$ and $E(H_*, x_{xji})$ are calculated dishonestly, TM_1, ---, TM_P cannot decrypt $E(K_*, x_{xji})$ and $E(H_*, x_{xji})^{Bj}$ into x_{xji} and x_{xji}^{Bj} because they do not know B_j or x_{xji}. Because (P-1) remaining x_{xji}s are still unknown to anyone except TM_i, no one can know the decrypted form of x_{xj} unless all *TMs* conspire, and V_j can calculate $E(\{K_*, H_*\}, \{v_j, r_jx_{xj}\}) = \{E(K_*, v_jr_jx_{xj}), E(H_*, r_jx_{xj})\}$ while making r_jx_{xj} unknown to anyone. Here, to maintain the equality of 2 forms of x_{xj}, *i.e.* x_{xj} (mod p_1) and x_{xj} (mod p_2), each x_{xji} must be defined so that x_{xj} is less than p_1 and p_2.

Repeatedly signing mechanisms on re-encrypted forms can be implemented in the same way. However, each TM_i can calculate signing keys $\{M_{(1)k}, M_{(2)k}\}$ of other TM_k when its verification keys $\{U_{(1)k}, U_{(2)k}\}$ are disclosed from the relation $M_{(1)k} = M_{(1)i}U_{(1)i}/U_{(1)k}$ and $M_{(2)k} = M_{(2)i}U_{(2)i}/U_{(2)k}$. Therefore, verification keys must be disclosed only after all votes are decrypted. In the proposed scheme all votes are put in bulletin board (*BB*), where a *BB* is a public broadcast channel with memories, and information sent to a *BB* is readable by anyone and at anytime. Then, no one can forge signatures on votes in *BB* even the signing keys are revealed *i.e.* before the disclosure of verification keys, no one knows all signing keys; and after the disclosure of verification keys, votes are already decrypted and cannot be modified. Here, V_j can verify the correctness of signatures without knowing the verification keys in the same way as in the encryption processes.

Probabilistic and commutative re-encryption schemes also can be constructed based on ElGamal or threshold ElGamal encryption. However, to identify dishonest authorities without disclosing privacies of voters, ElGamal based schemes require complicated ZKP processes.

In the remainder, $E(\{K_*, H_*\}, \{v_j, r_jx_{xj}\})$ is denoted as $E(\{K_*, H_*\}, v_j)$ to make notations comprehensive.

4. Configuration of the Voting Scheme

Entities involved in the scheme are N voters V_j (j = 1, ---, N), Voting manager *VM*, P (at least 2) mutually independent Tallying managers TM_i (i = 1, ---, P), Disruption detection manager *DM* and 6 public *BB*s that maintain authorized communication transcripts *i.e.* *VoterList*, *TokenList*, *ConfNoList*, *ActiveTokenList*, *VotingPanel* and *TallyingPanel*. Figure 4 depicts the configurations of individual *BB*s. By putting relevant information on several *BB*s, interactions among the entities at every stage of the election become publicly verifiable. In the followings V_j is the j-th voter, v_j is the vote of V_j, and C_{Cj}, T_{tj} and x_{xj} are the *CN*, token and unknown random number assigned to V_j. ID_j and P/W_j are the identifier and password of V_j. The roles of each entity and the *BBs* are as follows:

Voter V_j: Each V_j generates its encrypted vote $E(\{K_*, H_*\}, v_jC_{Cj})$ while combining its vote v_j with its assigned encrypted C_{Cj} *i.e.* $E(K_*, C_{Cj})$, and puts and approves it in *VotingPanel*. It has its own identifier ID_j and password P/W_j for proving its eligibility, and 2 secret encryption and decryption key pairs $\{a_j, g_j\}$ and $\{L_j, Z_j\}$. $\{a_j, g_j\}$ is used to acquire 2 different forms of signatures of all *TMs* on its token T_{tj} blindly *i.e.* $S(X_{(1)*}, T_{tj})$ and $S(X_{(2)*}, T_{tj})$, and key pair $\{L_j, Z_j\}$ is used to ask *TMs* to encrypt vote v_j without disclosing v_j itself.

Voting manager *VM*: *VM* is responsible for authenticating voters, for assigning *CNs* to voters, and for putting encrypted votes of voters in *VotingPanel*. It also puts other data about voters and votes in *VoterList*, *TokenList*, *ConfNoList* and *ActiveTokenList*. *VM* can be constructed by multiple independent entities to distribute its responsibility if necessary.

Tallying managers *TMs*: Mutually independent P ($P \geq 2$) *TMs* sign on blinded tokens, perform encryptions and shuffles of *CNs* and votes, repeatedly sign on encrypted votes and encrypted *CNs* in *VotingPanel*, and perform decryptions and shuffles of votes in *VotingPanel* to compute the tally and to put results on *TallyingPanel*. For encryption and decryption of votes and *CNs*, each TM_i has 2 encryption and decryption key pairs $\{K_i, F_i\}$ and $\{H_i, G_i\}$. Also to sign on blinded token $E(a_j, T_{tj})$, TM_i has 2 signing and verification key pairs *i.e.* $\{X_{(1)i}, B_{(1)i}\}$ and $\{X_{(2)i}, B_{(2)i}\}$, and to repeatedly sign on encrypted votes and encrypted *CNs* in 2 different forms, it has 4 secret signing and verification key pairs $\{\{M_{(1)i}, U_{(1)i}\}, \{Q_{(1)i}, W_{(1)i}\}\}$ and $\{\{M_{(2)i}, U_{(2)i}\}, \{Q_{(2)i}, W_{(2)i}\}\}$.

Disruption detection manager *DM*: *DM* detects inconsistent votes in *TallyingPanel*, and when inconsistencies are detected it identifies the entities that cause the inconsistencies.

