A Verifiable E-voting Scheme with Secret Sharing

Lifeng Yuan[1,2], Mingchu Li[1,2], Cheng Guo[1,2], Weitong Hu[3], and Zhihui Wang[1]
(Corresponding author: Cheng Guo)

School of Software Technology, Dalian University of Technology[1]
No.8 Road, Jinzhou District, Dalian 116620, China
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province[2]
No.8 Road, Jinzhou District, Dalian, 116620, China
School of Computer and Technology, Hangzhou Dianzi University[3]
No.1 Street 2, Xiasha High Education Zone, Hangzhou, 310018, China
(Email: guocheng@dlut.edu.cn)
(Received Nov. 28, 2015; revised and accepted Apr. 9 & May 31, 2016)

Abstract

Traditional e-voting schemes use centralized and non-transparent count centers, which leads to people's distrust of the centers and doubt of the voting on impartiality and correctness. In this paper, we propose a distributed and verifiable e-voting scheme based on Mignotte's threshold secret sharing scheme, which effectively balances the conflict of interest between voters and count centers. Additionally, this scheme can also resist potential attacks from malicious participants, and satisfy special requirements of e-voting, especially for privacy and accountability, which contradict each other. Moreover, voters take on the major computation load in our scheme, which effectively reduces the computation burden of the vote counter.

Keywords: Accountability, e-voting, secret sharing, verifiability

1 Introduction

As we know, voting is important in democratic society. In paper voting, it is possible for tally clerks to obtain and even tamper with the contents of voters' ballots during printing and delivery phase, so voters may doubt the authenticity of the final result. Moreover, paper voting takes great cost to count votes in the voting process. To fix these problems, a multitude of e-voting schemes based on various cryptographic techniques are developed, which are more convenient for voters to vote at any time and place. Therefore, e-voting is widely used to replace paper voting.

In recent years, various security technologies (such as mix-net [4, 5, 8, 24, 25], blind signature [3, 6, 12, 15, 29], homomorphic encryption [7, 9, 11, 26, 28] and secret sharing [10, 11, 17, 33]) provide a solid foundation for the development of e-voting. Compared with paper voting, an e-voting scheme should be able to satisfy more requirements [15, 21, 36], such as privacy, verifiability, fairness and transparency. Since one requirement may conflict with another (for example, accountability and privacy), it is challenging to satisfy all of them.

In this paper, we propose an e-voting scheme based on Mignotte's secret sharing schemes [23] with the following advantages:

1) It can balance the conflict of interest between voters and central vote counter by mutual supervision. In some verifiable voting schemes, the voter can only verify whether his/her own vote is computed correctly, but any voter in our scheme can verify whole vote result without affecting the privacy of the scheme, so their trust in this scheme can increase greatly.

2) It improves computational efficiency. Schemes that use central entities to execute all computation tasks often make central entities overloaded. However, in our scheme, voters take the majority of computation tasks. Meanwhile, for a single voter, the computation burden is acceptable.

3) It resolves the conflict between accountability and privacy. In our scheme, no one can obtain legal voters' selections. But, in order to identify attackers, the third-party authority can recover the voter's selection with t or more voters' assistance. However, this is inevitable and understandable. Note that the third-party authority also cannot obtain any voter' selection unless t or more voters agree.

Our scheme uses Mignotte's threshold secret sharing technique, which is based on the Chinese Remainder Theorem. Also, our scheme uses some special sequences of integers, referred to as Mignotte sequences. For reader's convenience, we describe Mignotte sequences as follows.

Let $n \in \mathbb{Z}$, $2 \leq t \leq n$. A (t, n) Mignotte sequence is a sequence of pairwise co-prime positive integers $p_1 < p_2 < \cdots < p_n$ such that

$$\prod_{i=0}^{t-2} p_{n-i} < \prod_{i=1}^{t} p_i \tag{1}$$

Given a publicly known (t, n) Mignotte sequence, the scheme works as follows:

1) The dealer chooses a secret $s \in \mathbb{Z}$, such that

$$\prod_{i=0}^{t-2} p_{n-i} < s < \prod_{i=1}^{t} p_i. \tag{2}$$

2) For all $1 \leq i \leq n$, the secret share s_i for participant P_i is computed by the dealer as $s_i = s \bmod p_i$.

3) If the number of participants with secret shares is greater than or equal to t, the secret s can be recovered. Without loss of generality, assume that t participants $P_{i_1}, P_{i_2}, \cdots, P_{i_t}$ $(1 \leq i_1 < i_2 < \cdots < i_t \leq n)$ provide their secret shares $s_{i_1}, s_{i_2}, \cdots, s_{i_t}$, then the system of congruences can be built as

$$\begin{cases} s_{i_1} \equiv s \bmod p_{i_1} \\ s_{i_2} \equiv s \bmod p_{i_2} \\ \quad\quad \vdots \\ s_{i_t} \equiv s \bmod p_{i_t} \end{cases} \tag{3}$$

where $s_j \equiv s \bmod p_j$ $(j \in \{i_1, i_2, \cdots, i_t\})$ means $s_j \bmod p_j = s \bmod p_j$. Let $P = \prod_{k=1}^{t} p_{i_k}$. For all $1 \leq k \leq t$, $R_{i_k} = P/p_{i_k}$ and $R_{i_k} r_{i_k} \equiv 1 \bmod p_{i_k}$. Then, the secret s can be recovered as

$$s = \sum\nolimits_{k=1}^{t} s_{i_k} R_{i_k} r_{i_k} \bmod P. \tag{4}$$

The rest of this paper is organized as follows: we present the related work in Section 2, and introduce some concepts of e-voting in Section 3. Section 4 describes the details of our e-voting scheme. In Section 5, we discuss correctness, security, features, computation complexity of our scheme, and then, compare our scheme with related schemes. Finally, Section 6 concludes this paper.

2 Related Work

In recent years, various e-voting schemes have been developed [18, 19, 20], and these schemes are based on different security methods, such as mix-net [4, 5, 8, 24, 25], blind signature [3, 6, 12, 15, 29], homomorphic encryption [7, 9, 11, 26, 28] and secret sharing [10, 11, 17, 33]. However, to the best of our knowledge, existing e-voting schemes cannot satisfy all the special requirements, which will be discussed in Section 3.1.

