Int. Computer Symposium, Dec. 15-17, 2004, Taipei, Taiwan.

A Public Verifiable Security-Enhanced Voting Protocol for a Committee Board Meeting

Pei-yih Ting , Po-Yueh Hung, Ching-Yi Chen
{pyting, m92570002, m91570019}@mail.ntou.edu.tw

Abstract- *In this paper, we focus on the design of an electronic voting protocol used in critical elections at the meeting of board of a company or a parliamentary committee. The number of participating voters is limited to several hundreds but the security requirement is expanded that the accumulated vote-count of each candidate should be kept as secret as possible. Most former electronic voting frameworks simply take the announcement of vote-counts for granted and neglect the privacy aspect of this part, therefore, they are not applicable directly. In the proposed protocol, the ElGamal cryptosystem is used with its nice homomorphic property. An electronic bulletin board is used to hold public announced values during the protocol. A ballot includes separate encrypted 'yes'/'no' vote for each candidate such that the accumulated vote-counts can be concealed in the secret comparison algorithm. The secrecy goals of this protocol are achieved not only by the design that each encrypted ballot is never decrypted but also by the deliberately designed comparison algorithm, in which a 'mix-and-match' sub-protocol is utilized.*

1. Introduction

Consider the following scenarios: In a small meeting room, members of the board of directors are voting for the chairman and the vice chairman of the next term. Each one can vote for two candidates without priority in his ballot. The company organization rule specifies additionally that the winning candidate must have majority supports. After all the anonymous ballots are disclosed and tallied, the winners are fairly generated. However, the disclosed ballots leave many intricate traces that might have sufficient political influences to the management level and are detrimental to the cooperative framework of the company. For example, if the vote count for the CEO is just on the legal margin of majority, some members might challenge the competence of the winner's leadership. If some candidates who actively campaign for the position and are promised privately a certain amount of support before the election, but the results turns out to have less supporting votes, some distrust and disgust might arise. In such a situation, a practical solution is by resorting to a trusted third party who counts all ballots secretly and announces only the resulting winners.

A second situation happens in parliamentary elections in which members are representatives from various political parties. There might be some predetermined principles on a certain topic for each member of a political party to obey. Sometimes the principles might even be against their own judgment or against the benefits of most people in the country. However, the public vote-counting and comparison process forces most participants to obey the principles of their political party. Otherwise, the tallied vote-counts would reveal some traces about possible traitors. In such a situation, extensive discussions, which usually appear at the meeting before the voting, seem to be superfluous. The seemingly fair public vote-counting process actually makes voters difficult to vote according to his sober and independent judgment. Somehow, the advantage of voting as a better collective decision method is lost. Although it is well known that many independent decisions with correctness probability slightly higher than a half will accumulate to a good decision with correctness probability far away from a half and hopefully close to unity. In many such occasions, the interests of the whole company or country could be sacrificed for maintaining the superficial peaceful cooperating atmosphere. In this paper, we would like to present a secret and fair vote-counting and comparison procedure that does not disclose the vote counts directly. The proposed protocol implements a vote-counting and comparison procedure that tries to hide as much information as possible except the names of the winning candidates. At the same time, it is public verifiable thus retains the fairness requirements.

In the past decade, there were extensive researches on electronic voting schemes, [8], [4], [13], [15], [3], [17], [18] to name a few. These schemes make a general electronic election for thousands or millions of voters feasible. The primary achievements include maintaining the privacy of each independent votes, preserving the fairness, preventing vote-buying and coercion, realizing vote-and-go concepts while reducing the computation and communication of the voting center and the voter. However, while focusing on achieving the above properties, all electronic voting schemes announce the final vote count of each candidate in order to determine the winner fairly. For some systems, this announcement is the by-product in the course to achieve effectiveness. For some other schemes, this announcement leaves sufficient auditing traces to ensure the fairness of the trusted authorities. However, this type of counting and comparison procedure maintains the privacy of each voter's independent choice only when the number of voters is sufficiently large. None of the above schemes can be applied directly to our applications without considerable modifications.