ID	token
ID_1	$E(a_1,T_{t1})$
...	
ID_j	$E(a_j,T_{tj})$
...	
ID_N	$E(a_N,T_{tN})$

(a) VoterList

token	flag
T_1	unused
...	
T_{tj}	used
...	
T_N	used

(b) TokenList

CN	random number
$E(K_*,C_k)$	$E(K_*,x_k),E(H_*,x_k)$
...	
$E(K_*,C_{Cj})$	$E(K_*,x_{xj}),E(H_*,x_{xj})$
...	
$E(K_*,C_u)$	$E(K_*,x_u),E(H_*,x_u)$

(c) ConfNoList

token	*CN*
$S(X_{(1)*},T_1)$	$E(K_*,C_y)$
...	
$S(X_{(1)*},T_{tj})$	$E(K_*,C_{Cj})$
...	
$S(X_{(1)*},T_N)$	$E(K_*,C_s)$

(d) ActiveTokenList

vote	approval
$S(\{M_{(1)*},Q_{(1)*}\},E(\{K_*,H_*\},v_qC_q)),S(M_{(1)*},E(K_*,C_q)),S(\{M_{(2)*},Q_{(2)*}\},E(\{K_*,H_*\},v_qC_q))$	$S(X_{(2)*},T_1)$
...	
$S(\{M_{(1)*},Q_{(1)*}\},E(\{K_*,H_*\},v_jC_{Cj})),S(M_{(1)*},E(K_*,C_{Cj})),S(\{M_{(2)*},Q_{(2)*}\},E(\{K_*,H_*\},v_jC_{Cj}))$	$S(X_{(2)*},T_{tj})$
...	
$S(\{M_{(1)*},Q_{(1)*}\},E(\{K_*,H_*\},v_hC_h)),S(M_{(1)*},E(K_*,C_h)), S(\{M_{(2)*},Q_{(2)*}\},E(\{K_*,H_*\},v_hC_h))$	$S(X_{(2)*},T_N)$

(e) VotingPanel

vote	*CN*
$S(\{M_{(1)*},Q_{(1)*}\},v_gC_n)$ $S(\{M_{(2)*},Q_{(2)*}\},v_gC_n)$	$S(M_{(1)*},C_n)$
...	
$S(\{M_{(1)*},Q_{(1)*}\},v_jC_{Cj})$ $S(\{M_{(2)*},Q_{(2)*}\},v_jC_{Cj})$	$S(M_{(1)*},C_{Cj})$
...	
$S(\{M_{(1)*},Q_{(1)*}\},v_eC_r)$ $S(\{M_{(2)*},Q_{(2)*}\},v_eC_r)$	$S(M_{(1)*},C_r)$

(f) TallyingPanel

Fig. 4 Configurations of bulletin boards.

VoterList: *VoterList* consists of *ID* and token parts. *ID* part maintains *IDs* of eligible voters, and *VM* puts $E(a_j, T_{tj})$, a token encrypted by voter V_j, at the token part corresponding to V_j's *ID* when V_j shows it to obtain *TMs*' signatures on it as shown in Fig. 4 (a). Therefore anyone can know voters who had acquired signatures of *TMs* on their tokens. However no one except voters themselves can know tokens on which *TMs* had signed.

TokenList: *TokenList* consists of the token and flag parts, and enables voters to acquire tokens without collision. The token part maintains tokens *i.e.* unique numbers prepared by *VM*. When V_j picks unused token T_{tj} from *TokenList* anonymously, *VM* makes the corresponding flag part used as shown in Fig. 4 (b).

ConfNoList: It consists of *CN* and random number parts, and for N voters, N different *CNs* and unknown random numbers are generated and each *CN* and random number pair $\{C_{Cj}, x_{xj}\}$ is encrypted to $E(K_*, C_{Cj})$ and $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$ to be posted here at random by *VM* as shown in Fig. 4 (c).

ActiveTokenList: It consists of the token and the *CN* parts, and enables anyone to know anonymous V_j who had been assigned C_{Cj} in its encrypted form. The t_j-th position of the token part maintains the 1st signed form of the t_j-th token T_{tj} *i.e.* $S(X_{(1)*}, T_{tj})$. The corresponding *CN* part maintains encrypted C_{Cj}, *CN* assigned to the voter who obtains T_{tj} *i.e.* $E(K_*, C_{Cj})$ as shown in Fig. 4 (d). Here, by comparing the items in *ActiveTokenList*, *ConfNoList* and *VoterList*, anyone can verify that only voters with their signed tokens acquire *CNs*, and *VM* is not misusing or adding any extra *CN* illegally.

VotingPanel: *VotingPanel* consists of the vote and the approval parts, and enables anyone to know encrypted votes approved by their voters. The vote part corresponding to the t_j-th position maintains 2 different signed forms of encrypted vote of the voter to whom the t_j-th token T_{tj} is assigned. Namely it maintains anonymous V_j's encrypted vote v_j repeatedly signed by 2 secret signing key pairs $\{M_{(1)i}, Q_{(1)i}\}$ and $\{M_{(2)i}, Q_{(2)i}\}$ of all TM_i and encrypted C_{Cj} in the 1st signed form *i.e.* $S(\{M_{(1)*}, Q_{(1)*}\}, E(\{K_*, H_*\}, v_jC_{Cj}))$, $S(\{M_{(2)*}, Q_{(2)*}\}, E(\{K_*, H_*\}, v_jC_{Cj}))$ and $S(M_{(1)*}, E(K_*, C_{Cj}))$, and the approval part maintains the 2nd signed form of T_{tj} *i.e.* $S(X_{(2)*}, T_{tj})$ as shown in Fig. 4 (e). In the above, $S(\{M_{(h)*}, Q_{(h)*}\}, E(\{K_*, H_*\}, x))$ represents pair $\{S(M_{(h)*}, E(K_*, xr)), S(Q_{(h)*}, E(H_*, r))\}$ for h = 1 and 2, provided that r is a secret random number used for encrypting x to $E(\{K_*, H_*\}, x))$.

