Design and Analysis of Blockchain-based E-voting Protocols

Sarbajit Ghosh

Design and Analysis of Blockchain-based E-voting Protocols

DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology in Cryptology and Security
from

Indian Statistical Institute, Kolkata
by
Sarbajit Ghosh
[Roll No: CrS1911]

Under the Supervision of:

Dr. Souradyuti Paul
Associate Professor
EECS Department
Indian Institute of Technology (IIT) Bhilai

Institute Supervisor

Dr. Debrup Chakraborty
Associate Professor
Cryptology and Security Research Unit
Indian Statistical Institute, Kolkata

Dedication

To mom and dad ...

Declaration

I would first like to express my profound gratitude to my advisor, **Dr. Souradyuti Paul**, for his patient guidance, enthusiastic encouragement, and useful critiques needed for this research work. His advice and knowledge helped me in pursuing good research and writing this thesis. I am privileged to have such a great supervisor as he consistently supported me both academically and personally.

My thanks are also extended to my institute supervisor **Dr. Debrup Chakraborty** who have paved the way towards my research acumen.

Sarbajit Ghosh
Student
M.Tech. in Cryptology and Security
Indian Statistical Institute
Kolkata

Acknowledgements

My thanks to **Mr. Manish Vuppala**, MTech Scholar, IIT Bhilai, for full collaboration and various interesting discussions.

Sarbajit Ghosh
Student
M.Tech. in Cryptology and Security
Indian Statistical Institute
Kolkata

Abstract

Voting is the most fundamental cornerstone in the context of representative democracy. With the advent of the internet technologies, various types of e-voting mechanisms have emerged. Building an e-voting system capable of performing as good as a traditional voting system is a very challenging task. On the positive side blockchain technology inherently provides critical security properties to design secure e-voting protocols. In this work, we have studied decentralized boardroom scale e-voting protocols designed by McCorry et al. in 2017. We have extended their work in multiple directions. Our protocol supports (A) arbitrary number of candidates[1]; (B)majority-base countings; (C) voting absentation (D) we also did a theoritical analysis of all the protocols.

Keywords: Blockchain, E-voting, Ethereum, Decentralized e-voting, Boardroom scale voting, Majority counting, Borda counting.

[1]This is a joint with researchers at IIT Bhilai

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In a representative democratic country [30], voting is the process of selecting parliamentary representatives. Selecting a representative is immensely important, as it gives the power to citizens to select candidates of their choice, and protect the rights of citizens. Traditional voting systems include ballot-based systems and electronic voting systems.

1.1 Motivation

Conducting a fair voting mechanism, and selecting a candidate through this system is still a challenging task. Although the traditional voting systems successfully served the purpose of voting for many years, they also have limitations. Some of them are:

- Sometimes, people do not cast votes because the voting centre is far from their houses.
- The communication between the voting centre and voters' houses may not be very good.
- It is difficult for some voters to stand in a long queue to cast their votes.
- The cost of these voting systems is large, which can be significant for a developing country with a high population.

Therefore, searching for a secure, easily accessible, and cost-effective voting system is a natural pursuit. However, building an internet-based e-voting system having comparable to the traditional voting system is a challenging task. Also, transforming traditional voting systems to their digital counterparts often requires special domain knowledge, e.g. cryptography and communication protocols.

On the other hand, in 2008 blockchain technology has emerged, supposedly written by the pseudonymous Satoshi Nakamoto. This was a technological breakthrough as it provides the first immutable distributed ledger which can be accessed through a peer-to-peer network. Bitcoin blockchain maintains a public ledger by keeping track of all transactions in chronological order, which was decided by an appropriate consensus algorithm. This technology emerged as a boon to the financial industry because peer-to-peer information sharing is most secure and transparent. Additionally, blockchain technology provides many security properties such as immutability, fairness, anonymity, public verifiability and irreversibility.

Though blockchain's original purpose was to keep track of Bitcoin's public ledger its application soon goes beyond the scope of Bitcoin and even cryptocurrency. A good example of such is Ethereum [7]. Ethereum is an "open source, globally decentralized computing infrastructure that executes programs called smart contracts." [1] Also Ethereum is referred to

as "the world computer." [1] Bitcoin contracts are written in a stack-based language, but in Ethereum contracts are written in a Turing complete language, which gives more flexibility in terms of writing smart contracts. Smart contract are programs that can enforce the execution of any protocols.

Thus, in our work, we are focusing on designing blockchain-based e-voting protocols using Ethereum like blockchains.

