doi:10.3772/j.issn.1006-6748.2020.01.004

End-to-end verifiable electronic voting scheme of blockchain based on random linear block code①

Liu Ting(刘 霆)[*,**,***], Cui Zhe②[*,**], Pu Hongquan[*,**,***], Peng Xingyi[*,**]
([*] Chengdu Institute of Computer Applications, Chinese Academy of Sciences, Chengdu 610041, P. R. China)
([**] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049, P. R. China)
([***] Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, Nanning 530006, P. R. China)

Abstract

Blockchain is an emerging decentralized technology of electronic voting. The current main consensus protocols are not flexible enough to manage the distributed blockchain nodes to achieve high efficiency of consensus. For practical implementation, the consensus based on random linear block code (RLBC) is proposed and applied to blockchain voting scheme. Along with achieving the record correctness and consistency among all nodes, the consensus method indicates the active and inactive consensus nodes. This ability can assist the management of consensus nodes and restrain the generating of chain forks. To achieve end-to-end verifiability, cast-or-audit and randomized partial checking (RPC) are used in the proposed scheme. The voter can verify the high probability of correctness in ballot encryption and decryption. The experiments illustrate that the efficiency of proposed consensus is suitable for blockchain. The proposed electronic voting scheme is adapted to practical implementation of voting.

Key words: random linear block code (RLBC), electronic voting (e-voting), blockchain, consensus, end-to-end verifiable

0 Introduction

With the development and application of electronic technology, electronic voting (e-voting) has become an important method in elections[1]. Using machines to cast votes, count votes, and do others, the e-voting system accelerates the tally process and overcomes the problems introduced by human errors[2]. Researchers have applied many methods to improve the security and credibility of e-voting systems. Many systems encrypt the cast ballot to preserve voting privacy. Ballot auditing is implemented to assure the voter that the encryption indeed contains its opinion. Cast-or-audit is an auditing approach for voter that can still preserve the secrecy of ballots[3,4]. The voter inputs opinion into the system, and the system encrypts it. The voter can cast or audit the encryption to prevent itself from being cheated by the system. Mix-nets is a method to prevent tracing an encrypted vote back to its original form, which was proposed by Chaum[5]. Neff[6] realized universal verifiability using verifiable mix-nets. Groth[7] proposed verifiable mix-nets of homomorphic encryptions and its application in e-voting.

The emerging blockchain technology can make recorded information in it tamper-free. Some e-voting schemes apply blockchain to increase voting credibility. Lee et al.[8] proposed conducting e-voting by means of bitcoin transaction and designed a third-party qualification audit mechanism. Hsiao et al.[9] applied smart contract to achieve blockchain consensus of voting information and used a cryptography algorithm for information security. Zhao et al.[10] developed a voting system based on bitcoin with a mechanism incentivizing voting, which uses the zero-knowledge-proof of the vote commitment. Hjalmarsson et al.[11] proposed e-voting based on permissioned blockchain using consensus mechanism manipulated by administrators. Hanifatunnisa et al.[12] applied blockchain to reduce the risk of voting record database manipulation. So far, some blockchain voting systems have been developed, such as SecEVS[13], Follow My Vote[14], BroncoVote[15].

The verifiability of e-voting provides transparency to the voter and the public. Randomized partial chec-

① Supported by the National Natural Science Foundation of China (No. 61501064), Sichuan Technology Support Program (No. 2015GZ0088) and Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis (No. HCIC201502, HCIC201701).
② To whom correspondence should be addressed. E-mail: cuizhe@casit.com.cn
Received on Jan. 18, 2019

king (RPC) is method of verifiability for mix-nets based e-voting. Scantegrity scheme mixes options of voter and candidate IDs using mix-nets. Voter can verify that its vote was recorded correctly using randomized partial checking[16]. In Bingo scheme, this method verifies the encrypted voting options in the receipt of voter without ruining the voting privacy[17,18].

In real distributed application, the large scale blockchain e-voting has much more complex situation. The distributed consensus nodes of the voters' devices have different computation performance and network environment. It is hard to realize that most of all nodes perform the consensus operation efficiently. The continuity and stability of consensus operation are easy to be influenced by the voters' artificial reasons on voting device operations, such as cutting off the network, exiting the consensus program, etc. To implement the blockchain voting scheme practically, a new consensus method needs to be applied to settle the above problems.

This work proposes an electronic voting scheme based on blockchain with consensus of random linear block code (RLBC). The vote encryption correctness is proven by cast-or-audit, and decryption correctness is proven by randomized partial checking. The scheme satisfies end-to-end verifiability and achieves primary information security of e-voting.