TallyingPanel: *TallyingPanel* consists of the vote part and the *CN* part and enables anyone to know the election results. It maintains decrypted data of *VotingPanel* *i.e.* the vote part maintains $\{S(\{M_{(1)*}, Q_{(1)*}\}, v_jC_{Cj}), S(\{M_{(2)*}, Q_{(2)*}\}, v_jC_{Cj})\}$ and the *CN* part maintains $S(M_{(1)*}, C_{Cj})$ as shown in Fig. 4 (f). Based on *CNs*, anyone can verify that only and all votes from eligible voters are included in *TallyingPanel*. However, because votes on *VotingPanel* are decrypted while being shuffled to be put on *TallyingPanel*, no one can identify linkages between voters and their votes.

4.1 Overview of the Scheme

The proposed voting scheme consists of 5 stages as follows. The relationships and the data flows among the entities are shown in Fig. 5.

Token acquisition: Anonymously authenticated voter V_j picks unique token T_{tj} while maintaining tokens collision free.

Registration: Voter V_j whose eligibility is checked by its identifier ID_j and password P/W_j obtains 2 kinds of blind signatures of Tallying managers on T_{tj} *i.e.* $S(X_{(1)*}, E(a_j, T_{tj}))$ and $S(X_{(2)*}, E(a_j, T_{tj}))$. Therefore later on V_j can prove its eligibility by showing decrypted signatures $S(X_{(1)*}, T_{tj})$ and $S(X_{(2)*}, T_{tj})$, without disclosing its identity. Here a_j is a secret encryption key of V_j.

Voting: This stage consists of 2 sub-stages.

***CN* assignment:** V_j proves its eligibility by showing $S(X_{(1)*}, T_{tj})$ and obtains repeatedly encrypted confirmation number C_{Cj} *i.e.* $E(K_*, C_{Cj})$ and encrypted unknown random number $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$ from Voting manager *VM*. Also, V_j verifies the correctness of encryption of $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$.

Vote submission: V_j asks Tallying managers to repeatedly encrypt its vote v_j to $E(\{K_*, H_*\}, v_j)$ while randomizing it by secret numbers r_j and x_{xj} and verifies its correctness. Then V_j calculates $\{E(\{K_*, H_*\}, v_jC_{Cj}), E(K_*, C_{Cj})\}$ while combining $E(\{K_*, H_*\}, v_j)$ with its assigned $E(K_*, C_{Cj})$ and submits it as its vote, and TM_1, ---, TM_P sign on them by the 1st form of their signatures *i.e.* calculate $S(\{M_{(1)*}, Q_{(1)*}\}, E(\{K_*, H_*\}, v_jC_{Cj}))$ and $S(M_{(1)*}, E(K_*, C_{Cj}))$. After verifying its vote on *VotingPanel*, V_j approves the registration of its vote by $S(X_{(2)*}, T_{tj})$, and finally TM_1, ---, TM_P sign on $E(\{K_*, H_*\}, v_jC_{Cj})$ by the 2nd form of their signatures *i.e.* calculate $S(\{M_{(2)*}, Q_{(2)*}\}, E(\{K_*, H_*\}, v_jC_{Cj}))$.

Tallying: Multiple decryptions and shuffles of votes in *VotingPanel* by Tallying managers compute the election results and they are disclosed on *TallyingPanel* while concealing links between votes in *VotingPanel* and *TallyingPanel*. However *CNs* attached to individual votes ensure that all and only eligible votes are counted.

Disruption detection: If inconsistency is found for any disclosed vote, the responsible entity is identified while maintaining the privacy of voters.

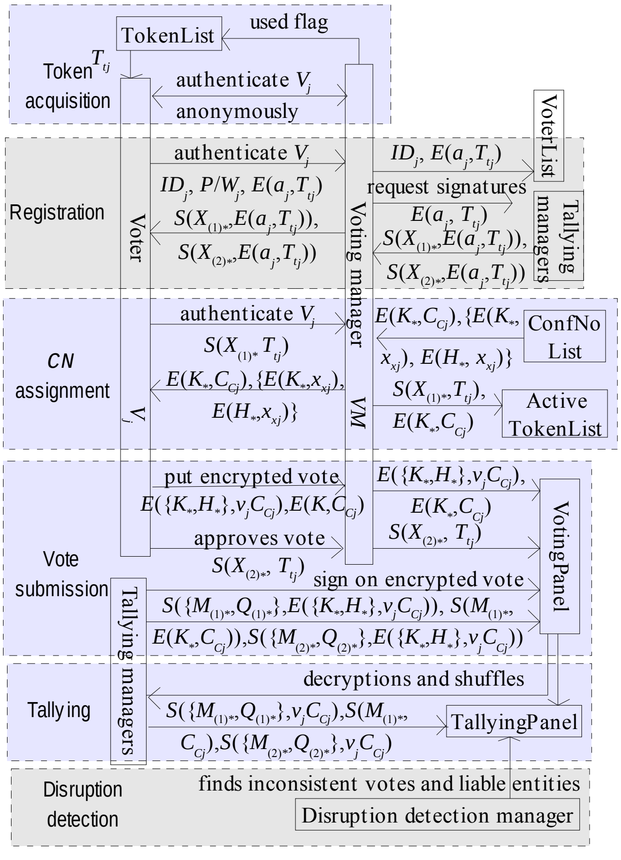

Fig. 5 Relationships and data flow among entities

5. Individual Stages of the Scheme

Individual stages of the scheme proceed as follows:

5.1 Token Acquisition Stage

An objective of this stage is to assign voter V_j a token T_{tj} which is unique in the system while maintaining anonymity of V_j. To achieve this objective, anonymously authenticated V_j picks T_{tj} from *TokenList*. Here, more than N different numbers are generated as tokens in advance and they are put in *TokenList* to be picked by voters; where N is the number of eligible voters. To enforce V_j to pick T_{tj} from *TokenList*, every T_{tj} has the signature of

VM (this signature is different from $S(X_{(1)*}, T_{tj})$ and $S(X_{(2)*}, T_{tj})$, and ensures that T_{tj} is picked from *TokenList*). However tokens in *TokenList* are open to the public only in non-signed forms to disable entities to pick them in unauthorized ways. Theoretically, V_j authentication is not necessary. But by protecting T_{tj} from being picked by unauthorized entities, it becomes possible to make *TokenList* as small as possible *i.e*. unauthorized entities cannot request tokens. V_j and *VM* interact as follows:

1. *VM* authenticates eligible V_j anonymously *e.g*. through anonymous authentication mechanism [10].

