# Blockchain based Voting System

Ruchit Vithani
*DAIICT*
Gandhinagar, India
201701070@daiict.ac.in

Jaydeep Kakadiya
*DAIICT*
Gandhinagar, India
201701268@daiict.ac.in

***Abstract*—The traditional voting processes conducted in some of the large democracies of the world lack in terms of providing transparency and verifiability. Generally the voters do not have access to their voting data after the election is over. Also they must visit the voting booth in order to cast their vote. In such traditional systems, voters can not verify after the election that their vote is indeed cast to the candidate for whom they voted for. In this paper, we introduce a different voting protocol that leverages the security, pseudonymity, transparency, and verifiability in the election process. No any central user or entity can control this system single handedly. In addition to voting protocol, we also explain the inner workings in the blockchain network for such system. There are less complex cryptographic systems designed for the purpose of voting, however, in such systems, everyone must trust the third parties involved in various tasks like allocating private-public keys, storing the election data, counting votes and publishing the results. We argue that a decentralised system where no single entity, rather, the whole community is responsible for whatever actions happening in the system, is ideal for such voting processes.**

***Index Terms*—Blockchain, Evoting, Distributed protocols, P2P, Cryptography**

## I. Introduction

The process of election has been evolved a lot and many protocols for elections have been proposed. With the invent of internet, the online platform for conducting election has been established. These systems mainly use cryptographic methods to conduct online elections in secure manner. In this paper, we propose a new distributed remote voting system working on the top of blockchain protocol. Such system leverages security and transparency benefits of the blockchain. We analyse various challenges and cyber threats to such systems. We propose a modified blockchain protocol for transaction flow and updating of the blockchain in the network. Here we first define what should be the criteria of successful voting system, what services should such system provide.

- Characteristics of a good voting system :
  1) Only eligible voters should be able to vote at most once. Non eligible attempts to cast votes should be invalidated and should not be counted in total vote count.
  2) A single person or entity should not be able to control the whole system.
  3) Voters after completion of election should be able to verify that their vote is counted and is not tampered.
  4) Anonymity about who voted whom should be maintained.
  5) Once a valid vote is casted to the system, it should not be able to get tampered in any case.

## II. Related work

Many electronic voting systems have been proposed by many researchers in the field of cryptography and computer science in order to achieve the main criteria of successful voting systems. Such digital systems can be mainly classified into two categories. (1) Cryptographic systems to ensure voter privacy and election security, and (2) Distributed election systems generally based on the blockchain like the one we are proposing in this paper.

- Cryptographic systems : Several systems making use of public key cryptography for the encryption of votes have been proposed. Some of the systems use a protocol called Zero-knowledge proof.
  - Zero-knowledge proof : Zero-knowledge protocol is a method by which one party (the prover) can prove to another party (the verier) that they know some information x without conveying any other information about the prover apart from the information that they know the information x. Such a Zero-knowledge proof based voting systems have been employed in [1] [6] [7].
  - Blind signature based systems : The concept of Blind signature was developed by David Chum in cryptography. This is a method by which contents of the message are disguised before the message is signed. This scheme is built on the top of well-known public key cryptography algorithms like RSA in order to hide the contents of the original message. In blind signatures, a vote transaction can be verified even without looking at the original message. The concept of blind signatures have been employed by [8] for implementing an electronic voting system. However, in such systems, everyone must trust the signer. If the signer is compromised, the privacy and security is compromised as well.
- Blockchain based voting systems : Because of central control over voting system in some way or another in the cryptographic systems described above, many researchers have been exploring the idea of making this system decentralised. Such systems take the control out of central entity controlling the election and puts it in the hands of

trusted community of handful of people. People can cast their votes without worrying about the integrity of their votes and privacy of their own identity in such systems. Such systems have been explored in [2] [3]. In this paper, in addition to describing a new distributed protocol for voting, we also demonstrate the inner workings of such system. We explain how the transaction flow and updating of blockchain may be carried out in such system. The implementation of our system is also available on the GitHub.

## III. Study of challenges

Security of voting system is main concern for election . Although the digital voting system over internet might reduce the immense costs involved in the traditional voting processes, it also invites a large number of security threats on such system. In this section, we describe the extensive study of various cyber attacks that are possible on any digital voting system over internet. The author in [4] describes a wide range of attacks which are possible on internet voting system, either based on blockchain or without blockchain.