The rest of this work is organized as follows. Section 1 proposes the preliminaries used in the construction of e-voting scheme. Section 2 describes blockchain consensus based on random linear block code. Section 3 describes the proposed scheme of e-voting on blockchain. Section 4 provides the security analysis of the proposed scheme. Section 5 presents the experiments. Finally, the conclusion is discussed in Section 6.

1 Preliminaries

1.1 The properties of the random matrix

Let $\boldsymbol{R}_{n\times k}(n > k)$ be a matrix over a Galois field $GF(2)$, in which elements are independent. The probability of an element in $\boldsymbol{R}$ being 0 is p, and being 1 is $1-p$, $p \in (0,1)$. $\boldsymbol{R}$ is defined as a random matrix when $p = 0.5$.

Theorem 1 Let $\boldsymbol{R}_{n\times k}(n > k)$ be a random matrix over $GF(2)$. The probability of $\boldsymbol{R}$ being column full rank is $P = \prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)$, $\delta = n-k$.

Proof Apply elementary row transformation t to the random matrix $\boldsymbol{R}$ to obtain the row simplest matrix $\boldsymbol{R}_1$. If each column of $\boldsymbol{R}_1$ has elements with value of 1, $\boldsymbol{R}$ must be a column full rank matrix. Perform inverse transformation of swapping 2 rows in t on $\boldsymbol{R}_1$ to obtain the matrix $\boldsymbol{R}_2$. In $\boldsymbol{R}_2$, the element that has a value of 1 is still in the same position of the original position in $\boldsymbol{R}$. Because $\boldsymbol{R}$ is a random matrix, the elements that have value of 1 in $\boldsymbol{R}_2$ are randomly positioned. Add the elements with value 1 into $\boldsymbol{R}_2$ column by column to prove that the probability of $\boldsymbol{R}_2$ being column full rank is $P = \prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)$. The proof is shown in the next paragraph.

Event 1 $\boldsymbol{R}_2(x,1)$ is an element with value of 1 in column 1 and random row position x of $\boldsymbol{R}_2$. The opposite of Event 1 is that all of the elements in column 1 of $\boldsymbol{R}_2$ are 0, and its probability is $\frac{1}{2^{\delta+k}}$. So the probability of Event 1 is $1-\frac{1}{2^{\delta+k}}$.

Event 2 $\boldsymbol{R}_2(y,2)(x \neq y)$ is an element with value of 1 in column 2 and random row position y of $\boldsymbol{R}_2$. The opposite of Event 2 is that all of the elements $\boldsymbol{R}_2(y,2)(x \neq y)$ in column 2 of $\boldsymbol{R}_2$ are 0, and its probability is $\frac{1}{2^{\delta+k-1}}$. So the probability of Event 2 is $1-\frac{1}{2^{\delta+k-1}}$. Other columns are analogous in turn. Each column in $\boldsymbol{R}_2$ has an element with value of 1 that means $\boldsymbol{R}_2$ is a matrix of full rank, and its probability is $P = \prod_{i=1}^{k}\left(1-\frac{1}{2^{\delta+i}}\right)$.

Because the elementary transformation does not change the rank of the matrix, $\boldsymbol{R}$ has the same probability of being full column rank as $\boldsymbol{R}_2$. Therefore, Theorem 1 is proved. This proof process was obtained from Ref. [19].

Theorem 2 The probability of random matrix $\boldsymbol{R}_{n\times k}(n > k)$ being full column rank is extremely close to 1 as δ increasing, where $\delta = n-k$.

Proof

$$S = 1-\frac{1}{2^{j}} \tag{1}$$

From Eq. (1), it can be observed that the value of S approaches to 1 as j increasing. The value of S is extremely close to 1 when $j \geqslant 20$, and the variation in S is illustrated in Fig. 1.

Moreover, it can be concluded that $S = \prod_{j=\delta+1}^{\delta+k}\left(1-\frac{1}{2^{j}}\right)$ is extremely close to 1 when $\delta \geqslant 20$ and $k \geqslant 0$.

According to Theorem 1, the probability of $\boldsymbol{R}$ being full column rank is

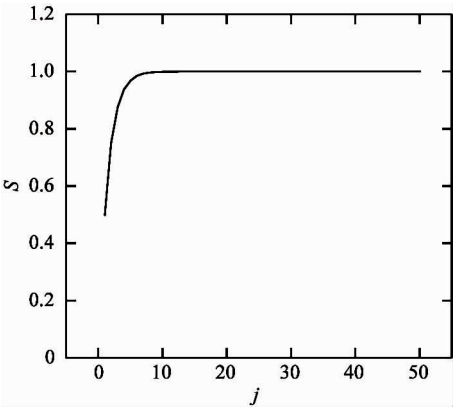

Fig. 1 Variation in S

$$P = \prod_{j=1}^{k}\left(1-\frac{1}{2^{\delta+j}}\right) = \prod_{j=\delta+1}^{\delta+k}\left(1-\frac{1}{2^{j}}\right) = \frac{\prod_{j=1}^{\delta+k}\left(1-\frac{1}{2^{j}}\right)}{\prod_{j=1}^{\delta}\left(1-\frac{1}{2^{j}}\right)}$$

Therefore, it can be concluded that P is extremely close to 1 when $\delta \geqslant 20$ and $k \geqslant 0$, and Theorem 2 is proved.