2. Authenticated V_j picks unused token T_{tj} from *TokenList*, and *VM* signs on T_{tj} (this signature is omitted in the following notations).

3. In order to avoid collision, *VM* makes T_{tj} in *TokenList* used as shown in Fig. 4 (b).

Security problems of this stage are solved as below:

- *Voters may get multiple tokens:* Because only tokens with signatures of Tallying managers, which are given at the registration stage while confirming the eligibility of individual voters, are effective, voters can use only single tokens even they get multiple tokens.
- *Voters may not get tokens:* As multiple tokens cause no inconvenience, V_j can request T_{tj} assignment repeatedly.

5.2 Registration Stage

Objectives of this stage are: (1) to let Tallying managers sign on token T_{tj} that is shown by eligible voter V_j without knowing T_{tj} itself [14], so that V_j can show its eligibility anonymously by it at the later stages, and (2) to let all entities know V_j who is assigned signed T_{tj}. To make voters that obtain signed tokens publicly visible, *VM* maintains *VoterList*, as shown in Fig. 4 (a), but at this stage V_j shows T_{tj} in its blinded form, *i.e*. *VM* puts $E(a_j, T_{tj})$ in *VoterList*. Therefore anyone can monitor V_j who is registered, however, only V_j knows its token T_{tj}. As a consequence, V_j can abstain from vote submission without being noticed even it is registered in *VoterList* for example. The interactions between V_j and *VM* in this stage are as follows:

1. V_j encrypts T_{tj} by using its secret encryption key a_j *i.e*. V_j calculates $E(a_j, T_{tj})$.

2. V_j shows its ID_j, P/W_j and its blinded token $E(a_j, T_{tj})$ to *VM*.

3. *VM* authenticates V_j and post $E(a_j, T_{tj})$ in *VoterList* so that anyone can know that V_j has been registered. *VM* also sends $E(a_j, T_{tj})$ to Tallying managers for their signatures.

4. Mutually independent TM_1, ---, TM_P sign on $E(a_j, T_{tj})$ with their 2 different signatures *i.e*. calculate $S(X_{(1)*}, E(a_j, T_{tj}))$ and $S(X_{(2)*}, E(a_j, T_{tj}))$ and sends them to *VM* to be sent to V_j.

5. V_j checks the validity of signatures on T_{tj}.

Here the 3rd step ensures that ineligible voters cannot obtain signed tokens and even eligible voters cannot get multiple signed tokens. Also the 4th step ensures that anyone cannot forge signed tokens unless all *TMs* conspire. Security problems of this stage are as follows:

- *Multiple entities request signatures on T_{tj} picked by V_j:* By this threat, V_j's vote will be rejected. There are 2 possibilities, the 1st one occurs when signed T_{tj} is stolen, however V_j is responsible for that. The other possibility is a case where *VM* uses signed T_{tj}. This possibility can be excluded, if necessary, by duplicating *VM*, *i.e*. no entity can obtain signatures of all *TMs* on T_{tj} in unauthorized ways unless all *VMs* conspire.
- *Voters cannot get correct signed tokens:* V_j can prove *VM*'s dishonesty by showing $E(a_j, T_{tj})$ and the incorrect signed token.

5.3 Voting Stage

This stage consists of 2 sub-stages, which are: i) *CN* assignment and ii) Vote submission.

5.3.1 *CN* Assignment Sub-Stage

In this sub-stage: (1) Voting manager *VM* authenticates voter V_j anonymously by signed token $S(X_{(1)*}, T_{tj})$, and (2) V_j receives encrypted C_{Cj} *i.e.* $E(K_*, C_{Cj})$ and encrypted unknown random number $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$. While *VM* sends $E(K_*, C_{Cj})$ to V_j, it also discloses $E(K_*, C_{Cj})$ and $S(X_{(1)*}, T_{tj})$ in *ActiveTokenList*. Here as shown in Sec 3.1, anyone even V_j itself cannot identify the correspondence between original C_{Cj} and $E(K_*, C_{Cj})$, and hence between C_{Cj} and V_j. However, because *CNs* are unique and registered, and no one can forge signatures of all Tallying managers on them, any entity can confirm the accuracy of votes by *CNs* disclosed in *TallyingPanel*. The interactions between V_j and *VM* in this sub-stage are as follows:

1. V_j submits $S(X_{(1)*}, T_{tj})$ to *VM*, and *VM* checks the validity of $S(X_{(1)*}, T_{tj})$. Here *VM* can verify the authenticity of V_j by checking only the signatures on T_{tj} that is not used repeatedly.

2. *VM* sends $E(K_*, C_{Cj})$ and $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$ to V_j, and V_j verifies the correctness of encryption of $\{E(K_*, x_{xj}), E(H_*, x_{xj})\}$.

3. *VM* also puts $S(X_{(1)*}, T_{tj})$ and $E(K_*, C_{Cj})$ in *ActiveTokenList* as shown in Fig. 4 (d).

Security problems of this sub-stage are as follows:

- *VM may put signed tokens in ActiveTokenList before voters: VM* knows neither of V_j's secret key nor the signing keys of all *TMs*, therefore it cannot generate $S(X_{(1)*}, T_{tj})$ from $S(X_{(1)*}, E(a_j, T_{tj}))$ or T_{tj} to put it before V_j.
- *VM may not put signed token in ActiveTokenList: VM* cannot deny putting of $S(X_{(1)*}, T_{tj})$ on *ActiveTokenList* because $S(X_{(1)*}, T_{tj})$ has the signatures of all *TMs*.
- *VM* may not give C_{Cj}, or give incorrect C_{Cj} to V_j*:* As $S(X_{(1)*}, T_{tj})$ is open to the public, *VM* cannot deny giving of C_{Cj}. Also as $E(K_*, C_{Cj})$ is open on *ConfNoList*, *VM* cannot give non-registered C_{Cj}. Although it is possible that *TMs* encrypt C_{Cj} incorrectly, this dishonesty and the responsible entities are detected at the disruption detection stage, therefore *TMs* cannot encrypt *CNs* incorrectly.