From the aspect of saving computation efforts, there seems no obvious reason to automate electronically a voting scheme in a moderate-size meeting room. However, if we rely on a trusted person to compare all votes secretly in the anonymous voting scenario where vote-counts are also required to be secret, this trusted person is very likely to be bribed or coerced later. Therefore, a good cryptographic voting scheme without trusted authority is necessary in this application scenario.

In the proposed protocol, the ElGamal cryptosystem is used with its nice homomorphic property. Each ballot includes separate encrypted 'yes'/'no' vote for each candidate such that the accumulated vote-counts can be concealed in the secret counting-and-comparison algorithm. The correctness of each encrypted ballot is guaranteed with zero knowledge proofs. Each encrypted ballot is never decrypted. Even the accumulated vote counts are not decrypted directly. The proposed protocol then compares these encrypted vote counts to determine the winner. Two types of elections, 'R-out-of-m ordered' and 'R-out-of-m unordered', are implemented with heapsort-based algorithm and quicksort-based algorithm, respectively. The privacy of each vote-count in a pairwise ciphertext comparison is maintained by the 'mix-and-match' sub-protocol [14]. All the public values in the proposed protocol are announced on an electronic bulletin board in such a way that every participant can write in his specified fields along the process of the protocol but cannot modify other's fields.

In section 2, we introduce the underlying cryptographic primitives. In section 3, we present the proposed secure voting protocol. Some variations and performance issues are presented in section 4. Section 5 contains the concluding remarks.

2. Cryptographic Primitives

2.1 Secret Sharing Scheme

In our protocol, we use Pederson's (t, n) threshold secret sharing scheme[16] which features a key sharing protocol without any trusted dealer. Each voter V_i chooses a random degree t-1 polynomial $f_i(\theta) = x_i + \sum_{j=1}^{t-1} a_{ij}\theta^j$ and announces $g^{a_{ij}}$ where g is the generator of the subgroup in the ElGamal system. A voter V_h shares his secret x_h with another voter V_i by sending $s_{hi}=f_h(i)$ to V_i. On receiving a share s_{hi}, V_i can verify it with announced public values. After voter V_i receives all correct sub-shares s_{hi}, he needs to apply the secret sharing homomorphism to obtain the t-share of the real secret $X = \sum_{i=1}^{n} x_i = F(0)$ where the degree t-1 super polynomial $F(\theta)$ is defined as $\sum_{i=1}^{n} f_i(\theta)$.

Let $T=\{V_p\}_{p=1\ldots k}$, $t \leqq k \leqq n$, be the set of cooperative voters. To jointly decrypt an ElGamal ciphertext (α, β) on the bulletin board, a voter V_p in the set T needs only compute independently $\alpha^{F(p)\prod_{q\neq p, q\in T}(-q)/(p-q)}$ and publishes it.

The decrypted plaintext is $D(\alpha,\beta) = \beta / \alpha^{F(0)} = \beta \Big/ \left(\prod_{p\in T} \alpha^{F(p)\prod_{q\neq p,q\in T}(-q)/(p-q)} \right)$. To thwart disruptions by malicious participants, the following ZKP should be supplied by V_p for proving that the F(p) used in $\alpha^{F(p)\prod_{q\neq p, q\in T}(-q)/(p-q)}$ is consistent with all previous published values.

2.2 The Ballots

In the proposed protocol, each voter V_i encrypts his 'yes'/'no' vote for each candidate as a vector of ElGamal ciphertexts, denoted as C_i:

$C_i = (c_{i1}, \ldots\ldots\ldots, c_{ij}, \ldots\ldots\ldots\ldots\ldots, c_{im}) = (E_K(z), E_K(1), \ldots\ldots\ldots, E_K(z), E_K(z), \ldots\ldots, E_K(1))$, $1 \leqq i \leqq n$, $1 \leqq j \leqq m$, n is the number of voters and m is the number of candidates. An $E_K(z)$ in the j-th component of C_i represents a 'yes' vote for the j-th candidate, and an $E_K(1)$ represents a 'no' vote. There are w_i 'yes' vote in each C_i where w_i is an integer between zero and L, where L is the maximum number of 'yes' votes in each ballot. The constant z is chosen such that its order in G_q is larger than 2n.