1.2 Random linear block code

On the linear block code, the coding matrix $\boldsymbol{G}_{n\times k}(n > k)$ is used to encode the information vector $\boldsymbol{v}_{k\times 1}$ and generate the code word $\boldsymbol{u}_{n\times 1}$, which is expressed as $\boldsymbol{G}\times\boldsymbol{v} = \boldsymbol{u}$. The encoding process of RLBC is as follows.

1) Calculate $m = n - k$.

2) Construct a random matrix $\boldsymbol{H}_1 = \boldsymbol{R}_{m\times k}$.

3) Construct a full rank random matrix $\boldsymbol{H}_2 = \boldsymbol{R}_{m\times m}$.

4) Set the parity-check matrix $\boldsymbol{H}^{\mathrm{T}} = [\boldsymbol{H}_1 \mid \boldsymbol{H}_2]$. $\boldsymbol{H}^{\mathrm{T}}$ has n columns and m rows.

5) From $\boldsymbol{H}^{\mathrm{T}}\times\boldsymbol{G} = 0$, set the coding matrix $\boldsymbol{G}_{n\times k} = \left[\frac{\boldsymbol{I}_{k\times k}}{\boldsymbol{H}_2^{-1}\boldsymbol{H}_1}\right]$ over $GF(2)$.

6) Use the formula $\boldsymbol{G}\times\boldsymbol{v} = \boldsymbol{u}$ to generate the code word $\boldsymbol{u}$.

Decoding of RLBC: the parity-check formula $\boldsymbol{H}^{\mathrm{T}}\times\boldsymbol{u} = 0$ is applied in decoding of RLBC. The first k elements of $\boldsymbol{u}$, that are equal to the elements in vector $\boldsymbol{v}$ at same positions, are obtained by solving the equations deduced from $\boldsymbol{H}^{\mathrm{T}}\times\boldsymbol{u} = 0$.

Because RLBC is erasure code, the information word $\boldsymbol{v}$ can be decoded from the elements in last $n - k$ of $\boldsymbol{u}$, which is not containing $\boldsymbol{v}$. From the parity-check formula, equations of decoding can be deduced. The set of decoding equations need to have more equations than unknown elements. The coefficient matrix of the decoding equation set also needs to be full column rank, the unknown elements can be calculated by solving the equations. According to the properties of the random matrix in Section 1.1, set the size of $\boldsymbol{H}^{\mathrm{T}}$ to get the full column rank of the decoding equations as follows.

Suppose that decoding of $\boldsymbol{v}$ needs at least $n - k - e$ elements in last $n - k$ of $\boldsymbol{u}$ as threshold and other e elements are not necessary. So there are $k + e$ elements to be calculated from the decoding equation set. To decode $\boldsymbol{v}$ successfully, the row number m of $\boldsymbol{H}^{\mathrm{T}}$ must be $k + e + \delta$ to ensure that the coefficients matrix of the decoding equation set is of full column rank when $\delta \geqslant 20$. Because $m = n - k$, the column number n of $\boldsymbol{H}^{\mathrm{T}}$ is set with $2k + e + \delta$. In this occasion, RLBC has a threshold $r = \left[\frac{n-k-e}{n-k}\right]$ of decoding elements in the last $n - k$ elements of the code word $\boldsymbol{u}$, which is not containing $\boldsymbol{v}$. From $n = 2k + e + \delta$, $r = \left[\frac{k+\delta}{k+e+\delta}\right]$ can be deduced. Given fixed k, threshold r can be set to be less than 50% with δ and e.

From $n = 2k + e + \delta$ and $r = \left[\frac{k+\delta}{k+e+\delta}\right]$, $n - k = \left[\frac{k+\delta}{r}\right]$ can be deduced. Maximum value of $n - k$ can be got by setting δ. When $r = 0.4$ and $k = 40$ which means the information word is 5 bytes, the relation between δ and $n - k$ is shown as Fig. 2.