5.3.2 Vote Submission Sub-stage

In this sub-stage: (1) anonymous voter V_j submits its verifiable secret vote, (2) Tallying managers TM_1, ---, TM_P repeatedly sign on the vote, (3) after confirming the successful registration of the vote on *VotingPanel*, V_j approves its vote by putting the 2nd signed form of T_{tj} *i.e.* $S(X_{(2)*}, T_{tj})$ in *VotingPanel* as shown in Fig. 4 (e), and (4) finally *TMs* repeatedly sign on the vote by the 2nd form of their signatures. Here, $E(K_*, v_jr_jx_{xj})$ and $E(H_*, r_jx_{xj})$ are computed as the product of $E(K_*, v_jr_j)$ and $E(K_*, x_{xj})$, and $E(H_*, r_j)$ and $E(H_*, x_{xj})$ respectively, and vote $E(\{K_*, H_*\}, v_jC_{Cj})$ is constructed as the product of $E(K_*, v_jr_jx_{xj})$ and $E(K_*, C_{Cj})$. As V_j asks *TMs* to encrypt v_jr_j instead of v_j while generating secret random number r_j, TM_1, ---, TM_P cannot know v_j. Also encrypted v_jr_j is further multiplied by encrypted x_{xj}, of which decrypted value is not known to anyone; therefore even V_j cannot identify its vote at the tallying stage. About the approval of votes, because no one except V_j knows $S(X_{(2)*}, T_{tj})$ even after $S(X_{(1)*}, T_{tj})$ had been disclosed, only V_j can approve its vote, consequently V_j cannot claim any dishonesty about its vote after its approval. Figure 6 depicts the steps of vote constructions; they proceed as follows:

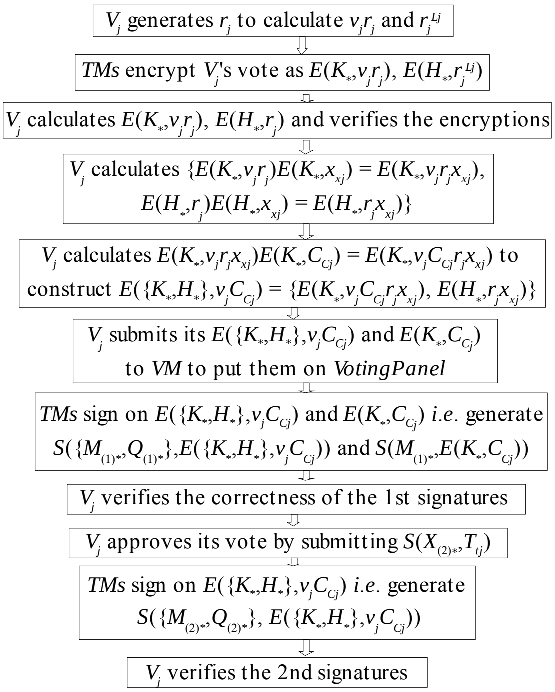

Fig. 6 Vote construction procedures

1. V_j generates its secret random number r_j to calculate v_jr_j and r_j^{Lj}, and asks TM_1, ---, TM_P to encrypt them into $E(K_*, v_jr_j)$ and $E(H_*, r_j^{Lj})$. By using these results, V_j calculates $E(\{K_*, H_*\}, v_j) = \{E(K_*, v_jr_j), E(H_*, r_j)\}$ as described in Sec 3.4.

2. V_j verifies the correctness of encryption of $E(\{K_*, H_*\}, v_j)$, and calculates $E(K_*, v_jr_j)E(K_*, x_{xj}) = E(K_*, v_jr_jx_{xj})$ and $E(H_*, r_j)E(H_*, x_{xj}) = E(H_*, r_jx_{xj})$. Then it multiplies $E(K_*, v_jr_jx_{xj})$ by its $E(K_*, C_{Cj})$, *i.e.* calculates $E(K_*, v_jr_jx_{xj})E(K_*, C_{Cj}) = E(K_*, v_jC_{Cj}r_jx_{xj})$, and constructs its vote as $E(\{K_*, H_*\}, v_jC_{Cj}) = \{E(K_*, v_jC_{Cj}r_jx_{xj}), E(H_*, r_jx_{xj})\}$.

3. V_j submits $E(\{K_*, H_*\}, v_jC_{Cj})$ and $E(K_*, C_{Cj})$ as its vote and puts them on the position corresponding to T_{tj} in *VotingPanel*.

4. TM_1, ---, TM_P repeatedly sign on $E(\{K_*, H_*\}, v_jC_{Cj})$ and $E(K_*, C_{Cj})$ in *VotingPanel* by the 1st form of their signatures *i.e.* calculate $S(\{M_{(1)*}, Q_{(1)*}\}, E(\{K_*, H_*\}, v_jC_{Cj}))$ and $S(M_{(1)*}, E(K_*, C_{Cj}))$.

5. After confirming the correctness of signatures on its vote in *VotingPanel*, V_j submits $S(X_{(2)*}, T_{tj})$ to *VM* as its approval.

6. TM_1, ---, TM_P repeatedly sign on $E(\{K_*, H_*\}, v_jC_{Cj})$ by the 2nd form of their signatures *i.e.* calculate $S(\{M_{(2)*}, Q_{(2)*}\}, E(\{K_*, H_*\}, v_jC_{Cj}))$. Finally V_j verifies the signatures.