- Threats to blockchain based systems
  - There are at least two possible ways to implement such blockchain based voting systems, the *single-owner blockchain* and *multi-owner blockchain*. Single owner chains are deployed and maintained by the single organisation or company. On the other hand, multi-owner chains are deployed and maintained by multiple owners. The owners in multi-owner chains may or may not trust each other, they just rely on the distributed protocol that keeps them functioning. Both types of chains have different security threats. In case of single-owner blockchain, the attacker might find a single penetration point in the organisation using which he can hack multiple numbers of nodes easily. The other pin over such system is every stakeholder of the system must trust the organisation hosting the blockchain. Blockchain contents can also be manipulated by some insiders in the organisation. So such system invites easy attacks from both internal as well as external sides.

    In multi-owner blockchain, the threats of internal attacks are reduced considerably since the main blockchain is hosted and maintained by multiple organisations at a time. However, such systems can be easily taken down or misbehaved if a subset of owner organisations manage to aggregate the computation power of more than half of the nodes in the network, then the network can easily be taken down.
  - Even if we consider the blockchain as secured within the organisation(s), the outsiders might attempt to attack the whole network and take the control over more than half of the nodes. Any system deployed over internet always has one or other points of penetration. We just simply can not ignore the possibility of an attacker finding a way to attack over half of network at a time.

- Curse of the ”INTERNET”
  - Security of confidential credentials : Even if the attacker can not attack the whole network, he may easily be able to get access to a single node in the network. He can easily steal and misuse the confidential information stored in the network. For example, He can access the authentication details of the users, can know the identities of the users etc.
  - Malware : If such systems are allowing the voters to connect and cast their votes remotely, there is always a threat to the users. The attackers can easily attack the devices of these remote users via malware and malicious programs using the mediums like email, sms etc. The internet provides them a medium to attack millions of devices at a time. Such attacks might lead to manipulation of UI during voting process (leading to tampered vote), device misbehaviour, failures in devices etc.
  - Penetration : As discussed before also, the medium of internet provides the attackers to always find new ways sneaking into system. Such penetrations can allow attacker to control more than half of the system, and at the end the whole system according to their wish.

- Implementation challenges
  - *Transaction Integrity* : Different nodes in the blockchain receive different transactions continuously in the network. Our network must maintain the integrity of the transactions. The blocks in such systems are proposed more easily and frequently compared to cryptocurrency blockchain. Our system must make necessary changes in the protocol and prove the integrity of the transactions and blocks in the protocol.
  - *Scalability* : To use such system at large scale elections (like nation-wide elections of large democracies), our system should be able to work with large number of nodes, voters and transactions at any time. We can test our system in terms of scalability by first running large number of transactions and online users (e.g. 1M voters) at a time, and assuring that the system is working as expected.
  - *Performance* : The performance of the system matters a lot in terms of its speed in handling transactions, speed in updating chains across the network, the waiting time for the voters to get their transaction id after voting etc. In order to handle large scale elections, our system must provide guarantees in the quality of such services.