1.2 Related work

In India, we sometimes heard of tampering with electronic voting machine's security [31]. Even in America's presidential election for the year 2016, there was a problem with electronic voting machine's security [23]. Thus designing an e-voting protocol that can replace a traditional voting system is indeed very challenging. The first e-voting was introduced by David Chaum, dates back to the early eighties, based on public-key cryptography. But the main drawback of that work was it was a centralised voting system. Thus voters have to trust, the central authority for voting. This method fails if the trusted centralized authority gets malicious. In the year 2002, there was a groundbreaking work by Kiayias and Yung [15]. This work was the first theoretical proposal of the decentralised self tallying protocol. Consequently in 2010 Hao et. al. proposed another self tallying protocol, which was more efficient in terms of the number of rounds [12]. But unfortunately for the first implementation of self tallying election, we have to wait for up to 2017. As self tallying elections require a public bulletin board. In 2017 Macorry et. al. has implemented first self tallying election [20] using Ethereum as a public bulletin board. They have implemented the protocol of Hao. et. al. [12].

1.3 Our contribution

We have proposed a decentralized blockchain-based multiparty protocol (see section 6.1.1)for e-voting, where several candidates could be more than two, based on majority counting, by adapting the works of Panja et al. [25] and Hao et al. [12]. In our protocol, all voters provide NIZK (non-interactive zero-knowledge proof) to show that the protocol, executes honestly without revealing the secret votes. Also, we have discussed a new method (see section 6.2.1) for tackling absentee voting and minimized the trust assumption as applicable. In the open vote network (OVN) protocol, [20] a trusted authority was mentioned who creates the eligible voter list. But when some voter outside of the list tries to register illegally, how to prevent such requests was not discussed. We addressed this issue and also provide a solution. Further in the protocol open vote network, the situation when for some voters their committed vote and revealed vote does not match was not covered. We addressed this issue by counting these voters as absentee voters. Below we have given our contributions in tabular format:

	Open Vote Network (OVN) McCorry et al.	Our 1st Protocol	Our 2nd Protocol	Our 3rd Protocol
Blockchain Based	Yes	Yes	Yes	Yes
Absentee Voting	Deposit-Refund Paradigm	No	Yes	Yes
Re-Execution Required	Yes	—	Not required	Not required
Registration Check	No	No	Yes	Yes

Table 1.1: Table Showing Comparisons of Our Protocol

1.4 Organization of the thesis

The rest of the thesis is organized as follows:

In **Chapter 2**, we have discussed the two major papers that we have read. Also, we have discussed the theoretical loopholes. In **Chapter 3**, we have listed and summarized our major contributions. In **Chapter 4**, we gave a gentle introduction to blockchain technology. In **Chapter 5**, we have described implementation details of our simple voting protocol over a private Ethereum network, generated by Ganache. Description of our three modified protocols is given in **Chapter 6**. We have concluded our thesis by mentioning our future work in **Chapter 7**.

Chapter 2

Starting Point: Works of Hao et al.

In a centralized voting system, a group of trusted authorities performs the counting of votes. But the main disadvantage of the centralized voting system is, if any one member of the group has been compromised then the privacy of voters can trivially be broken. Naturally, it brings the need for a decentralized voting system. On the other hand blockchain technology provides a distributed immutable peer-to-peer network, suited for developing decentralized applications. The two most recent works on the decentralized voting systems are Open Vote Network developed by McCorry, Shahandashti, and Hao [20] and its modification by Panja, Bag, Hao, and Roy [25] for supporting Borda count. The open vote network was the first implementation of the self tallying protocol [20]. As they each self tallying protocol requires a public bulletin board, so for a long time there were no public bulletin boards available. With the advent of blockchain technology, a practical implementation was possible. These protocols confirm the most common requirements related to voting viz.

- The anonymity of vote [There should not be any 1-1 correspondence between vote & voter]
- Votes should be recorded as given, change of recorded votes are not feasible and votes are counted as recorded.
- Legitimate voters should not be able to cast multiple votes. False voters [Residents outside of the voting region] should not be able to cast any vote.
- Until the publication of the result any trend related to voting counting should not be leaked.
- The counting process should not be influenced by any single authority.
- Has to confirm the voting machine is starting from "state zero". [By "state zero" we refer to vote counter for every party should be set to 0, at time of starting of voting.]

2.1 Open Vote Network

The first implementation of self tallying decentralized voting protocol, which requires no central authority for vote counting. It is implemented over the Ethereum network. A boardroom scale vote can be conducted with two contestants. The security of this protocol is based on the hardness assumption of discrete logarithm problems in some appropriate groups. It is a two-round protocol. Each voter exhibits their eagerness to vote in the first round and register into the voting protocol. For, each voter registration process consists of, selecting a secret key, also called “voting secret key” and broadcast the public key along with a non-interactive proof of

knowledge of the secret key. This protocol uses "Schnorr zero-knowledge" to show proof of knowledge of "voting secret key" (see appendix **??**). Between the registration phase and voting phase, the smart contract calculates reconstruction keys for each voter. In the second round, each voter broadcasts their votes along with a non-interactive zero-knowledge proof that their vote consists of valid responses. In order to give proper zero-knowledge proofs, this protocol uses CDS method [8]. This protocol is self-tallying any one of the voters or an observer who can calculate the vote.