For this sub-stage security problems are as follows:

- *Voters may submit invalid votes to disrupt the voting:* V_j cannot claim that its vote is disrupted even its vote is meaningless when disclosed C_{Cj} is valid and signatures *i.e.* $S(\{M_{(1)*}, Q_{(1)*}\}, v_jC_{Cj})$ and $S(\{M_{(2)*}, Q_{(2)*}\}, v_jC_{Cj})\}$ are consistent.
- *VM may not put vote or put incorrect vote on VotingPanel:* As $S(X_{(1)*}, T_{tj})$ is open to the public, V_j can repeatedly submit its vote before its approval, therefore *VM* cannot deny putting. If *VM* puts incorrect vote, V_j can disapprove it.
- *Someone may disrupt votes in VotingPanel:* As *VotingPanel* is open to the public, no one can modify or delete votes without being detected.

5.4 Tallying Stage

Objectives of this stage are to decrypt all encrypted votes in *VotingPanel* and to disclose the results on *TallyingPanel* while concealing links between voters and their votes. When the deadline of vote submission comes, mutually independent *TMs* repeatedly perform decryptions and shuffles of votes by using their secret decryption keys to post the results on *TallyingPanel*, as shown in Fig. 7. In the figure, 3 Tallying managers TM_2, TM_1 and TM_3 execute decryptions and shuffles. In this example, multiple decryptions are executed in the order different from encryptions.

For this stage security problems are as follows:

- *Tallying managers may change votes:* No one can generate 2 different forms of votes consistently unless all *TMs* conspire, and when votes are changed, responsible *TMs* are detected at the disruption detection stage based on this inconsistency. For example, although TM_i can forge $S(\{M_{(1)*}, Q_{(1)*}\}, v_kC_{Cj})$ from $S(\{M_{(1)*}, Q_{(1)*}\}, v_jC_{Cj})$, $S(M_{(1)*}, C_{Cj})$, $S(\{M_{(1)*}, Q_{(1)*}\}, v_kC_{Ck})$ and $S(M_{(1)*}, C_{Ck})$, and replace $S(\{M_{(1)*}, Q_{(1)*}\}, v_jC_{Cj})$ by it based on the homomorphic property, TM_i cannot forge $S(\{M_{(2)*}, Q_{(2)*}\}, v_kC_{Cj})$ consistently because it does not know $S(M_{(2)*}, C_{Cj})$.
- *TMs may add votes:* Anyone can detect the added votes by duplicated or by non registered *CNs*.
- *TMs may delete votes:* By this the numbers of votes on *VotingPanel* and *TallyingPanel* become different which is detectable by anyone.

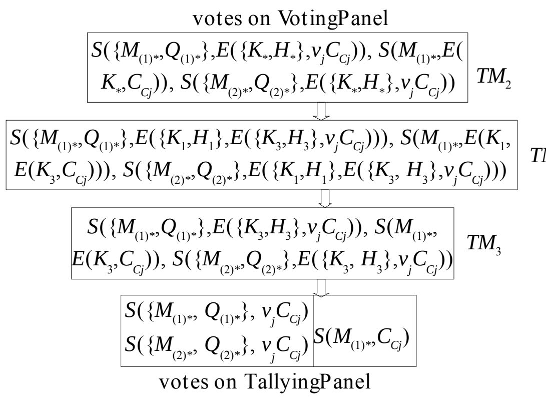

Fig. 7 Procedures in Tallying stage

5.5 Disruption Detection Stage

If any inconsistency is found in *TallyingPanel*, Disruption detection manager *DM* identifies liable entities. Figure 8 shows examples of consistent and inconsistent votes on *TallyingPanel*. The 1st vote (1st row) is consistent because 2 different forms of vote v_2C_{18} are same and also C_{18} is registered. The 2nd vote (2nd row) is not consistent because the candidates within the 2 signed forms are different. The 3rd vote (3rd row) is inconsistent because C_{-10} is not registered. The 4th and 5th votes (4th and 5th rows) are also inconsistent because of duplicated *CNs*.

DM identifies the liable entities as follows. When an inconsistent vote $\underline{v}$ is found, *DM* asks *TMs* to encrypt $\underline{v}$ again in the reverse order of the tallying stage, namely each TM_i encrypts $\underline{v}$ and discloses the result with its input vote in the tallying stage that matches with $\underline{v}$. When this matching chain fails, the dishonest TM_i is found. Here TM_i cannot encrypt votes dishonestly because anyone can check the correctness of its encryption in the same way as in Sec. 3.4. Also when $\underline{v}$ had been submitted in the authorized way, dishonest managers are identified before the chain reaches *VotingPanel*. Therefore privacies of voters are maintained.

$S(\{M_{(1)*}, Q_{(1)*}\}, v_2C_{18}), S(\{M_{(2)*}, Q_{(2)*}\}, v_2C_{18})$	$S(M_{(1)*}, C_{18})$	√
$S(\{M_{(1)*}, Q_{(1)*}\}, v_bC_2), S(\{M_{(2)*}, Q_{(2)*}\}, v_aC_2)$	$S(M_{(1)*}, C_2)$	x
$S(\{M_{(1)*}, Q_{(1)*}\}, v_hC_{-10}), S(\{M_{(2)*}, Q_{(2)*}\}, v_hC_{-10})$	$S(M_{(1)*}, C_{-10})$	x
$S(\{M_{(1)*}, Q_{(1)*}\}, v_bC_{25}), S(\{M_{(2)*}, Q_{(2)*}\}, v_bC_{25})$	$S(M_{(1)*}, C_{25})$	x
$S(\{M_{(1)*}, Q_{(1)*}\}, v_aC_{25}), S(\{M_{(2)*}, Q_{(2)*}\}, v_aC_{25})$	$S(M_{(1)*}, C_{25})$	x

"√" and "x" imply consistent and inconsistent votes, respectively

Fig. 8 Possible vote disruptions

6. Evaluation of the Scheme

6.1 Security Analysis

The proposed scheme satisfies the requirements of e-voting as follows.