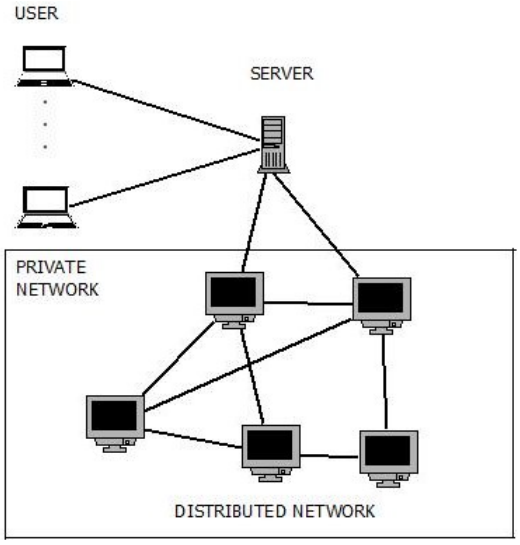


Fig. 1. Overview of system

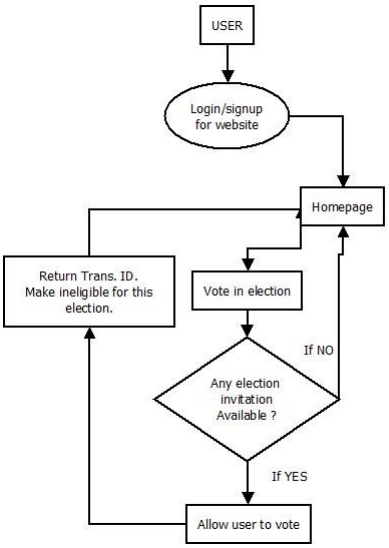


Fig. 2. Voter interaction with server

## IV. Design

In this paper, we present a hybrid model consisting of one central client-server interface, to handle remote voting, configuring peers for nodes in the network, transaction redirections etc, and one distributed network which stores and validates the voting data, maintains the integrity of the data, increases the security of the data stored. Fig 1 describes the architecture of our system.

### A. Client server Interface

Client-server interface provides an interface for the remove voters to vote remotely, hosts to organise the election. The server authenticates the voters and redirects their transactions to the network. The server can also be used for configuration of peers in distributed network. The new nodes that need to join the distributed network will find the IP address of the peers via the server. Server will perform various election related functions as described below.

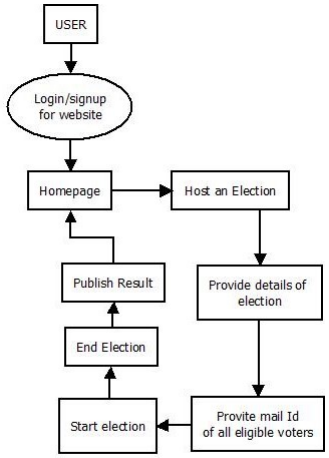


Fig. 3. Flow of election organisation

- *Voter registration phase* : All the voters will be registered first via their email. During the registration they will be assigned an encryption key on the back-end which will encrypt their public address and vote transaction. This encryption key will not be visible to voters and the encryption will be done automatically on the back-end. This provides anonymity to voters as encrypting public keys hides the identities of the users. The encryption of vote transactions makes them inaccessible to outsiders. This makes sure that no one can find out to whom the given vote is casted. Users are also assigned a private key using which they can sign their transactions. This signature will be used to verify the transactions of the voters.
- *Functions implemented in the Server* : Server verifies the eligibility of the voter when they attempt to send a vote transaction to the distributed system. If the voter is not registered for an election, the server will block its request. Server maintains a list of users who have already voted in order to block them from further voting. Server can also create the election and initialise the new blockchain instance in the network according to the parameters provided by the host of election.
  Further, server will also be responsible for counting votes and publishing the results after the election is over. Server does not store any vote data, it just accesses the distributed network to tally the results.
  Server will ensure connection with SSL and will use ECC encryption and SHA256 as signature algorithm. Fig. 2 shows the voter-server interaction flow, and fig. 4 shows the flow of election organisation.

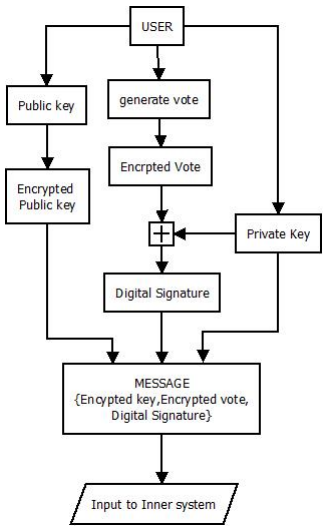


Fig. 4. transaction formation

### B. Distributed inner system

Distributed system consists of number of nodes connected with each other in a distributed manner. Each node has several number of peers connected to it. Nodes in the distributed network maintain the queue of transactions they receive. Each node also maintains the local copy of their own blockchain. The main protocols running inside this system during the voting process are (1) Transaction flow (2) Synchronous updating of blockchain across the network. Here we describe these protocols in detail.