2.2 Borda Count Smart Contract

This paper [25] modifies the open vote network for supporting Borda counting. In Borda count voting method instead of voting for the candidate of the highest choice, each voter gives ranks for each candidate. The major change in Borda count smart contract from open vote network is that each candidate has to select a separate private key for each candidate in the first round. Also in the second round, each candidate has to transmit a vector of votes and a suitable zero-knowledge proof corresponding to it [i.e. his voting responses are permutation of ranks].

2.3 Advantages and Disadvantages

2.3.1 Advantages and Disadvantages of OVN

Here we will discuss the advantages of disadvantages the open vote network protocol.

Advantages:

This voting system was end-to-end verifiable. An end-to-end verifiable voting system must satisfy three properties.

- Every vote should be cast as intended.
- Every vote should be recorded as cast.
- Every vote should be counted as recorded.

Disadvantages:

Adaptive and Abortive issue:

Each decentralised self tallying protocol has two common issues. These are adaptive and abortive issues. As the protocol is self tallying, the last voter is able to calculate the tally before giving his vote. The **adaptive issue** is, after knowing the tally the final voter can decide his vote in such a way, so that candidate of his choice become victorious and the **abortive issue** is after knowing who could be the possible victorious candidate the last voter stops giving his vote, which will enforce the protocol to be restarted.

The solution to the adaptive issue was already proposed by authors by introducing one additional commitment round, where each voter has to commit their vote before revealing it. But abortive is still remains.

Three solutions were given to the abortive issues.

1. This solution was proposed by Kiayias and Yung [15] and Gorth [10]. It consists of the re-introduction of a trusted authority who will always cast the last vote.

2. This solution was given by Kiayias and Yung [15] and Khader et. al. [14]. It consists of engaging the other voters in an additional round. For key recovery. This method fails once again if any one of the voters does not give their vote.

3. In the realm of blockchain money deposit and refund, the paradigm was introduced in the work of R Kumaresan et. al. [18]. In the third solution, this paradigm was by the authors of the open vote network protocol [20].

Small Scale Issue:

The open vote network is protocol is suited for small scale voting. The nature of the open vote network is decentralized the on the other hand nature of national scale election is centralized. This protocol consists of several rounds of interactions between voters, which is not feasible in large scale voting. Large scale voting requires an excessive amount of time in an open vote network [19]. In today's Ethereum one block is created in 12 seconds. And according to Mccorry et. al. [19] the gas fee requires an open vote network, which makes it fit 1 vote per block. So in a city like Kolkata where the population is almost 15 million, only the vote recording time will require approximately 9.5 years.

2.3.2 Advantage and Disadvantage of Borda Count Smart Contract

Here we will discuss the advantages of disadvantages the open vote network protocol. Borda count smart contract is similar to that of open vote network. It is also a decentralized self tallying protocol. Thus its advantages and disadvantages are similar to the open vote network, and already discussed in the section 2.3.1.

Chapter 3

Summary of the Thesis

In this chapter, we will give a summary of our works. Our first work was an implementation of a simple voting protocol using Ethereum. We have identified the limitations of our simple protocol. Also, we have studied a more secure voting protocol, by McCorry et al. [20]. We have combined these ideas to propose new protocols which can be implementable over the blockchain. We have four major contributions. We describe these four contributions briefly in this chapter, starting from section 3.2.

3.1 Simple Voting Protocol (SVP)

We have implemented a **Simple Voting Protocol** as an Ethereum smart contract. We have deployed the protocol over the private test network, generated by Ganache. Details of this protocol are given in the chapter 5.

3.2 Extension to Multiple candidate voting

This is joint work with **Mr. Manish Vuppala**, former MTech scholar of IIT Bhilai. We have non-blockchain e-voting multiparty protocols which support several candidates to compete in voting via majority counting [12]. On the other hand, we have a blockchain-based e-voting protocol where two candidates can compete in the vote [20]. But the blockchain-based e-voting protocol where several candidates can compete was missing (in majority based counting). We have designed such a protocol. For this, we have adapted the idea from Panja et. al. where blockchain-based e-voting was described in the Borda count setup and with multiple candidates. We have shown the available literature and our protocol in the figure 3.1. **Note:** in picture k is denoting the number of candidates in the voting protocol. $k = 1$ is denoting the vote on a particular question or memorandum.