Privacy: Voters submit their votes anonymously while showing their tokens, therefore no one except voters themselves can know votes of individual voters. Anyone cannot know voters who did not submit their votes either.

Accuracy and universal verifiability: For obtaining tokens the eligibility of voters are checked by their *ID* and *P*/*W* pairs, and no one can forge signatures on their tokens, therefore any unauthorized entity cannot put its vote. Also as voters put their votes in the positions of *VotingPanel* corresponded to their tokens, multiple voting is prevented. Moreover, uniqueness of registered *CNs* and signatures on them ensure that all and only votes approved by individual voters are counted.

Fairness: No single entity can decrypt interim voting results because votes on *VotingPanel* are repeatedly encrypted by multiple Tallying managers.

Receipt-freeness: Voters know only their tokens, encrypted votes and encrypted *CNs*, and all of them cannot be linked to their votes in *TallyingPanel*. Therefore the scheme is *receipt-free*.

Incoercibility: When decrypted votes in 2 different signed forms are equivalent, no one can claim that votes are disrupted, therefore coercers cannot invalidate elections by claiming vote disruptions while forcing voters to submit disrupted votes. Receipt-freeness of the scheme disables coercers to identify voters who had put meaningless votes, therefore voters can abstain from elections without being noticed by coercers by casting meaningless votes. Also the uniqueness of signed tokens that enable voters to prove their eligibilities, disables coercers to submit votes on behalf of voters.

Dispute-freeness: Publicly-verifiable data about interactions among entities on the *BBs*, signature pairs on votes and the disruption detection processes enable entities to resolve disputes.

Robustness: Voters can disrupt only their votes by submitting invalid votes. Either *VM* or *TMs* cannot disrupt votes. Because the correctness of votes in *VotingPanel* is ensured by individual voters' approvals, and inconsistent votes and the liable entities are identified at the disruption detection stage, inconsistencies can be recovered by simply decrypting inconsistent votes again.

Scalability: *CNs* simplify the computations of individual entities *e.g*. voters, *TMs* etc. while maintaining the total accuracy and the incoercibility of the election as demonstrated in Sec 6.2.

Practicality: The scheme is based on weaker assumptions about trustworthiness regarding entities *i.e*. nothing can make the scheme unreliable unless multiple entities conspire. The scheme does not assume any absolutely trustworthy authority.

6.2 Performance Evaluation

A prototype system of the proposed scheme consists of 3 Tallying managers has been developed, and the computation times required for registration, voting and tallying are measured and the performances are compared with those of Scratch & Vote (S&V) [11] and Coercion-Resistant Voting (CRV) [16] which are available for comparisons. The environment consists of a 1.60 GHz CPU with 504 MBytes of RAM, and GMP [17] 1024 bit modulus running on Windows XP is used for encryptions. The time required for registering a voter is 0.0471 secs, for generating a vote is 0.308 secs, and for tallying a vote is 0.171 secs. Regarding the tallying, 1000 votes can be counted within 171 secs (*i.e*. 0.171 * 1,000 = 171), and this shows that the scheme is scalable and practical enough.

Table 1. Computation time required by the proposed scheme

Registration (m. secs)		Voting (m. secs)		Tallying (m. secs)	
Blinding	0.3	V_j's encryption	3.0	Decryption	133.0
Signing	45.0	*TMs* encryption	17.0	Verification	38.0
Unblinding	1.8	V_j's decryption	9.0		
		Verification	108.0		
		Signing	135.0		
		Verification	36.0		
Total	47.1	Total	308.0	Total	171.0

The registration of voter V_j is comprised of token T_{tj} blinding, signnature pair generations of 3 *TMs i.e*. generating 6 different signatures on blinded T_{tj}, and unblinding of 6 signed blinded T_{tj}. Here it is assumed that encrypted *CNs* and encrypted unknown random numbers are prepared in advance, therefore their computation time is not considered. The construction of vote v_j is comprised of the encryption of v_j by V_j itself, 3 *TMs*' triple encryptions of v_j, V_j's decryption of it, V_j's verification of *TMs*' encryptions of v_j and x_{xji}, *TMs*' repeatedly signing on encrypted vote and *CN* and V_j's verification of both forms of *TMs*' signatures. The time for tallying is comprised of decryptions and shuffles and verifications of 2 signed forms of votes and single signed form of *CNs*. Table 1 shows the computation time of each stage.

Table 2. Computation time comparisons with other schemes

	CPU	Registration	Voting	Tallying
Proposed scheme	1.6 GHz	0.0471 secs	0.308 secs	0.171 secs
CRV	2.0 GHz	-	-	26 ~ 62 secs
S&V	2.8 GHz	-	1 ~ 2 min	-

As Table 2 shows, compared with CRV that rely on ZKP, *CNs* of the proposed scheme substantially reduced the computation times *i.e*. the time required for the tallying is reduced at least more than 1/100 times. In the table all computation times of all schemes do not include the communication time. Here CRV adopts threshold (n, t) ElGamal as the base encryption algorithm while using 5 and 3 as n and t values, where n is the total number of authorities and t is the threshold. The tallying process of CRV is comprised of verification of votes by NIZKPs, shuffling of verified votes, elimination of duplicated votes, shuffling of votes with unique credentials, shuffling of encrypted credentials of registered voters, collision detections of registered credentials, separations of votes with invalid credentials, decryptions and tallying.