Each voter must prove that his encrypted vote vector C_i is valid. It ensures that every voter votes at most L candidates such that the remaining homomorphic counting process is correct. First, a voter must prove that each element c_{ij} of his ballot corresponds to a plaintext which is either z or 1. Second, using the multiplicative homomorphic property of the ElGamal cryptosystem[7], the voting center can calculate $\chi_i = \prod_{j=1}^{m} c_{ij}$ which equals $E_K(z^{w_i})$. Each voter is required to prove either that c_{ij} decrypts to one element of $\{1, z\}$ or that χ_i decrypts to one element of a finite set of plaintexts $S=\{z^1, z^2, \ldots, z^L\}$ with the following "decryption of an ElGamal ciphertext belongs to a finite set" ZKP.

ZKP of "decryption of an ElGamal ciphertext belongs to a finite set S":

Consider the following public parameters of an ElGamal cryptosystem: p, q, g, K, (α, β), and S where p and q are large prime numbers, p=2q+1, g is a generator in the order q quadratic residue subgroup G_q of Z_p^*, $K \equiv g^X \pmod p$ is the public key, X is the private key, X is chosen such that gcd(X, p-1) is 2 and the order of K in G_q is q, $(\alpha \equiv g^r \pmod p$, $\beta \equiv M_i \cdot K^r \pmod p))$ is an ElGamal ciphertext, $r \in_R Z_{p-1}$, S is the finite set of specified messages $\{M_1, \ldots, M_J\}$, $M_i \in G_q$. Peggy wants to prove to Victor that she knows i and r without revealing them.

1. Peggy picks randomly J-1 even values $\{e_j\}_{j\neq i} \in_R Z_{p-1}$ and J-1 even values $\{y_j\}_{j\neq i} \in_R Z_{p-1}$, then she computes $\{a_j \equiv g^{y_j} \alpha^{-e_j} \pmod p\}_{j\neq i}$, $\{b_j \equiv K^{y_j} \cdot \{M_j/\beta\}^{e_j} \pmod p\}_{j\neq i}$, chooses a random even $w \in_R Z_{p-1}$ and calculates $a_i \equiv g^w \pmod p$, $b_i \equiv K^w \pmod p$. Finally, she commits $\{a_j, b_j\}_{j=1\ldots J}$ to Victor.

2. Victor chooses a random even challenge $e \in_R Z_{p-1}$ and sends it to Peggy.
3. Peggy computes $e_i \equiv e - \sum_{j \neq i} e_j \pmod{p-1}$, $y_i \equiv w + r \cdot e_i \pmod{p-1}$, and sends the response $\{e_j, y_j\}_{j=1...J}$ to Victor.
4. Victor checks that $e \equiv \sum_j e_j \pmod{p-1}$ and that $g^{y_j} \equiv a_j \cdot \alpha^{e_j} \pmod{p}$, $K^{y_j} \equiv b_j \cdot (\beta/M_j)^{e_j} \pmod{p}$ for j=1...J.

A formal proof of this ZKP can be found in[19].

3. Secure Voting Protocol

The complete secure voting protocol, including initialization, voting, and counting stages, is presented as follows:

3.1 Initialization stage

Enroll, choose the secret, and decide the public key: Each participating voter needs to present a valid certificate signed by a specific certificate authority (CA) at this stage in order to enter the voting. Each voter is then given a one-time password for the electronic bulletin board. This allows a voter to write in his specific field exactly once.

Step 1. A voter V_i chooses a secret x_i, computes g^{x_i}, and publishes g^{x_i}.

Step 2. Each voter computes the public key $K \equiv g^X \equiv g^{x_1+x_2+......+x_n} \equiv g^{x_1} \cdot g^{x_2} \cdot ... \cdot g^{x_n} \pmod{p}$.

Share the secret key:

Step 3. V_i chooses secretly a degree t-1 polynomial $f_i(\theta) = x_i + \sum_{j=1}^{t-1} a_{ij}\theta^j$ which hides the secret x_i as the constant term, and publishes the exponentials $g^{a_{ij}}$ of all coefficients.

Step 4. V_i sends $f_i(h)$, which is the h-th share of x_i, to the voter V_h.