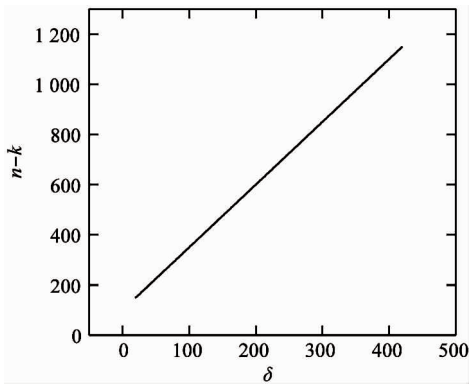

Fig. 2 Relation between δ and $n - k$

1.3 Blockchain

Blockchain was proposed by an anonymous scholar named 'Satoshi' in the digital currency paper of bitcoin in 2007[20]. Blockchain technology realizes trading bitcoins without the participation of a trust center. Blockchain is an open ledger which is stored and maintained together by various nodes. The transaction is shared and stored by each node of the whole network. The node records transaction information into a block body with the previous block head. Then the node

computes the hash values of this block body to get the block head. In this way, different blocks chain each other[21]. The construction of blockchain is shown in Fig. 3.

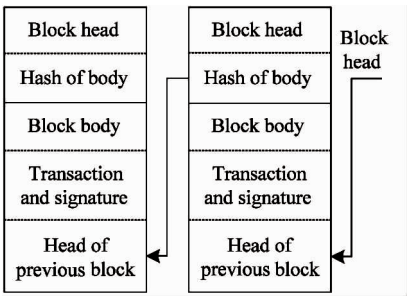

Fig. 3 Construction of blockchain

Agreeing on consecutive blocks among all nodes in blockchain network is called a consensus operation. The blockchain consensus mechanism is an algorithm of building trust and accessing right among different nodes. There are various consensus mechanisms of blockchain to enable distributed nodes to achieve block data validity and consistency efficiently. The practical Byzantine fault tolerance (PBFT), Paxos and Raft are important consensus protocols respectively tolerate 33%, 50% and 50% faulty nodes of the total[22-24].

1.4 Randomized partial checking

RPC is a method which checks part of shuffle and re-encryption results operated by mix-nets to ensure the correctness of the results[25]. It is manipulated by one or more servers and can implement any encryption scheme. In one server RPC, the server re-encrypts and shuffles the input information to the outputs. Then, the output becomes input of its own and the server repeats the operation.

The shuffle and re-encryption operation is repeated a few times. The server can produce strong evidence by revealing partial input and output relations. If one in p times of shuffle and re-encryption is revealed being true, a statement has one p-th probability of correctness. Finally, all encrypted information is decrypted. After RPC has been executed and the results are checked being correct a few times, the information decryption correctness can be proven of high probability. Because the information will not be traced back to its original form, RPC will not reveal the original information.

In this scheme, RPC proves that the decryption result is the same as the information encrypted in input. One server re-encrypts and shuffles the encrypted input information twice using different parameters. In the verification, the server works as the prover. The verifier randomly askes the server to reveal one of the re-encryption and shuffle operation details to verify the re-encryption correctness. The verification of the decryption by RPC is detailed as follows.

1) In this work, $E(b_i, w_i)$ denotes the encryption of b_i using parameter w_i. All $E(b_i, w_i)$ constitute a list E. At the beginning, the server publishes E on the bulletin board.

2) All the encryptions in E are re-encrypted with another random parameter w' by server, and $E(b_i, w_i)$ is transformed to $E(b_i, w_i + w')$. After being shuffled with an arrangement π', the re-encrypted results constitute a list E', which is published on the bulletin board by server.

3) The manipulation of server on $E(b_i, w_i)$ in the previous step is repeated to obtain $E(b_i, w_i + w' + w'')$ with another random parameter w'' and arrangement π''. The re-encrypted results constitute the list E'', and server publishes E''. Each encryption in E'' is decrypted to b_i and each $w_i + w' + w''$ is revealed by server.

4) The verifier chooses a random bit. If the bit is 0, the server publishes the shuffle arrangement π' and random re-encryption parameter w', from which it can prove the correctness of the conversion from E to E'. If the bit is 1, the server publishes π'' and w'', from which it can prove the correctness of the conversion from E' to E''.

5) The verifier checks all revealed conversions. If the check is passed, the verifier accepts that all b_i are the decryptions of E.

The server and verifier repeat the above steps for n times to ensure the decrypted numbers correctness at the probability of $1 - (0.5)^n$. The RPC procedure is shown as Fig. 4.

2 Blockchain based on RLBC

The blockchain in proposed scheme uses RLBC as the consensus method among all voters to make the saved vote consistent. For practical implementation, a more flexible and efficient consensus method should be applied to blockchain e-voting. The consensus needs to be achieved by less than 50% of all consensus nodes. The speed of consensus process also needs to be fast enough for voters. Because RLBC can decode information word with lower than 50% of distributed elements in the code word with high efficiency, RLBC consensus can meet this requirement. The consensus nodes management mechanism of RLBC can make blockchain consensus operation work efficiently. Therefore, the proposed RLBC consensus is more practical than PBFT, Raft and Paxos.