Among the above operations all shufflings are carried out by verifiable mixnets each consists of multiple Tallying managers. Although the computation volumes of individual encryptions/ decryptions and shuffles included in these mixnets are the same as those in the proposed scheme, *i.e*. they are propotional to key lengths, verifiable features of the mixnets supportrd by NIZKP make the whole computations complicated, when compared with the proposed scheme supported by *CNs*. Different from the proposed scheme, in which each TM_i executes multiplications corresponding to 6 decryptions for each vote, CRV requires huge number of multiplications for each vote to verify the correct behaviors of mixnets, *i.e*. to conduct each NIZKP process reliably usually about 80 times of challenges and responses are necessary each of which requires the same numbers of multiplications as encryptions and decryptions do. Also, the computation time required for tallying in the proposed scheme is strictly proportional to *N*, the number of voters, on the other hand that in CRV is the order of N^2 because it must eliminate duplicated votes, although it is suppressed to the linear order by using hashtables. Because voters carry out voting processes interactively, time required for voting is not so serious as tallying, therefore CRV did not mention the time of voting. Regarding the proposed scheme, 0.308 secs can be considered practical and scalable enough also. Moreover, many processes can be carried out in parallel by multiple managers if required. S&V is a paper based cryptographic voting system that offers entirely paper- and pen-based ballot casting, therefore the voting procedure is comparatively time consuming.

7. Conclusions

The proposed e-voting scheme achieves verifiability while disabling all entities including voters themselves to know the links between voters and their votes. Namely, the scheme satisfies all the essential requirements of e-voting. Most importantly, while being supported by *CNs*, these are achieved in a simple and efficient way. Unlike complicated ZKP based systems, the simplified computational requirements of individual election entities make the scheme practical and scalable.

(Manuscript received July 02, 2009, revised June 15, 2010)

References

(1) K. Sampigethaya and R. Poovendran, "A Framework and Taxonomy for Comparison of Electronic Voting Schemes," Elsevier Computers and Security, Vol. 25, pp. 137-153, 2006.

(2) B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang and S. Yoo, "Providing Receipt-freeness in Mixnet-based Voting Protocols," Proceedings of the ICISC '03, pp. 261–74, 2003.

(3) J. Benaloh and D. Tuinstra, "Receipt-free Secret-ballot Elections," Proceedings of 26th Symp. on Theory of Computing, pp. 544–553, 1994.

(4) M. Hirt and K. Sako, "Efficient Receipt-free Voting Based on Homomorphic Encryption," Proceedings of EUROCRYPT, LNCS, Vol. 1807, pp. 539-556. Springer, 2000.

(5) J. Schweisgut, "Coercion-resistant Electronic Elections with Observer," 2nd International Workshop on Electronic Voting, Bregenz, August 2006.

(6) A. Juels and M. Jakobsson, "Coercion-resistant Electronic Elections," Cryptology ePrint Archive, Report 2002/165, <http://eprint.iacr.org/>; 2002.

(7) A. Acquisti, "Receipt-free Homomorphic Elections and Write-in Ballots,". Cryptology ePrint Archive, Report 2004/105, <http://eprint.iacr.org/>; 2004.

(8) D. Chaum, "Secret-ballot Receipts: True Voter-verifiable Elections," IEEE Security & Privacy Magazine, Feb 2004.

(9) B. Riva and A. Ta-Shma, "Bare-Handed Electronic Voting with Pre-processing," Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop, Boston, MA, 2007.

(10) S. Tamura and T. Yanase, "Information Sharing among Untrustworthy Entities," IEEJ Trans. EIS, Vol. 125, No.11, pp.1767-1772, 2005.

(11) B. Adida and R. Rivest, "Scratch and Vote: Self-Contained Paper-based Cryptographic Voting," Workshop on Privacy in Electronic Society 2006.

(12) A. Fujioka, T. Okamoto and K. Ohta, "A Practical Secret Voting Scheme for Large Scale Elections," AUSCRYPT '92. LNCS, Vol. 718. Springer-Verlag, pp. 248–59, 1993.

(13) J. Wen-Shenq, L. Chin-Laung and L. Horng-Twu, "A Verifiable Multi-authority Secret Election Allowing Abstention from Voting," The Computer Journal, Vol. 45(6), pp. 672–82, 2002.

(14) D. Chaum, A. Fiat and M. Naor, "Untraceable Electronic Cash," in CRYPTO 88, Springer-Verlag, pp. 319 –327, 1988.

(15) S. Tamura, A. K Md. Rokibul and H. A. Haddad, "A Probabilistic and Commutative Re-Encryption Scheme," in Asia Simulation Conference 2009, ID 032, Ritsumeikan University, Shiga, Japan, October 7-9, 2009.

(16) S. Weber, "A Coercion-Resistant Cryptographic Voting Protocol -Evaluation and Prototype Implementation," Diploma thesis, Darmstadt University of Technology; 2006.

(17) T. Granlund. GNU Multiple Precision Arithmetic Library (GMP). Software available at http://gmplib.org/ April 2009.

(18) A. K. Md. Rokibul and S. Tamura, "Electronic Voting Using Confirmation Numbers," Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, pp. 4672-77, 2009.

Kazi Md. Rokibul Alam (Non-member) received the B.Sc. degree from Khulna University and M.Sc. degree from Bangladesh University of Engineering and Technology both in Computer Science and Engineering in 1999 and 2004, respectively. He is currently a doctor course student of University of Fukui. His research interests include cryptography, ubiquitous system security, machine learning and database management systems.

Shinsuke Tamura (Member) was born in Hyogo, Japan on Jan. 16 1948, and received the B.S., M.S. and Dr. (Eng.) degrees in Control Engineering from Osaka University in 1970, 1972 and 1991, respectively. During 1972 to 2001 he worked for Toshiba Corporation. He is currently a professor of Graduate School of Engineering, University of Fukui, Japan. Prof. Tamura is a member of IEEJ, SICE and JSST.

Shuji Taniguchi (Non-member) received the B.E. and Ph.D. degrees in electronics engineering from University of Fukui, Fukui, Japan, in 1973, 1996, respectively. In 1973-1978, he was with the Hitachi co. Ltd. He is currently an associate professor of Graduate School of Engineering in University of Fukui. His research interests include speech and hearing sciences and speech information processing.

Tatsuro Yanase (Non-member) received the Dr. (Eng.) degrees in Electric & Electronic Engineering from Nagoya University in 1977. During 1967 to 1969 he worked for Nippon Calculating Machine Corporation. He is now an associate professor of Graduate School of Engineering, University of Fukui. His research interests include distributed system architecture, computer graphics, and cellular automate.