Step 5. V_i cross verifies all the shares he received, $\{s_{hi}\}_{h=1...n}$, against published values in step 3. i.e.,

$$g^{s_{hi}} \equiv g^{f_h(i)} \equiv (g^{x_n})(g^{a_{b_1}})^i (g^{a_{b_2}})^{i^2} \cdots (g^{a_{b_{t-1}}})^{i^{t-1}}$$

Step 6. V_i calculates from $\{s_{hi}\}_{h=1...n}$ the share $F(i) = \sum_{h=1}^{n} f_h(i) = \sum_{h=1}^{n} s_{h_i}$ of the actual decryption key $X=F(0)$.

Prepare randomly re-encrypted permutation ciphertext sets:

Define $S_+=\{z^1, z^2, ..., z^n\}$. Calculate all elements in S_+, encrypt them, and feed these ciphertexts into a verifiable mix-net[1] to create a randomly permuted set $S_+^{(enc)}$. A voter acts as a mix-server in the mix-net. The input set is first blinded, i.e. each ElGamal ciphertext is multiplied by (g^γ, k^γ) where γ is a random number generated for each ciphertext. Then the blinded set is permuted. The blinding numbers and the permutation are secret to the mix-server. However, the mix-servers are required to provide NIZKPs jointly for the overall operations. After a sequence of distributed operations, the output is a randomly re-encrypted permutation $S_+^{(enc)} = \{s_k\}_{k=1...n} = \{E_K(z^1), E_K(z^2), ..., E_K(z^n)\}$. Each voter is assured that $S_+^{(enc)}$ contains the ciphertexts of all plaintexts in S_+, but he is not able to determine the correspondence between these two sets. Because each comparison used one $S_+^{(enc)}$, participants jointly prepare sufficient number of ciphertext sets $S_+^{(enc)}$, which consists of ciphertexts of z raised to all positive offsets powers. One should note that the number of voters, n, cannot be too large for practical reasons in this protocol and the order of z in the multiplicative group G_q is chosen to be larger than 2n.

3.2 Voting stage

Prepare the ballot:

Step 1. A voter V_i decides 'yes'/'no' for each candidate. He prepares a ballot C_i = ($E_K(z)$, $E_K(1)$, $E_K(z)$, ..., $E_K(1)$). An $E_K(z)$ represents a 'yes' vote and an $E_K(1)$ represents a 'no' vote. There are w_i 'yes' vote in the ballot C_i and w_i is no more than a constant L.

Prove the validity of a ballot: There are several illegal cases the protocol would like to avoid. First, a ballot contains an element $E_K(z^k)$. It is effectively k 'yes' votes to the same candidate. Second, a ballot contains more than L 'yes'-votes. Third, a ballot contains illegal encrypted elements other than $E_K(z)$ or $E_K(1)$ that would disrupt the homomorphic counting procedure.

Step 2. A voter proves that each ciphertext element of the ballot is the encryption of either z or 1 with the NIZKP described in section 2.2.

Step 3. A voter also proves that $\chi_i \equiv \prod_{j=1}^{m} c_{ij} \pmod{p}$ is the encryption of an element in the set $\{1, z^1, z^2, ..., z^L\}$ with the NIZKP described in section 2.2.

Publish the ballot:

Step 4. A voter logs on the electronic bulletin board with the one-time password received earlier.

Step 5. A voter publishes the ballot and the NIZKPs on his designated fields as shown in Figure 1.

Candidates / Voters	1	...	m-1	m	
V_1	c_{11}	...	$c_{1(m-1)}$	c_{1m}	NIZKP for c_{1j} NIZKP of χ_1
...	..		...	...	
V_n	c_{n1}	...	$c_{n(m-1)}$	c_{nm}	NIZKP for c_{nj} NIZKP of χ_n
Tallies	τ_1		τ_{m-1}	τ_m	

Figure 1. Ballots on the electronic bulletin board

Verify the validity of NIZKPs:

Step 6. In order to prevent malicious subversion in the remaining protocol, the voting center has the responsibility to verify the NIZKPs corresponding to each encrypted vote vectors.