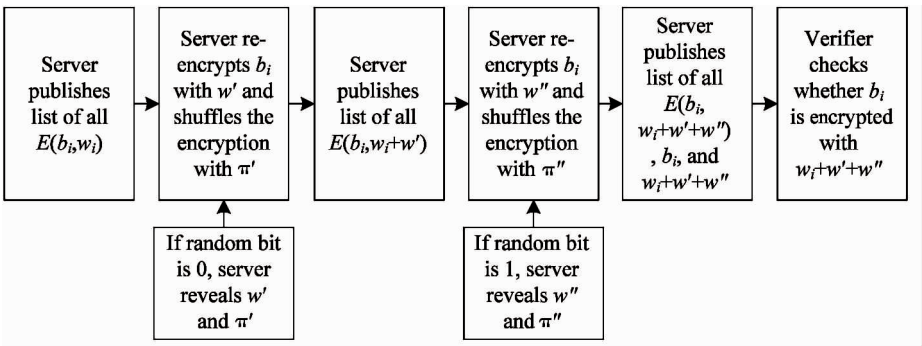

Fig. 4 RPC procedure

2.1 Blockchain consensus of RLBC

Choose the suitable column number n to make the size of matrix $\boldsymbol{H}^{\mathrm{T}}$ fit the required number of consensus nodes. Blockchain consensus of RLBC is implemented through that one node encodes information and others decode it for consistency through the methods discussed in Section 1.2. When one node makes a transaction information $\boldsymbol{v}$ with k elements consistent with others in the blockchain network, it encodes $\boldsymbol{v}$ as a binary vector to code word $\boldsymbol{u}$. Then this node equally distributes the last $n-k$ elements of $\boldsymbol{u}$ to other nodes including the elements positions in $\boldsymbol{u}$ with its signature of these information. Other nodes sign and exchange these received information between each other, then they decode the received elements to $\boldsymbol{v}$.

When one node finishes decoding and gets a group of $\boldsymbol{v}$, it builds the group with the decoding elements and signatures of them into a block. For the efficiency of the consensus, there are more than one ballot in a generated block. The node hashes all decoding elements and decoding results with the head of the previous block, then it sends the block to other nodes. After checking that the group of $\boldsymbol{v}$ and the block hash is correct, the other nodes chain this block.

2.2 Maximum number of consensus nodes

Each consensus node should receive at least 1 element from the voting node. The maximum number of the distributed code word elements is the maximum number of consensus nodes performing the consensus operation at same time. In practical terms, the maximum number of consensus nodes is decided on the encoding and decoding speed of RLBC using different number of distributed code word elements. As shown in Section 1.2, the number of distributed code word elements $n-k$ can be gotten by setting δ with information length k and decoding threshold r. RLBC operation speed under different number of the distributed code word elements is shown in Section 5.1.

3 Proposed scheme for e-voting on blockchain

In the proposed scheme, voters vote through their own device such as a computer or mobile phone. The device of voter is working as blockchain node. The nodes of voters make the consensus together to build the blockchain. The server initiates the blockchain operation, provides the encryption service, and verifies the decryption of ballots. If there are lots of voters in an election, it needs to have more than one verification servers. All voters are equally allocated to these servers.

3.1 System initiation

Before voting begins, all nodes generate their public and private keys. Then the nodes sign in to a server with their public keys. The server generates coding and parity-check matrixes of RLBC. The server chooses some nodes randomly to be the initial blockchain consensus nodes. The server publishes its public key, matrixes of RLBC, and the lists of all nodes with their public keys.

3.2 Vote preparation

The voter inputs its opinion on the node which is its device. After filling out the ballot, the voter ensures the correct encryption of its vote through the cast-or-audit method.

1) The node sends the ballot to the server through a security channel. The server encrypts the ballot with parameter and sends the encrypted ballot back. After receiving the information back, the node shows it to the voter with 2 options: cast or audit.

2) If the voter chooses to audit the encryption, the server shows the encryption parameter to the voter, and the voter checks the encryption correctness.

3) The voter continues to cast the ballot. The

server encrypts the ballot with another parameter and the node shows the options again. If the voter chooses option of audit again, it audits encryption as last step. The process will be repeated, until the voter chooses option of casting the encrypted ballot.

3.3 Voting on blockchain

After the voter chooses to cast the ballot, the node on its device encodes the vote and makes the consensus as detailed in Section 2. The node encodes the encrypted ballots, then equally distributes the elements and their position in the code words with its signature to other consensus nodes. After signing and exchanging the information of the elements, consensus node decodes all received elements to encrypted ballot. The node makes a block from the decoded encrypted ballots and the decoding elements, then publishes the block to the network. When the consensus is achieved, the node accepts the block into the chain.