In 1981, Chaum [5] proposed the first e-voting scheme based on public-key cryptosystem and mix-net. His scheme can be described as follows: firstly, each voter authenticates himself/herself and then sends his/her encrypted vote. The vote counter collects all votes and terminates the corresponding relationship between the ballots and voters by mix-net. Finally, these votes will be decrypted and counted. Note that this scheme requires a lot of computation to guarantee that each vote be properly processed, so it is inefficient and not suitable for large-scale voting. Moreover, because the mix-net is not transparent, voters may doubt the correctness of the vote result. More schemes based on mix-nets can be referred in [4, 8, 24, 25].

Blind signature, introduced by Chaum [3] in 1983, allows an authority to sign an encrypted message without knowing the plaintext. Fujioka et al. [15] proposed a FOO e-voting scheme, and then, Cranor and Cytron [12] implemented the FOO scheme named Sensus. The FOO scheme also has problems like ballot collusion. More schemes [6, 29] based on blind signature were proposed afterwards, for example, Radwin [29] proposed an untraceable, universally verifiable voting scheme and Chen [6] proposed a receipt-free voting scheme using double-trapdoor commitment.

In 1985, Cohen and Fischer [9] proposed a voting scheme based on homomorphic encryption. Exploiting the homomorphism of certain encryption algorithms, the schemes [7, 11, 26, 28] do not decrypt a single ballot, but decrypt the product of all ballots to get the vote result. They are efficient in yes/no voting, while in other types of voting, they have low efficiency. In these schemes, voters need to use zero-knowledge proofs to prove the correctness of their votes. If the voting is complex (such as selecting $|n/2 - 1|$ people from n candidates), these schemes will require lots of computation.

In 1979, Shamir [34] and Blakley [1] proposed the concept of secret sharing independently, and built a (t, n) threshold secret sharing scheme respectively. Subsequently, many researchers further studied secret sharing technology [13, 14, 22, 27]. The secret sharing scheme has also been applied to e-voting, for example, schemes in [10, 11, 33] based on Shamir's polynomial interpolation secret sharing scheme, and Iftene's scheme [17] based on Chinese remainder theory. Since these schemes use centralized entities without transparency, voters cannot verify the vote result and thus may doubt the correctness of the vote result. Moreover, these schemes lack consideration about the impact of attackers. In this paper, we propose a verifiable e-voting scheme based on Mignotte's threshold secret sharing scheme. Our scheme enable voters to verify the vote result, and balances the conflict of interest between voters and the central vote counter. Additionally, this scheme can also resist potential attacks from malicious participants.

In addition, other new methods, such as image hiding [31] and quantum mechanism [2], were also applied to e-voting schemes, and some e-voting schemes use multiple methods together. For example, Cohen and Fisher's scheme [9] uses mix-net and blind signature together.

3 Preliminaries

In this section, we will discuss the e-voting requirements, the classification of e-voting, the participants of our scheme and the attack models.

3.1 Requirements of E-voting Scheme

Some literatures [15, 16, 21, 32, 35, 36] have discussed the requirements of e-voting schemes. The most important requirements are as follows:

1) **Legality**: only legal participants can vote.
2) **Correctness**: if all participants are honest and execute the schemes strictly, the vote result can reflect voters' intentions correctly.
3) **Privacy**: no one can obtain another voter's personal vote information.
4) **Robustness**: attackers cannot disrupt the vote procedure.
5) **Verifiability**: each voter can independently check the correctness of the vote result.
6) **Fairness**: each voter only knows his/her own vote information, and cannot know the final vote result until the vote has been finished.
7) **Transparency**: the whole voting process and all technologies used in the voting scheme are transparent to each voter.
8) **Uniqueness**: a voter is not allowed to vote more than once.
9) **Accountability**: attackers can be revealed and punished.

Some requirements conflict with each other (such as privacy and accountability), so it is very challenging to satisfy all of them.

3.2 Classifications of E-voting

Up to now, many e-voting models are discussed, and e-voting models are classified into 5 types [30]:

1) **Yes/No voting**: every voter votes for or against the candidates.
2) **1-out-of-L voting**: every voter votes for one candidate from the set of L candidates.
3) **K-out-of-L voting**: every voter votes for K different candidates from the set of L candidates.
4) **K-sort-out-of-L voting**: every voter votes for K ordered different candidates from the set of L candidates. The order of the selected candidates represents the importance.
5) **1-L-K voting**: the voter picks out a subset of L candidates, and then chooses K candidates form this set.

Note that above five types of voting are relative. In fact, 1-out-of-L voting is the generalization of the other four types of voting. For example, taking $L = 2$, we obtain Yes/No voting. If 1-out-of-L voting executes K times, we build K-out-of-L voting and K-sort-out-of-L voting. The 1-L-K voting can be considered as the combination of 1-out-of-L voting and K-out-of-L voting. Hence, without loss of generality, we only consider 1-out-of-L voting in our schemes.

3.3 Participants

In our scheme, there are $n + 3$ ($n > 2$) participants, including n voters, a trusted third-party authority, a vote counter and a registration center. The obligations of these participants are as follows:

1) n voters: each voter casts his/her vote, and computes the authentication value. When voters doubt the vote result which is computed and broadcasted by the vote counter, they can verify it using the authentication value. If the verification is unsuccessful, voters can report to the third-party authority who can investigate malicious participants.
2) The third-party authority: the third-party authority is responsible for initializing the e-voting scheme and investigating attackers. In setup phase, he/she prepares for the voting, and selects suitable parameters to ensure the security of the scheme. When attackers are detected in the verification phase, the third-party authority will initiate the investigation and find out them.
3) The registration center: the registration center is responsible for registering all applicants who want to join the e-voting. He/she verifies each applicant's personal information, and then, assigns a unique identification code for the applicant who passes the verification.
4) The vote counter: the vote counter computes the final vote result in the vote tallying phase, and then, broadcasts it.

In our e-voting scheme, we assume that the third-party authority and registration center are trusted.

3.4 Attack Model

The activities of attackers will threaten the security and privacy of the e-voting scheme. They intend to obtain the vote selections of the legal voters and modify the final vote result.

There are two types of attacks. One type is the Single Attack, involving only one attacker. In our scheme, third-party authority and the registration center are trusted,

so we only discuss the vote counter and the voter's single attack. The other type is Collusion Attack, which has multi-attackers. Since the vote counter's behaviors will be verified in verification phase, there is no need to consider it as a collusion attacker. In this paper, we only discuss the collusion attack launched by voters. In Section 5, we will discuss all types of malicious participants' attacks in detail.