Step 7. Because none of the published c_{ij} will be decrypted in the protocol, any suspicious participant can also verify these proofs to establish his confidence on the ongoing protocol.

3.3 Counting-and-Comparison stage

In this stage, the encrypted votes for each individual candidate are first tallied using the homomorphic property of the underlying encryption system. Then R winners out of m candidates are generated without decrypting their individual vote-tallies with a public verifiable counting-and-comparison protocol. Protocols for two election schemes are constructed for this purpose: 'R-out-of-m ordered' election and 'R-out-of-m unordered' election. The first scheme concludes the R winners with their relative order while the second scheme tries to hide the order of the final R winners.

A. Homomorphic vote-tallying:

The ciphertexts of the j-th column on the electronic bulletin board are multiplied together as the encrypted tally of the j-th candidate, i.e.

$$\tau_j = \prod_{i=1}^{n} c_{ij}$$

B. Secure determination of R winners:

Depending on the choice of R and m, different sorting algorithm is required to determine the result of the election in order to obtain minimal number of comparisons.

1. 'R-out-of-m ordered' election:

To achieve this goal, a general sorting algorithm can be applied until the highest R elements come out. The 'match' sub-protocol[14], to be described later, is used to carry out each pairwise comparison without revealing both tallies. For example, a selection sort algorithm[12] can be applied on the ciphertext list $\{\tau_1, \tau_2, ..., \tau_m\}$. With (m-1) comparisons and exchanges, the largest element of the list can be sorted out. With (m-2) comparisons and exchanges the runner-up can be sorted out. In the same way, this procedure runs till the R-th largest element comes out.

Since the 'match' sub-protocol is the most time consuming part of this counting-and-comparison protocol, it is necessary to consider other sorting algorithm that reduces the number of comparisons. Consider the heap sort algorithm[12] that requires asymptotically m log(m) comparisons in the worst case:

Step 1: Use the 'heapify' algorithm to construct a maximal heap (a binary tree with every non-leaf node larger than both of its children) from the ciphertext list $\{\tau_1, \tau_2, ..., \tau_m\}$. The number of comparisons of this step is $2\sum_{1\le i\le k} 2^{i-1}(k-i)$ where $k = \lceil (\log m)+1 \rceil$.

The root of this maximal heap is the first winner.

Step 2: Replace the root node, with the rightmost leaf node of the bottom layer and adjust the heap to a maximal heap again. The number of comparisons of this step is less than 2 (k-1).

Repeat step 2 for (R-1) times will find the (R-1) ciphertexts corresponding to (R-1) maximum vote-tallies.

2. 'R-out-of-m unordered' election:

It is hard to find out the R winners without letting go any of their pairwise relations. In the following, an algorithm that only disclose approximately 2m pairs of relations is suggested:

Step 1: Randomly pick a pivot element τ_i from the ciphertext list $\tau_1, \tau_2, ..., \tau_m$, compare τ_i with $\{\tau_j\}_{j\neq i}$ using the 'match' sub-protocol, put those element with corresponding plaintext less than $D_K(\tau_i)$ in U_1 and others in U_2. (Note: $D_K(\tau_i)$ denotes the plaintext corresponding to τ_i.)

Step 2: a. If $|U_2| > R$, apply step 1 and step 2 recursively on list U_2 to find the maximal R elements out of U_2.

b. If $|U_2| < R$, apply step 1 and step 2 recursively on list U_1 and find the maximal R-$|U_2|$-1 elements out of U_1.

c. If $|U_2|$ equals R or R-1, the algorithm stops with U_2 or $U_2 \cup \{\tau_i\}$ as the result winners.

C. The 'match' pairwise comparison sub-protocol:

In the counting-and-comparison stage, random permuted sets $S_+^{(enc)}$ consisting of ciphertexts of z raised to all positive offsets are required to compare tallies of any two candidates without revealing their counts. To determine the relative relation of any $D_K(\tau_i)$ and $D_K(\tau_j)$ pair without revealing their difference, the following 'match' [14] steps are performed:

Step 1: Divide τ_i by τ_j to obtain the ciphertext $E_K(z^{t_i-t_j})$, where τ_i equals $E_K(z^{t_i})$ and τ_j equals $E_K(z^{t_j})$.