If the node receives a block that is checked being correct by itself before it generates one, it stops generating the block and accepts the received one. If the node receives a block and checks it being correct after accepting a block that records the same ballots, the node saves received block as fork block of chain. The node will continue to build the block on the longest chain. If all forks have the same length, the node works on the chain built earliest.

After finishing voting, the voter can record the encrypted ballots to check them in the blockchain later. The server does not take part in the consensus and only receives the generated blocks to verify the decryption correctness. The node which does not take part in the consensus can also download the blockchain from the server.

3.4 Open and tally votes

After the voting ends, the list of encrypted ballots in the blockchain is decrypted by the server. The decryption is verified by voters through the RPC method discussed in Section 1.4. The decryption and verification processes are detailed as follows.

1) All encrypted votes E are re-encrypted and shuffled twice using different parameters by the server. The server publishes the results of the 2 conversions on the bulletin board and the decryption of all ballots.

2) The previous step is repeated several times. The bulletin board shows all pairs of the conversions.

3) Each voter who verifies the decryption puts a random number chosen by itself on the bulletin board. Using these random numbers as seeds of a random generator, the server generates a random number and the verifier voters check it together. From the binary of the generated random number, an unpredictable bit string s is yielded and published on the bulletin board.

4) According to one bit in s, one conversion in each pair of the RPC is revealed. For example, if the j-th bit s_j in s is 0, the conversion from E to E' of the j-th pair published on the bulletin board is revealed. If s_j is 1, the conversion from E' to E'' of the j-th pair is revealed.

5) All verifier voters check the correctness of all revealed conversions on the bulletin board. Moreover, they check the equalities of the decrypted ballots from different RPC pairs. If all the published information is proven correct, the verifier voters accept the decrypted ballots as the correct. Each verifier voter tallies all votes to get the voting results.

If there are more than one group of voters voting and building different blockchains, all tally results of blockchains are tallied by these servers together.

3.5 Blockchain consensus nodes self-organization

To increase the total efficiency of the whole blockchain network, the nodes which take part in the consensus organize themselves. The consensus node excludes the inactive one and allows the new nodes in the consensus operation through distribution of the code word elements. The node checks the decoding element information and their signatures in the block. If a voter node does not have its received element in the nearly generated blocks, the node will stop sending code word elements to it. The node chooses a node which is not taking part in the consensus as new permitted one. Then, it sends the code word elements to the new permitted node of consensus. The new permitted node finds the active nodes according to the records in the blockchain and makes the consensus with them.

The node sends the elements for consensus to others that exchange elements successfully with it recently. The node stops sending elements to the node that cannot exchange elements several times. If a node is more active in exchanging elements, the node has more chance to decode the code words and build blocks. And the node has more chance to build its own received elements into blocks by itself or others. The node which exchanges elements with others and builds blocks actively has a less probability to be excluded from the blockchain consensus operation. The elements built in the blocks that take effect at final can be the proof of the consensus operation prize.

3.6 Blockchain fork control

The blockchain fork occurs because more than one

correct blocks containing the same ballots are published into the network by different nodes. If the node builds and publishes fork blocks after it has received the correct block which can achieve the consensus, the number of fork blocks will increase, and the efficiency of whole network will decrease. The storage of fork blocks on consensus nodes will also increase unnecessarily. To reduce fork blocks and prevent faulty nodes increasing forks, the consensus node will stop sending elements to the node that generates the most fork blocks and has least received elements in all blocks for a given period of time. To reduce the fork blocks, the consensus node stops publishing blocks generated by itself after receiving correct blocks from others.

4 Security analysis of the proposed scheme

In this section, according to all methods described above, this work summarized primary information security properties of the proposed scheme.

Eligibility: all voters need to register their devices in to the server by which their authorization is checked.

Privacy: voter votes on its own device and keeps the voting secret. The blockchain network makes consensus and publishes the encrypted ballots which cannot be decrypted by anyone.

Verifiability: the proposed scheme can achieve end-to-end verifiability that includes the following 3 parts.

1) Cast-as-intended: before casting a vote, voters can check the encryption correctness. The encrypted ballots are recorded in blockchain and free of being removed or tampered with.

2) Recorded-as-cast: after the voting ends, the voter checks that all encrypted ballots in the blockchain are published on the bulletin board. The voter can check that the ballots are decrypted correctly through RPC.

3) Tallied-as-recorded: the decrypted ballots can be tallied by all voters in the blockchain network.

5 Experiments of the consensus based on RLBC

This work experimented the elapsing time of RLBC encoding and decoding in different given conditions. And this work experimented the consensus operation speed of the scheme. The experiments were implemented on a machine with dual core 2.00 GHz CPU and 2 GB memory running 64 bit Ubuntu 16.04. RSA signature and SHA256 hash algorithms are used in the experiments. Each signature and hash value has 4 bytes data.