4 The Proposed Scheme

In our scheme, the participants include a third-party authority $A_{uthority}$, a registration center $R_{egistrar}$, a vote counter C_{ounter} and n voters $V_1, V_2, \cdots, V_n$. $A_{uthority}$ and $R_{egistrar}$ are trusted. As we mentioned in Section 3.2, without loss of generality, we use 1-out-of-L voting in our scheme. We also need the following notations and parameters in Table 1.

Table 1: Parameters declaration

Symbol	Meaning
n	the number of voters
L	the number of candidates
C_i	the candidate i
c_i	the candidate value representing C_i
vn_i	the number of voters who vote for C_i
Id_i	$V_i's$ identification code
V_i	the voter i
v_i	$V_i's$ vote selection, $v_i \in \{c_1, c_2, \cdots, c_L\}$
m_i	$V_i's$ vote mask
M	the sum of all masks, $M = \sum_{i=1}^{n} m_i$
p_i	the corresponding prime of V_i
B_i	$V_i's$ ballot
$B_{i,j}$	the sub-ballot which V_i sends to V_j
MST_i	the masked sub-tally computed by V_i
MT	the masked tally, $MT = \sum_{i=1}^{n} B_i$
T	the tally, $T = \sum_{i=1}^{n} v_i$

In the following, we will describe our e-voting scheme. Note that all related technologies and vote process are transparent to voters.

4.1 Setup

In setup phase, $A_{uthority}$ prepares for the voting, and selects suitable parameters to ensure the security of the scheme. He/she works as follows:

1) $A_{uthority}$ generates a candidate value sequence $c_1, c_2, \cdots, c_L \in \mathbb{Z}$ which satisfies

$$c_i > n \times c_{i+1} \quad (i = 1, 2, \cdots, L-1) \tag{5}$$

If a voter chooses c_i, it means that this voter votes for candidate C_i.

2) $A_{uthority}$ selects suitable threshold according to the security requirement of the scheme.

3) $A_{uthority}$ generates a (t, n) Mignotte sequence $p_1, p_2, \cdots, p_n$, and this sequence satisfies Equation (1).

4) $A_{uthority}$ generates an integer mask sequence $m_1, m_2, \cdots, m_n$, and computes the sum of all masks $M = \sum_{i=1}^{n} m_i$.

In order to ensure the security and privacy of the scheme, the above parameters should satisfy the following conditions.

Condition 1. *In order to ensure attackers cannot reduce the guess scope of vote's selection v_i even if attackers obtain the voter's ballot B_i, the candidate value sequence and the mask sequence need to satisfy*

$$c_L + \min\{m_1, m_2, \cdots, m_n\} > c_1. \tag{6}$$

Condition 2. *In following phase, we need to recover masked tally MT and each voter's ballot B_i $(1 \le i \le n)$, so MT and B_i need to satisfy $\prod_{k=0}^{t-2} p_{n-i} < B_i, MT < \prod_{k=1}^{t} p_i$. Since $MT = \sum_{i=1}^{n} B_i$ and $B_i = v_i + m_i$, the candidate value sequence and the mask sequence need to satisfy*

$$\begin{cases} n \times c_1 + M < \prod_{i=1}^{t} p_i \\ \prod_{i=0}^{t-2} p_{n-i} < c_L + \min\{m_1, m_2, \cdots, m_n\} \end{cases} \tag{7}$$

4.2 Registration

Each voter sends his/her personal information to $R_{egistrar}$. If the applicant's personal information is legitimate, $R_{egistrar}$ assigns a unique identification code to her/him. By the signature technology which is represented in Section 4.3, participants can know the corresponding relation between the sender and his/her information.

According to voters' registration information, $A_{uthority}$ sends mask m_i and prime p_i to voter V_i through secure communication channels for all $1 \le i \le n$. Then, $A_{uthority}$ broadcasts the sum of all masks M, candidate and candidate value pairs $< c_i, C_i >$ $(1 \le i \le L)$, and voter's identification code and corresponding prime pairs $< id_i, p_i >$ $(1 \le i \le n)$.

4.3 Signature

In our scheme, the data, which $A_{uthority}$ use to identify attackers in investigation phase, may be stored by malicious participants. Thus, we use signature technique to ensure the reliability of the data and their source. In this way, senders cannot deny the data they sent, and $A_{uthority}$ can

easily detect the malicious data provider who tamper the data.

Some signature technologies can be used in our scheme to create the corresponding relations between the sender and his/her information. For example, we can use RSA signature technology which works as follows: each participant will generate he/her private key and public key and send public key to $R_{egistrar}$. After getting these public keys, $R_{egistrar}$ will broadcast the corresponding relation between participants' identification codes and their public key. If a participant wants to send a message, he/she will encrypt this message and his/her identification code using private key. After receiving the message, the receiver can decrypt the message using the sender's public key and verify the identity of the sender. In this way, the corresponding relation between the sender and his/her information can be built up. Here, we will not describe the details of RSA signature technology.

4.4 Vote

Following voter $V_i's$ own will, he/she chooses a vote selection $v_i \in \{c_1, c_2, \cdots, c_L\}$ and forms ballot $B_i = v_i + m_i$. For example, if V_i wants to vote for candidate C_k $(1 \leq k \leq L)$, he/she can choose $v_i = c_k$. Then, V_i computes each sub-ballot $B_{i,j} = B_i \bmod p_j$ $(1 \leq j \leq n, j \neq i)$, and send it to corresponding voter V_j.

After receiving all sub-ballot $B_{j,i}$ from voter V_j $(1 \leq j \leq n$ and $j \neq i)$, voter V_i checks the number of each $B_{j,i}$. If a voter sends his/her sub-ballot more than once and his/her sub-ballots are different, voter V_i will report to $A_{uthority}$ and use the last $B_{j,i}$. Then, voter V_i computes the masked sub-tally

$$MST_i = \sum_{j=1}^{n} B_{j,i} \bmod p_i, \tag{8}$$

and then, sends it to C_{ounter}.