Step 2: Determining whether the decryption of this ciphertext (τ_i / τ_j) is unity, an element of S_+, or an element of S_-.

2.1: Raise (τ_i / τ_j) to a secret random power sequentially by the k-th participant, i.e. $(\tau_i / \tau_j)^{\lambda_k}$ such that no participant knows the overall random exponent $\lambda=\lambda_1+\lambda_2+...+\lambda_n$. Note, the notation $(\tau_i / \tau_j)^{\lambda_k}$ denotes a separate λ_k power to both elements of an ElGamal ciphertext. Each participant performs the exponentiation twice and supplies an 'equal discrete logarithm' ZKP[5] that he uses the same exponent λ_k in both operations.

2.2: Perform jointly a threshold decryption of $(\tau_i / \tau_j)^{\lambda}$; we reach a conclusion that τ_i equals τ_j if the result is unity.

2.3: Divide (τ_i / τ_j) by each element of $S_+^{(enc)}$ and obtain a set $\{\tau_i / \tau_j / s_l\}_{l=1...n}$.

2.4: Raise each elements $(\tau_i / \tau_j / s_l)$ to a secret random power sequentially by each participant, i.e. $(\tau_i / \tau_j / s_l)^{\lambda_k}$ such that no participant knows the random exponent λ.

2.5: Perform jointly threshold decryptions of $\{(\tau_i / \tau_j / s_l)^{\lambda}\}_{l=1...n}$; if there is a unity element inside, conclude that $\tau_i > \tau_j$; otherwise conclude that $\tau_i < \tau_j$.

If only one set $S_+^{(enc)}$ is used in all 'match' sub-protocol, and suppose two pairs of ciphertexts (a, b) and (c, d) matches to the same element of $S_+^{(enc)}$ in two rounds of 'match' sub-protocol. Then we know not only $D_k(a) > D_k(b)$ and $D_k(c) > D_k(d)$ but also $D_k(a)-D_k(b) = D_k(c)-D_k(d) > 0$, which leaks a lot more information than just relative order of them.

4. Efficiency and Security

In the proposed scheme, the ciphertext $E_K(1)$ or $E_K(z)$ represents a 'no'-vote or a 'yes'-vote for a specific candidate, respectively. In a ballot, approval or disapproval of a total of m candidates requires m separate ElGamal ciphertexts. This approach apparently uses a large amount of computation and storage. In contrast, if suitable coding is used for each ballot the representation is extremely compact. For example, the ballot can be represented as $E_K(z^{a_0+a_1N+a_2N^2+...+a_{m-1}N^{m-1}})$ [6], where $a_i \in \{0, 1\}$ is the 'no'/'yes' decision, $ord_p(z) > N^{m+1}$, N is a number chosen to be larger than the number of voters n. In this representation, $z^{a_iN^i}$ with a_i equal to 1 denotes an approval for the i-th candidate. The homomorphic property of ElGamal encryption system enables the correct counting without decryption of each individual ballot. However, before submitting this encrypted ballot, a voter has to provide a ZKP to ensure the validity of the ballot. One obvious yet impractical method is to create a set of all possible ballots $\{1, z^{N^0}, z^{N^1}, z^{N^2}, ..., z^{N^{m-1}}, z^{N^0+N^1}, z^{N^0+N^2}, ...z^{N^{m-2}+N^{m-1}}, z^{N^0+N^1+N^2}, ...z^{N^0+N^1+N^2+\cdots N^{L-1}}, ..., Z^{N^{m-L}+N^{m-L+1}+N^{m-L+2}+\cdots N^{m-1}}\}$ and to prove that the corresponding plaintext of one's ballot is in this set. Due to the large amount of elements in this set, the computation and communication cost of this proof is generally not affordable by a voter. In the proposed scheme, to prove the validity of each individual 'yes'/'no' vote requires only a proof that $D_K(c_{ij})$ is in {1, z}. The requirement that there are at most L approvals in each ballot (which comprises m ciphertexts) can be proved with a ZKP that $D_K(\chi_i) \in \{1, z, ..., z^L\}$. The complexity of this proof grows linearly instead of exponentially in L. The second problem with the above compact ballot representation is: at the decryption stage, a discrete logarithm problem with respect to the new z, which has far larger order than the z in the proposed method, has to be solved. The third problem with the compact ballot representation is: at the vote-counting stage, the vote-count for each candidate is revealed all at a time since the sum of all ballots has to be decrypted. This violates the secrecy requirement of our application system.