1) *Transaction flow* : Every node stores the local pool of set of transactions in a queue. Each transaction in a queue is a valid transaction. The nodes in our network follow a predefined protocol in order to work with new transactions. The transactions corresponding to remote voters are redirected to the distributed network via server. Whenever a node in the network receives the transaction, it first validates the transaction.
   - *Validation :* How do nodes validate the transaction? First we define the structure of a transaction. Transaction is corresponding to vote casted by a voter to any candidate. So, the main attributes of a transaction will be From, To and timestamp. At the voter side, these transactions will be first encrypted and then signed using the private key of the voter. Only the counting function written on the server will have access to the decryption key of the encrypted message. To hide the identity of the voters, we also encrypt the public key of the voter during voting. So, what the distributed system get as a transaction is : (i) Encrypted public key of voter (ii) Encrypted vote message (iii) signature of the voter. The nodes in the distributed network work as validator nodes for the transaction. Fig. 4 shows the process of encryption for validation. They will take the encrypted vote message from the transaction, obtain the public key of voter by decrypting the encrypted public key of the voter, and then verify the transaction using the signature and decrypted public key of the user. Such a cryptographic protocol achieves the two important criteria of a good voting system. It hides the identity of users by encrypting the public key of the sender. This way no one can find out the original user corresponding to the transaction. It also hides the contents of the transaction by encrypting the original message at the beginning. This way, no one can find out just by looking at the transaction that to which candidate the given vote is for.

   After validation, transactions are sent into the network in store and forward manner. If the transaction is validated successfully, it is stored in the local queue of transactions for that node. While forwarding, the nodes follow advertising mechanism in order to avoid flooding the whole network from the transaction traffic. This way every node receive the correct transaction exactly once.
   - *Advertising mechanism :* Whenever any node receives the transaction, it first advertises the transaction id of the received transaction to its peers. The peers request the transaction if they do not have the transaction with received transaction id in their local pool of transactions. The node only forwards the transaction to the nodes who request for it, not to all the peers. The nodes after receiving the requested transaction, store and forward them in similar manner.

   If the received transactions turn out to be invalid, the nodes drop them without storing or forwarding them in the network. This way, the manipulated/tampered transactions will be invalidated by the nodes in the network and no one will store them or eventually merge them in the blockchain. This protocol also proves that blockchain always stores the valid vote data and tampering the vote data once it is received by the distributed network is not possible.

2) *Protocol for New block proposal* : How do the nodes propose new blocks? How do the newly proposed block propagate in the distributed network? How is the blockchain updated? Our protocol for new block proposal answer these questions. Validation of transactions is not a computation intensive task. It's just the verification of transactions using the signatures provided with them. So ideally, the nodes can propose the new blocks right at a time they receive any transaction. Or the nodes may wait for some time to collect a number of transactions to put inside the block and once that number is collected, propose a new block. However, we can not afford too much of scenarios like two nodes in the network propose the block at the same time. There are solutions to this problem in

blockchain protocol. For example, the bitcoin network considers the longest chain from all possible forks as a valid chain. We have designed a protocol to prevent this scenario. Our protocol guarantees that at any time, exactly one node will be proposing the new block. First let us introduce the notion of global variable dependent on synchronised blockchain named "TURN".

- *Notion of TURN variable :* The node with id equal to TURN % N must propose a block. N is total number of nodes in the network. If any node does not have any transactions to include in the block, it must propose an empty block, but it must propose a block. the variable TURN will be maintained by the server. Once a node with id TURN%N proposes a block, the new block will be transmitted to the rest of the network. If the proposed block is a valid block, the consensus will be established in the network and it will be merged eventually. If the block is not valid, then the majority nodes in the network will not accept the new block and it will be dropped eventually. Additionally, if any other node except the one with id TURN%N try to propose a block, the peer nodes will neither store nor forward it. This proves that at any given time, only one block will be proposed by the nodes in the network.

- *Consensus algorithm :* The new blocks are propagated in the similar manner as newly arrived transactions. But there is also one additional step before forwarding the newly received block to the peer nodes. Remember that the blocks stored in the blockchain are permanent and no one can make any changes to data stored in the blockchain. Such permanent storing of data in the blockchain network is done on the basis of consensus mechanism. The data can only be stored in the main chain if majority of the nodes rich to a consensus for merging the block. Whenever any node receives the new block, it will first advertise the block-id to its peers, and put the block on hold. The node will hold the block until it receives the advertisement of the same block-id from the majority of its peers. Once the majority agreement established at a node, it will add it to its local chain and propose the same block in store and forward manner (Advertising followed by proposing). Note that whenever a node receives the new block, it do not necessarily have all the transactions put inside that block. In such cases, the node verifies the transactions that are not in its local pool of transactions. If even a single transaction is invalid, the block will be dropped and not advertised to its peers. When the node receives the majority advertisements from its peers about the new block it received, it will remove all that are present inside the block from its local pool of transactions. This guarantees that the same transaction will not be merged more than one time.