4.5 Vote Tallying

In this phase, C_{ounter} randomly selects t voters $V_{i_1}, V_{i_2}, \cdots, V_{i_t}$ $(1 \leq i_1 < i_2 < \cdots < i_t \leq n)$, and then, uses their masked sub-tallies $MST_{i_1}, MST_{i_2}, \cdots, MST_{i_t}$ to build the system of congruences, such that

$$\begin{cases} MST_{i_1} \equiv MT \bmod p_{i_1} \\ MST_{i_2} \equiv MT \bmod p_{i_2} \\ \qquad \vdots \\ MST_{i_t} \equiv MT \bmod p_{i_t} \end{cases} \tag{9}$$

where $MT = \sum_{i=1}^{n} B_i$ and, for all $j \in \{i_1, i_2, \cdots, i_t\}$,

$$MST_j = \sum_{i=1}^{n} (B_i \bmod p_j) \bmod p_j. \tag{10}$$

C_{ounter} computes masked tally MT, using the general variant of the Chinese remainder theorem (see details in Section 1). Then, tally T, which can be described as $T = \sum_{i=1}^{L} (vn_i \times c_i)$, can be computed as $T = \sum_{i=1}^{n} v_i = \sum_{i=1}^{n} (B_i - m_i) = MT - M$. Let $r_0 = T$, the vn_i, which presents the number of the voters who vote for candidate C_i, can be computed as

$$\begin{cases} vn_i = \left\lfloor \frac{r_{i-1}}{c_i} \right\rfloor \\ r_i = r_{i-1} \bmod c_i \end{cases} (i = 1, 2, \cdots, L). \tag{11}$$

After recovering the final vote result, C_{ounter} broadcasts the masked tally MT and the final vote result $< vn_i, C_i >$ $(1 \leq i \leq L)$.

4.6 Verification and Investigation

In this section, we introduce our verification and investigation methods.

Verification A: If voter V_i doubt the vote result, he/she can verify masked tally MT which is computed and broadcasted by C_{ounter} as

$$MST_i = MT \bmod p_i. \tag{12}$$

If $MST_i \neq MT \bmod p_i$, verifier V_i will report to $A_{uthority}$ who can investigate C_{ounter}'s misbehavior. The investigation steps are described as follows:

1) Using Equation (12), $A_{uthority}$ verifies all masked sub-tallies $MST_1, MST_2, \cdots, MST_n$. If the number of voters whose masked sub-tallies satisfy Equation (12) is less than t, $A_{uthority}$ will know the masked tally MT was forged by C_{ounter}, and then, he/she broadcast C_{ounter}'s misbehaviors and finish the investigation.
2) $A_{uthority}$ selects t voters $V_{j_1}, V_{j_2}, \cdots, V_{j_t}$ whose masked sub-tallies satisfy Equation (12), and gets all voters' sub-ballots from these voters. Then, all voters' ballots can be recovered. For example, voter $V_i's$ ballot B_i can be recovered, using the general variant of the Chinese Remainder Theorem. The system of congruences can be built as

$$\begin{cases} B_{i,j_1} \equiv B_i \bmod p_{j_1} \\ B_{i,j_2} \equiv B_i \bmod p_{j_2} \\ \qquad \vdots \\ B_{i,j_t} \equiv B_i \bmod p_{j_t} \end{cases} \tag{13}$$

3) $A_{uthority}$ computes vote selection $v_i = B_i - m_i$, for all $1 \leq i \leq n$, and then he/she can find all attackers by recovering all phases of the e-voting (see details in Section 5.1).

Verification B: After verifying masked tally MT, voter V_i can verify vn_i, the number of the voters who vote for candidate C_i, using Equation (11). If the number is different from the number broadcasted by C_{ounter}, voter will report to $A_{uthority}$. Then $A_{uthority}$ will check the vote result. If C_{ounter} forged the vote result, $A_{uthority}$ will detect it.

5 Discussions

5.1 Correctness Analysis

In this section, we discuss the correctness of our scheme. By Proposition (1), we know that if all voters follow the vote rules, C_{ounter} can count the right vote result which reflects voters' true will.

Proposition 1. *If all voters follow the vote rules, C_{ounter} can count the right vote result which reflects voters' true will.*

Proof. Without loss of generality, in vote tallying phase, assume that C_{ounter} selects t voters $V_1, V_2, \cdots, V_t$ and uses their masked sub-tallies $MST_1, MST_2, \cdots, MST_t$ to recover masked tally MT. Since, for all $1 \leq i \leq t$,

$$\begin{aligned} MST_i &= \sum_{j=1}^{n} B_{j,i} \bmod p_i \\ &= \sum_{j=1}^{n} (B_j \bmod p_i) \bmod p_i \\ &= \sum_{j=1}^{n} B_j \bmod p_i \\ &= MT \bmod p_i \end{aligned} \tag{14}$$

C_{ounter} can build the system of congruences

$$\begin{cases} MST_1 \equiv MT \bmod p_1 \\ MST_2 \equiv MT \bmod p_2 \\ \quad\vdots \\ MST_t \equiv MT \bmod p_t \end{cases} \tag{15}$$

According to Condition 2, i.e., $\prod_{i=0}^{t-2} p_{n-i} < MT < \prod_{i=1}^{t} p_i$, C_{ounter} can recover MT by the general variant of the Chinese remainder theorem.

Tally T is the sum of all voters' selection v_i $(i = 1, 2, \cdots, n)$, so it can be computed as

$$T = \sum_{i=1}^{n} v_i = \sum_{i=1}^{n} (B_i - m_i) = MT - M \tag{16}$$

In addition, T can be described as

$$T = \sum_{j=1}^{n} v_j = \sum_{i=1}^{L} (vn_i \times c_i), \tag{17}$$

where $v_j \in \{c_1, c_2, \cdots, c_L\}$ and $\sum_{i=1}^{L} vn_i = n$. For all $1 \leq i \leq L-1$, we know that $c_i > n \times c_{i+1}$ and $0 \leq vn_i < n$, such that

$$c_i > \sum_{j=i+1}^{L} (vn_j \times c_j). \tag{18}$$

Thus, for all $1 \leq i \leq L$, if we set

$$\begin{cases} r_0 = T = \sum_{j=1}^{L} (vn_j \times c_j) \\ r_i = r_{i-1} \bmod c_i = \sum_{j=i+1}^{L} (vn_j \times c_j) \end{cases} \tag{19}$$

vn_i can be computed as follows:

$$\begin{aligned} \left\lfloor \frac{r_{i-1}}{c_i} \right\rfloor &= \left\lfloor \frac{\sum_{j=i}^{L} (vn_j \times c_j)}{c_i} \right\rfloor \\ &= vn_i + \left\lfloor \frac{\sum_{j=i+1}^{L} (vn_j \times c_j)}{c_i} \right\rfloor \\ &= vn_i. \end{aligned} \tag{20}$$

Obviously, in our scheme, if all voters follow the vote rules, C_{ounter} can compute the right vote result which reflects voters' true will. □

5.2 Security Analysis

In the e-voting scheme, malicious participants may attack the system. Here, we will analyze this problem in detail according to the attack models mentioned above.