5.1 Experiment of RLBC encoding and decoding speed on different number of distributed elements

The experiment took e-voting with each encrypted vote which has 5 bytes as an example. The information word of RLBC is decoded with 40% threshold elements of distributed elements in the code word. The encoding and decoding time are shown in Fig. 5.

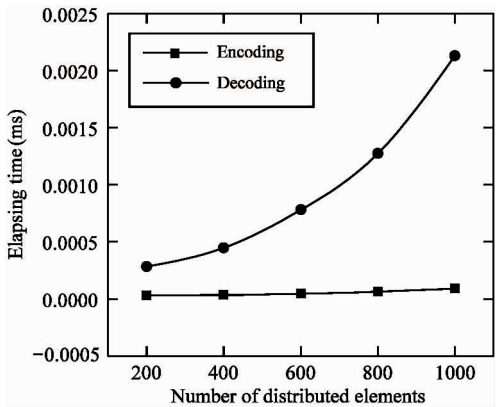

Fig. 5 Elapsing time of encoding and decoding on different number of distributed elements

5.2 Experiment of RLBC encoding and decoding speed on different decoding elements threshold of distributed element

This work experimented the elapsing time of RLBC encoding and decoding which can be decoded with different threshold elements proportions in distributed elements of the code word. In this experiment, the information word data size is 5 bytes and the distributed elements in code word is 600 under different given conditions. The encoding and decoding time are shown in Fig. 6.

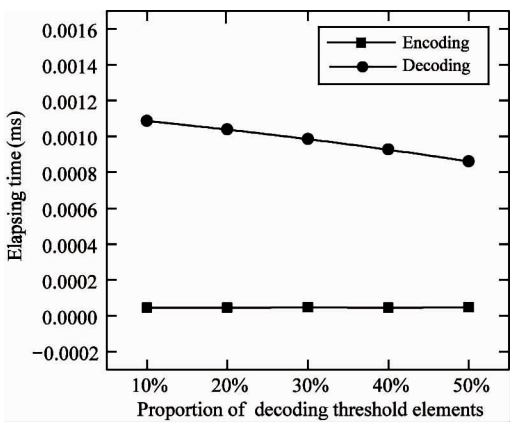

Fig. 6 Elapsing time of encoding and decoding on different decoding threshold elements proportions

5.3 Experiment of consensus speed comparison

In this experiment, an encrypted vote has 5 bytes, each block has 10 votes, and the distributed elements in code word of a vote is 600. The information word of RLBC is decoded with 40% threshold elements of distributed elements. When the numbers of nodes are different, the comparison of the consensus speeds among RLBC consensus, Paxos and Raft is shown in Fig. 7.

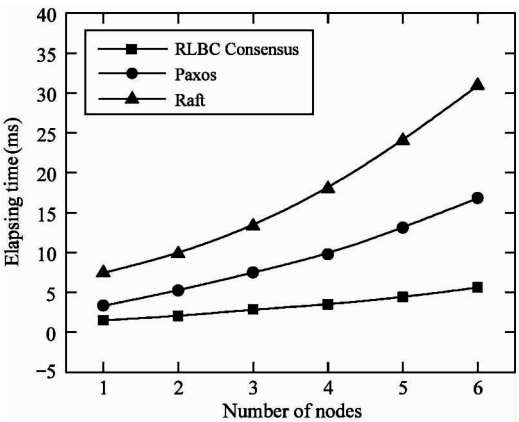

Fig. 7 Consensus time of different protocol

6 Conclusion

This work presents a blockchain e-voting scheme with end-to-end verifiability. The blockchain scheme uses RLBC as consensus protocol. RLBC based consensus can organize the nodes of consensus to achieve high efficiency of the whole blockchain in distributed application. The consensus can also limit nodes generating forks. The RLBC consensus has higher flexibility than the current protocols and is more practical in a realistic decentralized voting scenario. The voter can verify high probability correctness of its vote end-to-end by cast-or-audit and RPC without ruin of voting privacy.

This scheme has further scope for improvement. RPC needs a lot of computation and bulletin board. More compact verification techniques based on smart contact need to be developed as part of future research. More nodes should be implemented in future experiments.