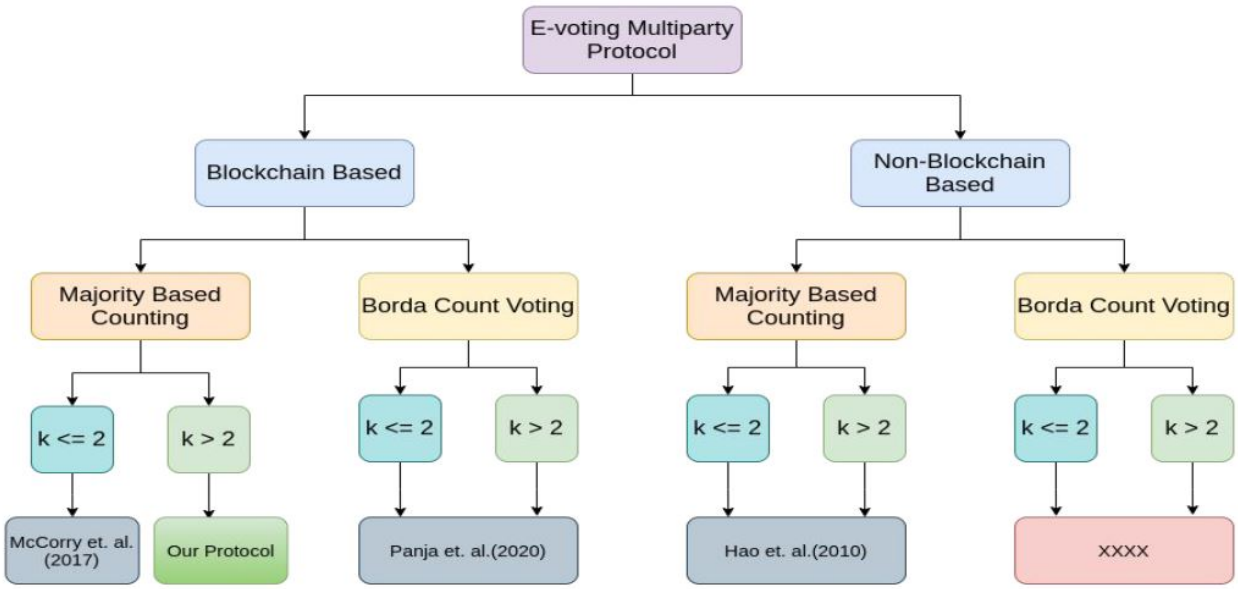

Figure 3.1: Landscape of various E-voting Multiparty Protocols [Picture courtesy **Mr. Manish Vuppala**]

3.3 Improved registration

In the open vote network smart contract there is a trusted party who creates a list of eligible voters [20]. But it might be the case some malicious voters who are not on the list tries to register into the voting protocol. But countermeasure to this problem was not addressed in the open vote network protocol. We have addressed this issue in the section 6.2.1.

3.4 Improved commitment verification

In the open vote network to prevent adaptive attacks, an additional commitment round was proposed [20]. But there may be cases where for some voters there committed vote and revealed vote might not match. How to handle these situations was not discussed in the open vote network. We propose to count these voters for a whose committed vote and revealed vote does not match as absentee voters. Thus we extended the definition of absentee voters to deal with this problem. According to our definition absentee voters are as defined as follows:

New definition of Absentee voters

According to our protocol, we define absentee voters as:

1. Those voters who have registered into the voting protocol vote but did not publish commitment of their vote before $T_{EndCommit}$.
2. Those voters who have registered into the voting system published their committed vote before $T_{EndCommit}$ but did not publish their vote before $T_{EndVote}$.
3. In addition, we will categorize those voters as absentee voters, whose committed vote does not match with the revelled vote.

3.5 Absentee voting

In the protocol **open vote network** [20] absentee voter was a major issue because, reconstructed keys are computed in such a way that $\sum_j x_{ij} y_{ij} = 0$, but if any voter does not give his vote within time, the vote can not be evaluated. Which leads to repeating the whole protocol. Thus we addressed this issue also and proposed a new method to deal with this issue. We would like to add a note that **we have addressed this absentee issue while using the new extended definition of absentee voters**. Below we are giving a short description of our method to deal with absentee voting. A detailed discussion of the whole protocol is given in the section 6.2.1 and section 6.3.1, respectively.