5.2.1 Single Attack

According to participants' property, there are two types of single attacker, i.e., C_{ounter} and voter. We will analyze their misbehavior as follows:

1) Malicious C_{ounter}.

   C_{ounter} engages in two kinds of misbehavior: one is vote result cheating, the other is privacy stealing.

   Vote result cheating: if C_{ounter} forges masked tally and the final vote result to cheat voters, it will be discovered by voters in the verification phase.

   Privacy stealing: even if C_{ounter} wants to acquire voter $V_i's$ $(1 \leq i \leq n)$ vote selection v_i by collecting related information, he/she cannot recover voter $V_i's$ ballot B_i from prime number p_i, masked sub-tally MST_i and masked tally MT.

2) Malicious voter.

   Assume that voter V_i is malicious voter, whose attacks can be involved in the following attack cases:

   Case A. Using illegal ballots IB_i;

   Case B. Voting more than once;

   Case C. Sending different sub-ballots to other voters;

   Case D. Sending illegal masked sub-tally $IMST_i$ to C_{ounter};

   Case E. Trying to obtain legal voters' vote selections (such as voter $V_j's$ vote selection v_j).

   Case A, B, C and D can influence correctness and Case E can influence privacy. The security analysis of single malicious voter's attack is as follows:

   Case A. In the investigation phase, $A_{uthority}$ can recover voter $V_i's$ ballot B_i by solving the system of Congruences (13). Then, vote selection v_i is computed as $v_i = B_i - m_i$. If

$v_i \notin \{c_1, c_2, \cdots, c_L\}$, voter $V_i's$ misbehavior will be detected.

Case B. $R_{egistrar}$ will assign a unique identification code Id_i for voter V_i. According to Section 4.2, participants know the corresponding relation between the sender and his/her information. If voter V_i vote a new ballot $B_{i_{\text{new}}}$ and $B_{i_{\text{new}}} \neq B_i$, the information receivers will detect this misbehavior.

Case C. If voter V_i sends different sub-ballots to other voters, $A_{uthority}$ will recover an illegal ballot IB_i in the investigation phase by solving the system of Congruences (13). Thus, this misbehavior will be detected as in Case A..

Case D. According to the definition of threshold secret sharing, C_{ounter} just needs t honest voters to recover the masked tally MT. If only voter V_i sends an illegal masked sub-tally $IMST_i$, C_{ounter} can still recover MT. Moreover, $A_{uthority}$ can check $IMST_i$ in the investigation phase using Equation (12). If $IMST_i \neq MT \bmod p_i$, $A_{uthority}$ will detect voter $V_i's$ attack.

Case E. Voter V_i only gets voter $V_j's$ sub-ballot $B_{j,i}$, which is computed as $B_{j,i} = (v_j + m_j) \bmod p_i$, so he/she cannot compute ballot B_j. Even Voter V_i obtain ballot B_j, he/she also cannot recover voter $V_j's$ vote selection v_j without mask m_j which only $A_{uthority}$ and voter V_j know.

The above analysis shows that the attacks, launched by a single malicious participant, can be resisted in our e-voting scheme.

5.2.2 Collusion Attack

Since C_{ounter}'s behaviors will be verified in verification phase, there is no need to consider it as a collusion attacker. In this paper, we only discuss the collusion attack launched by voters.

Collusion voters' attacks can be involved in the following attack cases: A. Modifying the vote result; B. Obtaining legal voters' vote selection (such as voter $V_i's$ vote selection v_i). We analyze the above two cases next.

Case A: In the setup phase, $A_{uthority}$ selects proper threshold t. Generally, we set $t \geqslant \lceil (n+1)/2 \rceil$. If the number of collusion voters is more than or equals to t, they can win the voting and this collusion attack is meaningless. On the other hand, if the number of collusion voters is less than t, they cannot forge enough masked sub-tallies to cheat C_{ounter}. C_{ounter} will recover illegal masked tally IMT which cannot pass the verification phase. Then this collusion attack will be detected by $A_{uthority}$ in the investigation phase.

Case B: If the number of collusion voters is less than t, legal voter $V_i's$ ballot B_i will not be recovered. Otherwise, there will be a situation in which t voters win the voting and they still want to know voter $V_i's$ vote selection v_i. In this situation, ballot B_i can be recovered by solving the system of Congruences (13). Since $B_i = v_i + m_i$, vote selection v_i still cannot be computed without mask m_i which is only known to $A_{uthority}$ and voter V_i. Moreover, according to Condition 1, they also cannot reduce the guess scope of vote selection v_i, even if ballot B_i was obtained (see Section 4.1).

5.3 Features Analysis

In this section, we will analyze the features of our scheme according to the requirements in Section 3.1.

Legality: $R_{egistrar}$ verifies voters' personal information and assigns a unique identification code to each legal voter, so illegal voters cannot be involved in our scheme.

Correctness: If each voter is honest and strictly executes our scheme, C_{ounter} can recover the correct masked tally MT by solving the system of Congruences (9). Then, C_{ounter} can compute the final vote result that reflects voters' true will, using Equation (11).

Privacy: Protecting the privacy of voters' selections is one of the most important security requirements. By using the threshold secret sharing technology and the mask codes, the content of vote selection in our scheme is hidden to ensure privacy. According to the analysis in Section 5.2, any person cannot obtain voter' selection without the corresponding mask which is only known to this voter and $A_{uthority}$, and $A_{uthority}$ cannot recover any voter' selection unless t or more voters agree either. Obviously, our scheme can protect the privacy of voters' selections very well.

Robustness: From the analysis in Section 5.2, we know our scheme can resist all attacks launched by voters and the counter.

Verifiability: In the verification phase, voters can verify masked tally MT and the final vote result, with their own information (see details in Section 4.6).

Fairness: In the process of voting, each voter only gets other voters' sub-ballots (for example, voter V_j gets voter $V_i's$ sub-ballot $B_{i,j}$ which is computed as $B_{i,j} = B_i \bmod p_j$), so he/she cannot recover the final vote result using these information until C_{ounter} broadcasts it.