## V. Correctness and Security analysis

### A. 51% attack

For smaller distributed networks, the chances of a node gaining computing power of over half of the network are considerable. Increasing the number of nodes in the network improves the security of the whole chain. Also there is trade-off between the number of peers each node have and the performance of the network. More number of peers for nodes improves the security of the network against 51% attacks, but this also reduces the performance of the network in terms of speed. Increasing the number of peers also increases the traffic of transactions in the network.

### B. security

The voter transactions on the blockchain are stored in encrypted manner. Only the counting function at server or that written on the smart contract knows the decryption key of the encrypted transaction. This ensures the privacy and security of the transactions stored in the network,

### C. anonymity

The election administrator or any user can not predict the committer of the transaction simply by looking at the transaction. The public keys have been encrypted asymmetrically. So, our system provides the required level of anonymity.

### D. variability

The owners of the blockchain and the voters can verify that their votes have been counted correctly without any manipulation. The voters can provide their transaction id, and the system will return him his transaction. The voters can verify the returned transactions. Even voters can clone the blockchain to their devices and call the verify() function in order to verify the every transaction stored in the blockchain.

### E. correctness

Since the transactions can be broadcasted over the whole network only if they are validated, the correctness of transaction is ensured. The validator nodes work in store-and-forward manner, where they first verify the transactions based on their signatures, and then store and forward them only if they are valid. Otherwise our system drops the transaction without storing or forwarding. When the transactions are merged in the main chain, the consensus protocol ensures that the transactions have been validated by more than half of the nodes in the network.

### F. Unit voting

The server maintains the Boolean list of voters who have voted. A voter with one address will only be allowed to vote at most once. This satisfies the important requirement of the voting system. One the voter votes for any party/candidate, the server will mark the address of that voter so that if he tries to vote again, he can be blocked.

### G. Voter eligibility

The server always maintains the data of registered voters. It can always verify the valid voters. This makes sure that only valid voters can cast their vote in the system. This also helps in calculation of percentage of voting done in particular election.

### H. vote integrity

Once the vote transactions are validated and merged by the distributed network, any attempt to manipulate the transaction will fail because of distributed consensus. This meets a very important requirement of a good election system.

### I. DDoS attacks

DDos attacks on the distributed network are not possible since the remote transactions are not directly connected to distributed network. Remote transactions are redirected to the network via a set of servers. This makes DDoS attacks impossible on our system. There are many different ways to prevent the DoS attacks on the servers.

## REFERENCES


[1] Neff, C.A.: A verifiable secret shuffle and its application to e-voting. In: Proceedings of the 8th ACM conference on Computer and Communications Security. pp. 116–125. ACM (2001)

[2] F. . Hjálmarsson, G. K. Hreiarsson, M. Hamdaqa and G. Hjálmtýsson, "Blockchain-Based E-Voting System," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, 2018, pp. 983-986, doi: 10.1109/CLOUD.2018.00151.

[3] Teogenes Moura and Alexandre Gomes. 2017. Blockchain Voting and its effects on Election Transparency and Voter Confidence. In Proceedings of the 18th Annual International Conference on Digital Government Research (dg.o '17). Association for Computing Machinery, New York, NY, USA, 574–575. DOI:https://doi.org/10.1145/3085228.3085263

[4] David Jefferson, Verified Voting. The Myth of "Secure" Blockchain Voting

[5] Yu B. et al. (2018) Platform-Independent Secure Blockchain-Based Voting System. In: Chen L., Manulis M., Schneider S. (eds) Information Security. ISC 2018. Lecture Notes in Computer Science, vol 11060. Springer, Cham

[6] Chow, S.S., Liu, J.K., Wong, D.S.: Robust receipt-free election system with ballot secrecy and verifiability. In: NDSS. vol. 8, pp. 81–94 (2008)

[7] Weber, S.: A coercion-resistant cryptographic voting protocol-evaluation and prototype implementation. Darmstadt University of Technology, http://www. cdc. informatik. tudarmstadt. de/reports/reports/StefanWeber. diplom. pdf (2006)

[8] Li, C.T., Hwang, M.S., Lai, Y.C.: A verifiable electronic voting scheme over the internet. In: Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on. pp. 449–454. IEEE (2009)