References

[1] Mursi M F M, Assassa G M R, Abdelhafez A, et al. On the development of electronic voting: a survey[J]. *International Journal of Computer Applications*, 2013, 61(16):1-11

[2] Esteve J B. A preliminary question: is e-voting actually useful for our democratic institutions? What do we need it for? [C]//Proceedings of the 2nd International Conference on Electronic Voting, Castle Hofen, Austria, 2006: 51-60

[3] Benaloh J. Ballot casting assurance via voter-initiated poll station auditing[C]//Proceedings of the 2007 USENIX Workshop on Accurate Electronic Voting Technology, Boston, USA, 2007: 14-20

[4] Benaloh J. Simple verifiable elections[C]//Proceedings of the 2006 USENIX Workshop on Accurate Electronic Voting Technology, Vancouver, Canada, 2006: 5-14

[5] Chaum D L. Untraceable electronic mail, return addresses, and digital pseudonyms[J]. *Communications of the ACM*, 1981, 24(2): 84-90

[6] Neff C A. A verifiable secret shuffle and its application to E-voting [C]//Proceedings of the 8th Conference on Computer and Communications Security, Philadelphia, USA, 2001: 116-125

[7] Groth J. A verifiable secret shuffle of homomorphic encryptions[J]. *The Journal of Cryptology*, 2010, 23(4): 546-579

[8] Lee K, James J I, Ejeta T G, et al. Electronic voting service using block-chain[J]. *The Journal of Digital Forensics, Security and Law*, 2016, 11(2): 123-136

[9] Hsiao J H, Tso R, Chen C M, et al. Decentralized E-voting systems based on the blockchain technology [C]//Proceedings of the 2017 International Conference on Ubiquitous Information Technologies and Applications, Taichung, China, 2017: 305-309

[10] Zhao Z, Chan T H H. How to vote privately using bitcoin [C]//Proceedings of the 17th International Conference on Information and Communications Security, Beijing, China, 2015: 82-96

[11] Hjalmarsson F P, Hreioarsson G K, Hamdaqa M, et al. Blockchain-based E-voting system [C]//Proceedings of the 11th IEEE International Conference on Cloud Computing, Seattle, USA, 2018: 983-986

[12] Hanifatunnisa R, Rahardjo B. Blockchain based E-voting recording system design[C]//Proceedings of the 11th International Conference on Telecommunication Systems Services and Applications, Lombok, Indonesia, 2017: 1-6

[13] Singh A, Chatterjee K. SecEVS: secure electronic voting system using blockchain technology[C]//Proceedings of the 2018 International Conference on Computing, Power and Communication Technologies, New Delhi, India, 2018: 863-867

[14] Adam K E. The key to unlocking the black box: why the world needs a transparent voting DAC[EB/OL]. https://followmyvote.com: Decentralized Autonomous Companies, 2014

[15] Dagher G G, Marella P B, Milojkovic M, et al. BroncoVote: secure voting system using Ethereum's blockchain [C]//Proceedings of the 4th International Conference on Information Systems Security and Privacy, Funchal, Portugal, 2018: 96-107

[16] Chaum D, Essex A, Carback R, et al. Scantegrity: end-to-end voter-verifiable optical-scan voting[J]. *IEEE Security and Privacy*, 2008, 6(3): 40-46

[17] Bohli J M, Müller-Quade J, Rührich S. Bingo voting: se-

cure and coercion-free voting using a trusted random number generator[C]//Proceedings of the 1st International Conference on E-voting and Identity, Bochum, Germany, 2007: 111-124

[18] Bohli J M, Henrich C, Kempka C, et al. Enhancing electronic voting machines on the example of Bingo Voting [J]. *IEEE Transactions on Information Forensics & Security*, 2009, 4(4): 745-750

[19] Teng P G, Zhang J Z, Chen L, et al. Random RAID: a RAID storage scheme with high fault-tolerance and flexibility[J]. *Advanced Engineering Sciences*, 2017,49(3): 110-116

[20] Nakamoto S. Bitcoin: a peer-to-peer electronic cash system[EB/OL]. https://bitcoin.org: Bitcoin.org, 2016

[21] Underwood S. Blockchain beyond bitcoin[J]. *Communications of the ACM*, 2016, 59(11): 15-17

[22] Castro M, Liskov B. Practical Byzantine fault tolerance [C]//Proceedings of the 3rd Symposium on Operating Systems Design and Implementation, New Orleans, USA, 1999: 173-186

[23] Lamport L. Paxos made simple[J]. *ACM SIGACT News*, 2001, 32(4): 51-58

[24] Ongaro D, Ousterhout J. In search of an understandable consensus algorithm[C]//Proceedings of the 2014 USENIX Annual Technical Conference, Philadelphia, USA, 2014: 305-320

[25] Jakobsson M, Juels A, Rivest R L. Making mix nets robust for electronic voting by randomized partial checking [C]//Proceedings of the 11th USENIX Security Symposium, San Francisco, USA, 2002: 339-353

Liu Ting, born in 1978. He is currently working toward the Ph. D degree in computer science at University of Chinese Academy of Sciences. He received his B. S. degree in computer science and technology from Central South University in 2006. His research interests include blockchain, cryptology and coding theory.