Uniqueness: In Section 4.3, the corresponding relation between voters' information and their identification code is established. If the voter casts his/her vote more than once, it can be detected in the vote phase.

Table 2: Computation complexities of all phases

Phases	Voter	Counter	Authority	Registrar
Setup	-	-	$O(n)$	-
Registration	-	-	-	$O(n)$
Vote	$O(n)$	-	-	-
Vote tallying	-	$O(n)$	-	-
Verification	$O(1)$	-	-	-
C_{ounter} *investigation*	-	-	$O(n)$	-
Voter investigation	-	-	$O(n)$	-
Sum	$O(n)$	$O(n)$	$O(n)$	$O(n)$

Transparency: In our scheme, we need participants to supervise their behaviors mutually, so that all working mechanisms and voting process are transparent to all participants in the whole process.

Accountability: According to the security analysis in Section 5.2, $A_{uthority}$ can find attackers and then reduce the damage. At the same time, $A_{uthority}$'s existence can also deter some attackers.

From the above analysis, our scheme obviously satisfies all the requirements and balances the conflict between privacy and accountability.

5.4 Computation Complexity Analysis

In order to analyze the computation cost of our scheme more clearly, we analyze the computation cost of voting and the computation cost of signature in this section, independently.

5.4.1 Computation Cost Analysis of Voting

We list the computation complexities of all work phases in Table 2.

In setup phase, $A_{uthority}$ should prepare for e-voting and select suitable parameters. $A_{uthority}$'s computation complexity is $O(n)$.

In registration phase, $R_{egistrar}$ should confirm whether the information of each applicant is legitimate, and then, assign a unique identification code to the legal applicant. In this phase, $R_{egistrar}$'s computation complexity is $O(n)$.

In vote phase, voters need to form the ballot and compute the masked sub-tally. Each voter's computation complexity is $O(n)$, so the computation complexity of n voters is $O(n^2)$.

In vote count phase, C_{ounter} need to recover the masked tally MT and compute the vote result. The computation complexity of recovering MT which need to solve the system of t congruence equations is $O(n)$, and the computation complexity of computing the vote result is $O(L)$ $(L < n)$, so the computation complexity of vote tallying phase is $O(n)$.

In verification phase, if the voter doubt the vote counter, he/she can verify the masked tally and the vote result. The verifier's computation complexity is $O(1)$.

In investigation phase, the computation complexity of investigating the vote counter is $O(n)$, and the computation complexity of investigating a voter, by recovering this voter's ballot, is $O(n)$. In worst situation, $A_{uthority}$ need to investigate n voters and the computation complexity is $O(n^2)$. In reality, the majority of participants are honest and follow the rules of the vote scheme, so $A_{uthority}$ only need to investigate a small number of voters.

Since the computation of masked sub-tally which has the highest computation complexity is allocated to n voters, each voter's computation complexity is $O(n)$. C_{ounter} and $A_{uthority}$'s computation complexity also is $O(n)$. In our scheme, each participant's computation load is balanced, which effectively avoid overload of the vote counter.

5.4.2 Computation Cost Analysis of Signature

In our scheme, voter's sub-ballot is stored by other voters. When $A_{uthority}$ needs to recover a certain voter's vote selection, he or she needs not less than t voters to provide their stored information about this voter. Since these information providers may be malicious, we need to guarantee information is reliable and not tampered. In our scheme, we use signature technology to guarantee information reliability and validity. Table 3 lists all participants' computation cost of signature and verification, which also reflect the communication situation between participants.

Since multiple signature techniques can be used in our scheme, we use T_s to represent the computation cost of signing a message, use T_v to represent the computation cost of verifying a message, and use T to represent the computation cost of signing and verifying a message. By Table 3, we know the total computation complexity of signature is $O(n^2T)$. The major computation cost generated by signature is in vote phase, because every voter needs to send her/his sub-ballot to other $n-1$ voters. In addition, for a single participant, signature computation complexity is $O(nT)$, which is acceptable.

In addition, we can avoid the computation cost of signature through requiring voters send backup information to trusted entity. For example, in our scheme, voters can send information to designated trusted entity (e.g., $A_{uthority}$). Since the trusted entity can ensure the data and their sources are valid and correct, $A_{uthority}$ can use them to identify attackers. Note that, when trusted entity receives the backup information, he/she should check data consistency with the information receiver.

5.5 Comparisons

In this section, we compare the functionality and computation complexity with related schemes.

5.5.1 Functionality Comparisons

Table 4 compares our scheme's functionality with Cramer et al.'s scheme [10], Iftene's scheme [17] and Li et al.'s scheme [21]. We explain Table 4 as follows:

1) All four schemes have verification function. In Cramer et al.'s and Iftene's schemes, multiple counters compute vote result, and then, implement mutual verification in order to verify the correctness of the vote result. In Li et al.'s scheme, after vote result is announced, the voter can verify her/his own vote, but cannot verify the whole vote result. Thus, in their schemes, voters may doubt the correctness of the vote result. In our scheme, any voter can verify whole vote result, which greatly improve vote result's trustworthiness.

2) Iftene's scheme lack consideration of the impact of attackers, so it cannot resist attacks. Cramer et al.'s and Li et al.'s schemes can only resist the attacks launched by voters. Our scheme can resist the attacks launched by voters and the counter. Thus, our scheme is more secure and feasible.

3) All four schemes can protect the privacy of voters' vote selection, and only Li et al.'s and our schemes can identify attackers by recovering voters' selections. But, in our scheme, $A_{uthority}$ cannot obtain any voter's selection unless not less than t voters agree, which can better balance the privacy and the accountability.

4) Compared with three other schemes, our scheme can balance the participants' computation overload, which can effectively avoid overloading the central counter.

5.5.2 Computation Complexity Comparisons

Table 5 compares computation complexity of our scheme with Cramer et al.'s scheme [10], Iftene's scheme [17] and Li et al.'s scheme [21].

Our scheme and Li et al.'s use some techniques (e.g., signature technique) to ensure the reliability of the data and their source, which authority use to identify attackers. Thus, our scheme and Li et al.'s have higher computation cost than Cramer et al.'s and Iftene's schemes. But, in order to ensure the robustness and accountability of the e-voting scheme, it's inevitable. In addition, our scheme distributes the computation burden to all of participants, thus improve computational efficiency, and avoid overloading the counter.