Brief discussion on our method to deal with absentee voting

As a solution, we propose that the election commissioner (we assume he is a trusted person) will own a public key private key pair. Each voter has to broadcast an encrypted version of their "voting secret key" (encrypted with the election commissioner's public key) when they register for voting. As the election commissioner is trusted he will not open any of the encrypted versions of "voting secret keys" unless any voter is absentee. After the voting is over [ie after $T_{EndVote}$] smart contract can easily detect absentee voters, by checking the presence and absence of commitment and votes which were published before appropriate time viz. $T_{EndCommit}$ and $T_{EndVote}$ respectively. It is possible since all votes were broadcasted into the blockchain. Also, there might be a case some voters give vote after the legitimate times. Accepting votes by checking timestamps is an efficient way to do so. Once the smart contract creates the list of absentee voters it broadcasts the list. The election commissioner decrypts the "voting secret keys" of absentee voters, gives a zero vote on behalf of them. By zero vote we mean that $v_i = 0_k$, if i^{th} voter is an absentee voter, 0_k is zero vector of k-dimension. We are using the El Gamal encryption scheme to publish the encryption version of the secret key of voters here. Details of this protocol are given in the section 6.2.1. Also, we have reduced the trust assumption, in a subsequent protocol, which is discussed in the section 6.3.1.

Chapter 4

Blockchain Basics

In 2008 first-ever decentralized currency exchange system, Bitcoin came into the existence [22]. At its heart, there was blockchain technology. The blockchain is a peer-to-peer public ledger, which maintains a record of all previous transactions with help of an appropriate consensus mechanism in chronological order. It inherently provides security properties, such as anonymity, immutability, and irreversibility. One may say it is the most groundbreaking technology of the previous decade. It reduces the dependency on a single centralized trusted authority. Thus it has no single point of failure. The principal chain of a blockchain is organised in a linear manner, with the first block being the genesis block. Each block in the blockchain is marked with a cryptographic signature, also it preserves the retention property of all transactions. A transaction on its creation is broadcasted to the entire peer-to-peer network. The networks nodes check the validity of the transaction using an appropriate consensus mechanism. This transaction is included in a newly proposed block. Finally, the block is appended to the blockchain. This process is shown in Figure 4.1.

4.1 History of Currencies: From Barter System to Cryptocurrency

The world has experienced different currency systems. The **barter system** was the oldest among all. This system does not require cash. Usually at the time of business peoples used to negotiate and make an exchange of the products. **This scheme has many limitations**,

1. There may be different demands for different goods.
2. Might be disagreement in exchange value.
3. Finding a trustworthy person to trade with.
4. Unavailability of necessary products.
5. Lack of insurance and warranty on goods.

To overcome one of the above issues, in ancient times coins made of gold, silver and copper were used for trading. Each coin contains a fixed amount of precious metal. Thus there value used to remain fixed. In the medieval era, fiat currencies came through different governments. Governments regulate fiat currencies. Fiat money acts as the nation's economy and the worth of the currency is dictated by the relationship between supply and demand, which is still in effect [6].