In the proposed vote-counting and comparison procedure, the security of individual primitives (NIZKPs) is guaranteed by their own security proofs. The sequential composition of these protocols are also well known to be secure[11]. The fulfillment of the new privacy requirements by the proposed protocol actually depends on the number of comparisons in the course of protocol execution.

Assuming an 'R-out-of-m ordered' election is conducted at a meeting, the proposed 'match' algorithm to choose the R winners from homomorphically processed vote-tally ciphertexts will reveal many relative relations of vote tallies (out of a total of m(m-1)/2 relations). From these revealed information and the range of total number of votes, one can construct an upper bound and a lower bound of each vote tally. The more number of comparisons performed, the more relations revealed, and the more exact the calculated bounds. In the heapsort-style algorithm, choosing R winners from m tallies requires at worst $2(\sum_{1\le i\le k} 2^{i-1}(k-i)+(R-1)(k-1))$ comparisons where $k = \lceil (\log m)+1 \rceil$. In the above equation, $2\sum_{1\le i\le k} 2^{i-1}(k-i)$ is the number of comparisons in the 'heapify' process and $2(R-1)(k-1)$ is the maximal number of comparisons in choosing R maximal numbers from the maximal heap. Note that if we record the result of each comparison in the course of the protocol and eliminate duplicated comparisons in the sequel; the average number of comparisons in the 'R-maximum-choosing' process could be reduced to as small as one fourth such that the average number of comparisons for heapsort-based protocol is less than $2\sum_{1\le i\le k} 2^{i-1}(k-i)+(R-1)(k-1)/2$.

Considering instead using the selection sort in combination with the 'match' algorithm, choosing R winners out of m candidates requires $\sum_{1\le i\le R}(m-i)$ comparisons, and at most that number of relations are revealed. In table 1, the number of comparisons for heapsort-based protocol and for selection-sort-based protocol are tabulated. The results of the average case are averaged over 10000 random independent experiments. It can be observed that as number of winners increases, heapsort-based protocol outperforms the selection sort-based protocol.

# of candidates / # of winners	Heap sort		Selection sort
	Average	Worst case	
7/1	7.41000	8	6
7/2	9.998333	12	11
7/4	13.898000	19	18
31/1	45.991333	52	30
31/2	53.129667	60	59
31/10	106.325667	124	255

Table 1. Number of comparisons in the protocol

In many election systems, the vote-count for a candidate is required to exceed a certain threshold before that candidate being declared as a winner. In the proposed protocol, the encrypted vote-tallies for each candidate can be first compared with the encryption of this fixed threshold using the 'match' protocol. In this way, only qualified candidates will be compared in the proposed voting protocol. Another way to implement electoral threshold is to apply the proposed protocol on all tallies and only compare the vote-tallies of the final R winners to the threshold. Both methods incur different comparisons; however, they only make limited differences.

5. Concluding Remarks

In our scheme, the privacy of all tallies is maintained and the fairness of the whole protocol is guaranteed by the public verifiability of homomorphic operations and ZKPs with the extra expense of computation and storage. Almost all previously proposed schemes in the literature neglected the privacy requirement raised in this paper, namely the privacy of all vote-counts. It is to our knowledge the first attempt to formulate this security aspect. Another difference between our scheme and previous schemes lies in the trust structure. In a large-scale election, all the voters have to trust a single vote-counting center or a set of non-collaborating vote-counting centers. Some schemes might have extra scrutinizers to ensure the fairness of the counting procedure executed by the authority. For our target application, which has only a moderate number of voters, the trust is distributed to each voter himself or herself. A voter only needs to trust / observe that most other voters participating the meeting are not collaborating. In that case, the voting mechanism would be fairly conducted. However, if convenience and efficiency are the first-priority considerations, the trust can still be placed on multiple authorities and scrutinizers.