6 Conclusions

In this paper, we propose an e-voting scheme which allows voters to verify the final vote result independently and balances the conflict of interest between voters and the vote counter. Moreover, the scheme is secure because it can resist attacks effectively. In this scheme, we suppose that the third-party authority and the registration center are credible. However, they might be non-credible in reality. Therefore, we plan to design a new mechanism which can avoid the supposition of those credible participants in the future.

Acknowledgments

A 5-page preliminary version of this paper appeared in the International Conference on Communication Technology, pp. 304-308, October, 2015. In addition, this paper is supported by the Nature Science Foundation of China under grant No. 61272173, 91315302, 61401060, 61501080, 61572095, the general program of Liaoning Provincial Department of Education Science Research under grants L2014017, and the Fundamental Research Funds for the Central Universities under grant No. DUT16QY09.

References

[1] G. Blakley, "Safeguarding cryptographic keys," in *Proceedings of the National Computer Conference*, pp. 313–317, Montvale: NCC, 1979.

[2] M. Bonanome, V. Buzek, M. Hillery, , and M. Ziman, "Toward protocols for quantum-ensured privacy and secure voting," *Physical Review A*, vol. 84, no. 2, pp. 022331, 2011.

[3] D. Chaum, "Blind signatures for untraceable payments," in *Advances in Cryptology*, pp. 199–203, Santa Barbara, CA, USA, 1983.

[4] D. Chaum, P. A. Ryan, and S. Schneider, "A practical voter-verifiable election scheme," in *Computer Security (ESORICS'05)*, pp. 118–139, Milan, Italy, 2005.

[5] D. L. Chaum, "Untraceable electronic mail, return add-resses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[6] X. Chen, Q. Wu, F. Zhang, and H. Tian et al., "New receipt-free voting scheme using double-trapdoor commitment," *Information Sciences*, vol. 181, no. 2, pp. 1493–1502, 2011.

Table 3: Computation costs of signature

Phases	**Voter**	**Counter**	**Authority**	**Registrar**	**Sum**
Setup	T_v	-	nT_s	-	nT
Registration	T	-	-	nT	$2nT$
Vote	$T_s + (n-1)T$	nT_v	-	-	n^2T
Vote tallying	T_v	-	nT_s	-	nT
Verification	-	-	-	-	-
C_{ounter} *investigation*	T_s	-	nT_v	-	nT
Voter investigation	T_s	-	tT_v	-	tT
Sum	$(n+2)T + T_s$	nT_v	$n(T+T_s) + tT_v$	nT	$(n^2+5n+t)T$

[1] T_s: The computation cost of signing a message
[2] T_v: The computation cost of verifying a message
[3] T: The computation cost of signing and verifying a message

Table 4: Functionality comparisons between our scheme and related schemes

Functionalities	**Cramer et al. [10]**	**Iftene [17]**	**Li et al. [21]**	**Our scheme**
Multiple counter	Yes	Yes	Yes	No
Trusted counter	Yes	Yes	Yes	No
Voting type	Yes/No	Yes/No	All	All
Verification function	Yes	Yes	Yes	Yes
Verifying vote result	Yes	Yes	No	Yes
Verifying own vote	No	No	Yes	No
Verifying by counter	Yes	Yes	No	No
Verifying by voter	No	No	Yes	Yes
Protecting privacy	Yes	Yes	Yes	Yes
Resisting attack	No	No	Yes	Yes
Identifying attacker	No	No	Yes	Yes
Robustness	No	No	Yes	Yes
Calculated balance	No	No	No	Yes

Table 5: Computation complexity comparisons between our scheme and related schemes

Phases	**Cramer et al. [10]**	**Iftene [17]**	**Li et al. [21]**	**Our scheme**
Setup	Null	$O(n)$	$O(n^2D)$	$O(nT)$
Registration	Null	Null	Null	$O(nT)$
Vote	$O(n^2\log^2 n)$	$O(n^2)$	$O(n^3+n^2D)$	$O(n^2T)$
Vote tallying	$O(n^2\log^2 n)$	$O(n)$	$O(n^2D)$	$O(nT)$
Verification	$O(n)$	$O(n)$	$O(n)$	$O(n)$
C_{ounter} *investigation*	Null	Null	Null	$O(nT)$
Voter investigation	Null	Null	$O(n^2+nD)$	$O(nT)$
Sum	$O(n^2\log^2 n)$	$O(n^2)$	$O(n^3+n^2D)$	$O(n^2T)$

[1] Null: don't have this phase
[2] T: the computation cost of message signing or verifying
[3] D: the computation cost of message backup or verification

[7] J. Clark and U. Hengartner, "Selections: Internet voting with over-the-shoulder coercion-resistance," in *Financial Cryptography and Data Security*, pp. 47–61, Gros Islet, ST Lucia, 2012.

[8] M. R. Clarkson, S. Chong, and A. C. Myers, *Civitas: A Secure Voting System*, Technical Report, Cornell University, TR 2007-2081, May 2007.

[9] J. D. Cohen and M. J. Fischer, *A Robust and Verifiable Cryptographically Secure Election Scheme*, Technical Report, Yale University, YALEU/DCS/TR-416, July 1985.

[10] R. Cramer, M. Franklin, B. Schoenmakers, and M. Yung, "Multi-authority secret-ballot elections with linear work," in *International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT'96)*, pp. 72–83, Saragossa, Spain, 1996.

[11] R. Cramer, R. Gennaro, and B. Schoenmakers, "A secure and optimally efficient multi-authority election scheme," *European Transactions on Telecommunications*, vol. 8, no. 5, pp. 481–490, 1997.

[12] L. F. Cranor and R. K. Cytron, "Sensus: a security-conscious electronic polling system for the internet," in *System Sciences*, pp. 561–570, Wailea, HI, 1997.

[13] X. Dong, "A multi-secret sharing scheme based on the CRT and RSA," *International Journal of Electronics and Information Engineering*, vol. 2, no. 1, pp. 47–51, 2015.

[14] B. Feng, C. Guo, M. Li, and Z. Wang, "A novel proactive multi-secret sharing scheme," *International Journal of Network Security*, vol. 17, no. 2, pp. 123–128, 2015.

[15] A. Fujioka, T. Okamoto, and K. Ohta, "A practical secret voting scheme for large scale elections," in *Advances in Cryptology (AUSCRYPT'92)*, pp. 244–251, Gold Coast, Qld., Australia, 1993.