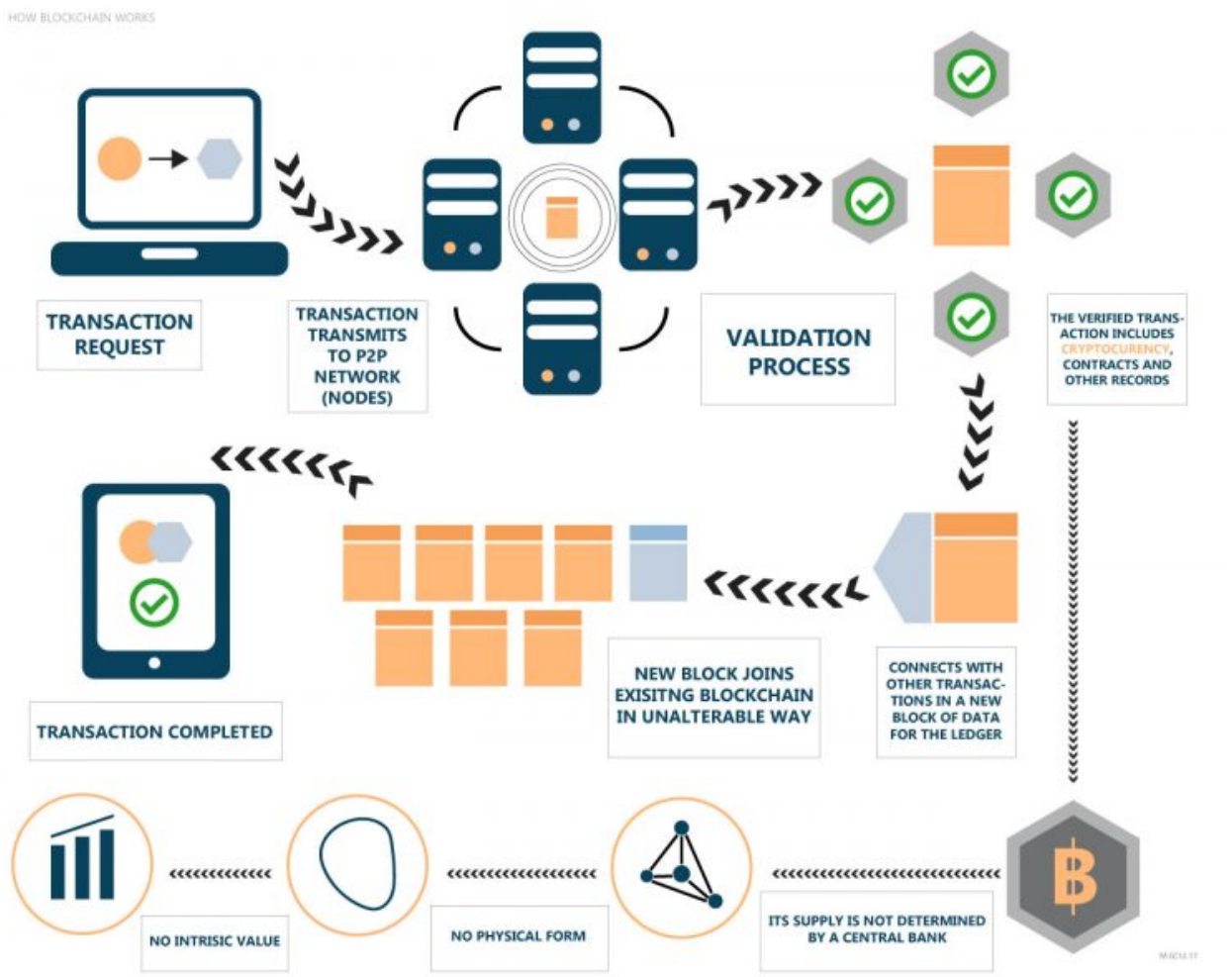

Figure 4.1: Blockchain workflow [5]

The Reserve Bank of India has approved the use of corporate tokens e.g. **Sodexo vouchers** for employee welfare, bonus appreciation and performance rewards. The advantage of these tokens is tax breaks while limited validity and acceptance by a limited number of stores are its drawbacks.

Nowadays online banking system is an efficient way to transfer money. Online banking systems include **debit cards**, **credit cards**, **internet banking**, **Real Time Gross Settlement (RTGS)**, **National Electronic Funds Transfer (NEFT)** and other banking applications like Pay Through Mobile (PayTM), Google Pay (GPay) etc which use Unified Payments Interface (UPI) [11]. However third party transactions through internet banking sometimes lead to security concerns.

Bitcoin is the first peer-to-peer and decentralized digital currency first published as a Bitcoin white paper, is implemented over the blockchain [22]. Unlike fiat currencies, it does not require any approval from the government and financial institutions. It can be used globally. The total supply of bitcoin is finite, 21 million only. It offers anonymity also double spending problem has been taken care of, which occurs when the same money has been spending twice. Bitcoin's components include cryptography, hardware, software and game theory. Bitcoin is run on the hardware of thousands of miners across the world by installing the bitcoin software on their devices.

Ethereum, is a cryptocurrency with multipurpose production environments that appeared as a white paper in 2014. Ethereum's programming language is Turing complete. It is designed as a global computer that can execute programs called smart contracts. Smart contracts can enforce automated services that are written into the blockchain and are permanent [7]. Once published, they cannot be deleted or altered thus they will continue to work normally, autonomously, and transparently in the absence of external stimuli.

4.2 Mining in Bitcoin

Miners create new blocks in Bitcoin. In Bitcoin, a block contains all the transactions.

- A user creates a new transaction and broadcasts it to the network.
- That generated transaction creates a cartographic request and waits in a pool of unconfirmed transactions on the blockchain network for confirmation.
- Miner nodes on the network choose transactions from these pools and combine them into a block.
- In Bitcoin, on average a new block is created in 10 minutes. To propose a new block a miner has to solve the hash puzzle. Miners use high computing capacity to solve the problem.
- In Bitcoin, any miner can compete to solve the hash puzzle.
- If a miner is able to solve the hash puzzle he publishes the solution. Other miners validate the result.
- A block is added to the blockchain, and the miner gets his reward. Currently, it is 6.25 Bitcoin per block.

4.3 Consensus Mechanisms

In this section, we will discuss blockchain consensus mechanisms in brief. In blockchain by using a consensus algorithm, blockchain users agree upon the current state of the blockchain without any help from the central trusted authority. There are many popular consensus algorithms that are used in different blockchains to meet different purposes. Some of the widely used consensus algorithms are Proof of Work, Proof of Stake, Proof of Authority, Proof of Elapsed Time and Byzantine Fault Tolerance (BFT).