It should also be noted that there is no point to apply the customized scheme proposed in this paper to a large-scale election. First, the ZKPs required for the correctness of each ballot demand an excessive amount of computation, while the advantage of hiding the individual tallies makes not much sense in a large-scale election. It is clear that small-scale elections in a democratic style of business management team control most decisions of an enterprise. Independent voting decisions are assured by the proposed scheme. The remaining difficulties include: the massive amount of computations and communications that are still not affordable by a medium-level personal device like a cellular phone or a smart card. Also, the vote-buying (or the coercion) problem remains unsolved as for all homomorphic encryption-based voting systems.

It is well known that secure electronic voting schemes are special instances and successful applications of secure multi-party computation protocols[9]. However, for efficiency considerations in the deployment of a large-scale election, most previous schemes avoided the general multi-party computation model[10] and sacrificed some secrecy properties such as the counts of vote-tallies. At first glance, it looks like that Yao's secure two-party comparison protocol[20] can be applied to do the comparisons. However, the use of homomorphic encryption on the vote-tallying process hides all the vote-counts and discourages the application of Yao's protocol.

6.References

[1] M. Abe, "Universally verifiable MIX with verification work independent of the number of MIX servers", Adv. in Cryptology–Eurocrypt '98.

[2] M. Abe and K. Suzuki, "M+1-st Price Auction Using Homomorphic Encryption", PKC 2002.

[3] O. Baudron, P. Fouque, D. Pointcheval, J. Stern, and G. Poupard, "Practical Multi-Candidate Election System", ACM 20-th Symp. on Principle of Distributed Computing, PODC'01, 2001.

[4] J. C. Benaloh, "Verifiable Secret Ballot Elections", PhD thesis, Yale University, 1987.

[5] D. Chaum and T. Pedersen, "Wallet Databases with Observers", Adv. in Cryptology – Crypto'92.

[6] R. Cramer, R. Gennaro, and B. Schoenmakers, "A Secure and Optimally Efficient Multi-Authority Election Scheme", Advanced in Cryptology – Eurocrypt'97.

[7] T. ElGamal, "A Public-key Cryptosystem and Signature Scheme Based on Discrete Logarithms", IEEE Trans. on Information Theory, Vol. IT-31, pp. 469-472, 1985.

[8] A. Fujioka, T. Okamoto, and K. Ohta, "A practical secret voting scheme for large scale elections", Adv. in Cryptology – AUSCRYPT'92.

[9] M. Franklin and Z. Galil, "An Overview of Distributed Secure computing", 1992

[10] O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game", STOC'87.

[11] O. Goldreich and Hugo Krawczyk, "On the Composition of Zero-Knowledge Proof Systems", SIAM Journal on Computing 1996.

[12] E. Horowitz, S. Sahni and S. Rajasekaran, "Computer Algorithm/ C++", Computer Science Press, New York, 1997.

[13] M. Hirt and K. Sako, "Efficient receipt-free voting based on homomorphic encryption" , Advanced in Cryptology – Eurocrypt '00, 2000.

[14] M. Jakobsson and A. Juels, "Mix and Match: Secure Function Evaluation via Ciphertexts", Advanced in Cryptology – Asiacrypt '00, 2000.

[15] M. J. Radwin, "An untraceable, universally verifiable voting scheme", 1995, http://www.radwin.org/michael/projects/voting.html.

[16] T. P. Pedersen, "A Threshold Cryptosystem without a Trusted Party", Advanced in Cryptology – Eurocrypt 1991, pp. 522-526, 1991.

[17] R. Rivest, "Electronic Voting", Financial Cryptography'91, 1991.

[18] Z. Rjaskova, "Electronic Voting Schemes", Ms Thesis, Comenius University, Bratislava, 2002.

[19] P.-Y. Ting, Y.-T. Lee, C.-Y. Chen "On The Public Verifiability of an M+1-st Price Auction Using Homomorphic Encryption", Technical Report, National Taiwan Ocean Univ. 2003.

[20] A. Yao, "Protocols for secure computations", Proc. of the 23rd IEEE Symp. On Foundations of Computer Science, FOCS'82, 1982.