[16] D. A. Gritzalis, *Secure Electronic Voting*, USA: Springer US, 2003.

[17] S. Iftene, "General secret sharing based on the chinese remainder theorem with applications in e-voting," *Electronic Notes in Theoretical Computer Science*, vol. 186, no. 1, pp. 67–84, Springer, 2007.

[18] C. C. Lee, T. Y. Chen, S. C. Lin, and M. S. Hwang, "A new proxy electronic voting scheme based on proxy signatures," in *Lecture Notes in Electrical Engineering*, pp. 3–12, Dordrecht, Netherlands, 2012.

[19] C. T. Li and M. S. Hwang, "Security enhancement of chang-lee anonymous e-voting scheme," *International Journal of Smart Home*, vol. 6, no. 2, pp. 45–52, 2012.

[20] C. T. Li and M. S. Hwang, "A secure and anonymous electronic voting scheme based on key exchange protocol," *International Journal of Security and Its Applications*, vol. 7, no. 1, pp. 59–70, 2013.

[21] H. Li, Y. Sui, W. Peng, X. Zou, and F. Li, "A viewable e-voting scheme for environments with conflict of interest," in *2013 IEEE Conference on Communications and Network Security (CNS'13)*, pp. 251–259, Washington, DC, 2013.

[22] Y. J. Liu and C. C. Chang, "An integratable verifiable secret sharing mechanism," *International Journal of Network Security*, vol. 18, no. 4, pp. 617–624, 2016.

[23] M. Mignotte, "How to share a secret," in *Proceedings of the Workshop on Crytography*, pp. 371–375, Burg Feuerstein, Germany, 1983.

[24] C. Park, K. Itoh, and K. Kurosawa, "Efficient anonymous channel and all/nothing election scheme," in *Advances in Cryptology (EUROCRYPT'93)*, pp. 248–259, Lofthus, Norway, 1994.

[25] K. Peng, "A general and efficient countermeasure to relation attacks in mix-based e-voting," *International Journal of Information Security*, vol. 10, no. 1, pp. 49–60, 2011.

[26] K. Peng and F. Bao, "Efficient vote validity check in homomorphic electronic voting," in *11th International Conference on Information Security and Cryptology*, pp. 202–217, Seoul, South Korea, 2008.

[27] Q. Peng and Y. L. Tian, "A publicly verifiable secret sharing scheme based on multilinear diffie-hellman assumption," *International Journal of Network Security*, vol. 18, no. 6, pp. 1192–1200, 2016.

[28] A. A. Philip, S. A. Simon, and A. Oluremi, "A receipt-free multi-authority e-voting system," *International Journal of Computer Applications*, vol. 30, no. 6, pp. 15–23, 2011.

[29] M. J. Radwin, "An untraceable, universally verifiable voting scheme," in *Seminar in Cryptology*, pp. 829–834, Nagercoil, India, 1995.

[30] Z. Rjaskova, "Electronic voting schemes," in *Diplomova Praca*, pp. 70–76, Bratislava, Slovakia, 2002.

[31] L. Rura, B. Issac, and M. K. Haldar, "Secure electronic voting system based on image steganography," in *2011 IEEE Conference on Open Systems (ICOS'11)*, pp. 80–85, Langkawi, Malaysia, 2011.

[32] K. Sampigethaya and R. Poovendran, "A framework and taxonomy for comparison of electronic voting schemes," *Computers and Security*, vol. 25, no. 2, pp. 137–153, 2006.

[33] B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," in *Advances in Cryptology (CRYPTO'99)*, pp. 148–164, Santa Barbara, CA, USA, 1999.

[34] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[35] F. Shirazi, S. Neumann, I. Ciolacu, and M. Volkamer, "Robust electronic voting: Introducing robustness in civitas," in *2011 International Workshop on Requirements Engineering for Electronic Voting Systems*, pp. 47–55, Trento, Italy, 2011.

[36] C. Staff, "Seven principles for secure e-voting," *Communications of the ACM*, vol. 52, no. 2, pp. 8–9, 2009.

Lifeng Yuan received the B.S. degree in Computer Science and Technology from Ningbo University in 2006 and the M.S. degree in software engineering from Dalian University of Technology in 2009. He is currently a Ph.D candiadate in Dalian University of Technology. His

current research interests include Secret Sharing, Data Hiding, Network and Information Security, and Image Processing.

Mingchu Li received the B.S. degree in mathematics, Jiangxi Normal University and the M.S. degree in applied science, University of Science and Technology Beijing in 1983 and 1989, respectively. He worked for University of Science and Technology Beijing in the capacity of associate professor from 1989 to 1994. He received his doctorate in Mathematics, University of Toronto in 1997. He was engaged in research and development on information security at Longview Solution Inc, Compuware Inc. from 1997 to 2002. From 2002, he worked for School of Software of Tianjin University as a full professor, and from 2004 to now, he worked for School of Software Technology of Dalian University of Technology as a full Professor, Ph.D. supervisor, and vice dean. His main research interests include theoretical computer science and cryptography.

Cheng Guo received the B.S. degree in computer science from Xi'an University of Architecture and Technology in 2002. He received the M.S. degree in 2006 and his Ph.D in computer application and technology, in 2009, both from the Dalian University of Technology, Dalian, China. From July 2010 to July 2012, he was a post doc in the Department of Computer Science at the National Tsing Hua University, Hsinchu, Taiwan. Since 2013, he has been an associate professor in the School of Software Technology at the Dalian University of Technology. His current research interests include information security and cryptology.

Wei-Tong Hu received the B.S. degree in 2008 and Ph.D. degree in 2015, both from the School of Software Technology, Dalian University of Technology, Dalian, China. Since 2016, he has been a lecturer in the School of Computer Science and Technology at Hangzhou Dianzi University. His current research interests include Secret Sharing, Data Hiding, Network and Information Security, and Image Processing.

Zhi-Hui Wang was born in Inner Mongolia in 1982. She received her B.S. degree in software engineering from the North Eastern University, Shenyang in 2004, M.S. degree in software engineering from Dalian University of Technology (DUT), Dalian in 2007, and Ph.D. degree in computer software and theory from DUT in 2010. Since 2014, she has been an associated professor in the School of Software Technology at the Dalian University of Technology. Her current research interests include information hiding and image processing.