Proof of Work (PoW): PoW consensus algorithm is used in public blockchain where unique nodes (also called miners) use loads of tools such as costly hardware, application-specific integrated circuits and energy, to solve a problem and create a block. After this block is created, it is applied to the public blockchain and the respective miner is credited with cryptocurrency [3]. **This method is not eco-friendly** as it uses lots of physical energy.

Proof of Stake (PoS): This consensus mechanism has unique nodes known as validators, and the validator for each block is selected in a deterministic manner depending on the skate he holds in the network [3]. PoS serves only the interests of the network's main owners. In the event of a disagreement in the consensus, the main stakeholder assumes the position of supreme

commander, and a 51 percent attack is unlikely. PeerCoin [17] is the example of Proof of Stake.

Proof of Authority (PoA): In this consensus mechanism, only approved nodes are able to validate blocks, and they are incentivized to secure the network. Unlike in PoS, where numerical worth is at stake, identity is at stake in PoA. To build a block, the Ethereum testnets Kovan [16] and Rinkeby [27] use the PoA consensus algorithm. PoA networks are distinguished by their use of fewer computing resources and the absence of the need for nodes to connect in order to achieve a consensus. PoA is ideal for private Ethereum blockchains in controlled organisations with centralized trustworthy authorities.

Proof of Elapsed Time (PoET): In this consensus mechanism, all nodes in the blockchain network will be in a waiting state for a random period of time, and the node that finishes the waiting time first will be the block maker. The Proof of Elapsed Time consensus algorithm is used by Hyperledger Sawtooth [24].

Byzantine Fault Tolerance (BFT): A device in BFT is distinguished by its ability to withstand failures categorised as the Byzantine Generals Problem. Byzantine failure may occur as a result of a faulty node in the system or as a result of an intentional attack by a dishonest node in the system. The dishonest node is someone who sends contradictory computations to their peers, resulting in an inaccurate outcome of the equation that the whole distributive mechanism is attempting to execute. The usage of proof of work to solve the Byzantine Generals Problem probabilistically was a significant achievement when Bitcoin was conceived.

4.4 Overview of Basic Cryptographic Primitives Used in Blockchain

In this section, we will discuss cryptographic primitives which were used in blockchain.

Public-Key Cryptography: A key pair also known as the public key and secret key pair is used in public-key cryptography. The public key as the name suggests can be revealed to anyone while the user has only access to the secret key. The public key is used for encryption purposes and the secret key is used for decryption purposes. Public key cryptography is also known as an asymmetric key encryption scheme.

Digital Signature: Purpose of the digital signature scheme is to authenticate messages. In this, the sender signs the message with his secret key. The message is verified by the receiver using the sender's public key. To guarantee the legitimacy of a digital contract, Blockchain employs digital signatures [3].

Hashing: In Blockchain, a hashing algorithm is used to determine the hash value of the transaction, which results in a fixed-length stream. Bitcoin, employs the SHA-256 hashing algorithm [3]. The unique properties of hashing include the following: collisions resistance (it is difficult to generate the same hash value for different inputs), compression, (the output hash value is often of fixed size even though the input value is greater than the output value) and pre-image tolerance, (determining the input value from the output is difficult).

Merkle tree is a binary tree. It uses a mixture of leaf nodes, intermediate nodes, and root nodes generated by cryptographic hash functions. In blockchain applications at the leaf nodes, transaction data are stored. Followed by a series of intermediate nodes generated by

calculating the hash value of the two child nodes, and finally, the Merkle root node is obtained by calculating the hash of its two child nodes, which generate the top of the tree [3]. Merkle trees decisively prove the authenticity of the data and can be regarded as the hallmark of all transactions that comprise a block. The efficiency of the Merkle tree lies in low computation time and low space complexity.

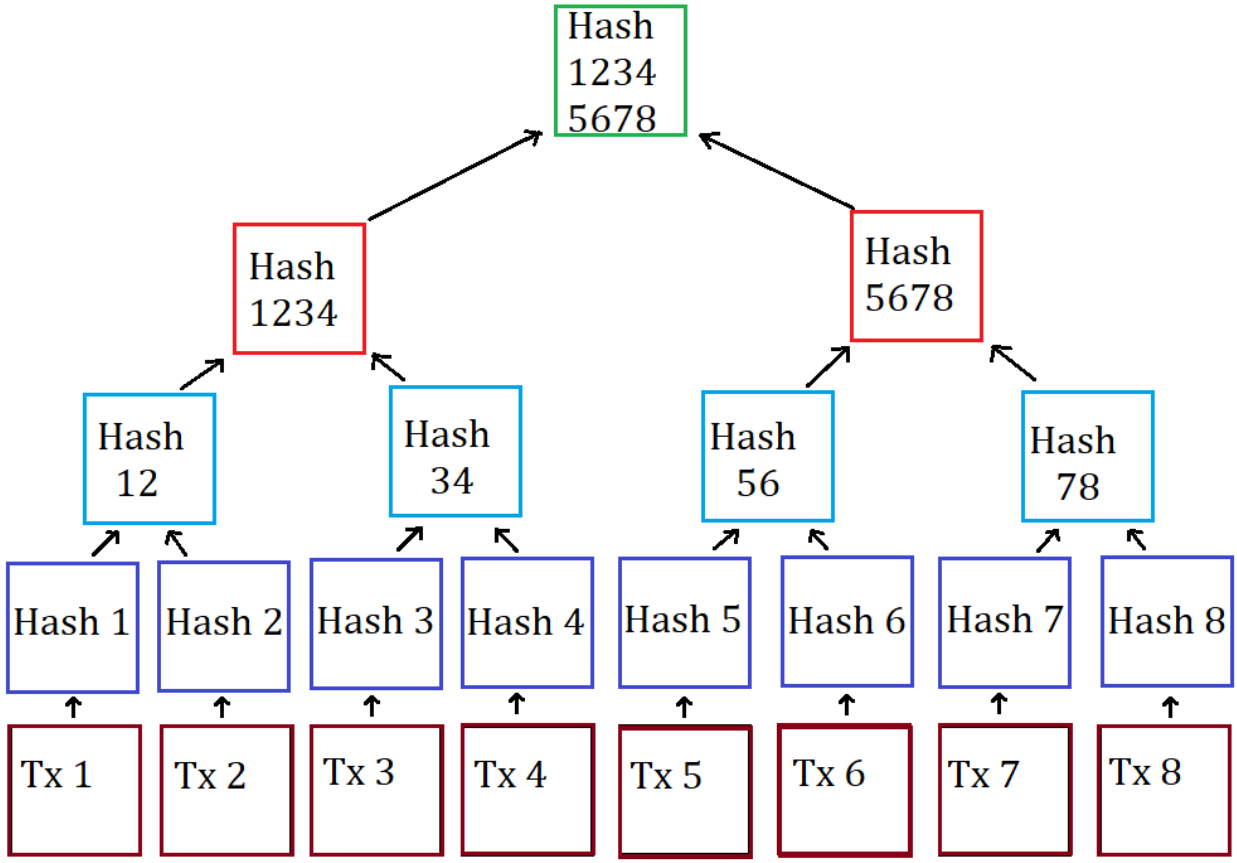

Figure 4.2: Merkle tree [21]

Types of Blockchain

A. Permissioned and Permissionless Blockchain

Blockchain networks are widely divided into permissioned and permissionless networks.

Permissioned Blockchain: Permissioned blockchains have an access control layer that requires only approved nodes to verify transactions in a block and only restricted nodes to contribute to network consensus, as well as limiting actors' ability to create smart contracts [2]. The inherent configuration of permissioned blockchain is built uniquely. Since only the approved nodes are involved in mining high end computing facilities are not required for mining and to achieve consensus. Permissioned blockchain uses consensus algorithms such as RAFT [13] or PAXOS [9] and is chosen for a variety of reasons including anonymity, scalability, and access control. Examples of permissioned blockchain are Ripple and R3 CORDA [4].

Permissionless Blockchain: Permissionless blockchain there are no restrictions on accessing the network or validating the transactions in a block [2]. Furthermore, everyone can build a consensus and a smart contract. Identity proof is not needed in permissionless blockchian, but it does require computing capacity to expands the blockchain network. The Bitcoin and

Ethereum blockchain networks are examples of permissionless blockchains, where anybody can join the network and begin mining. Miners that solves the cryptographic puzzle first and validates the transactions will get the mining reward.

B. Public, Private and Consortium Blockchain

Blockchain is divided into three categories: public blockchain, private blockchain, and consortium blockchain. [2].

Public Blockchain: The public blockchain is a permissionless blockchain in which transactions are open to anyone in public and everyone can run a complete node, read, write, validate, and add a new block to the blockchain [3]. Consensus mechanisms such as PoW and PoS are used to make decisions in these networks.

Private Blockchain: A private blockchain is a permissioned blockchain that is operated exclusively by a person or a entity. Running a complete node on a private blockchain is restricted. Everyone is allowed to execute, check, or inspect transactions on the blockchain [3].

Consortium Blockchain: Consortium blockchains are formed when a group of organisations collaborate to form a blockchain network and make decisions that favour the entire network [2]. These groups are known as consortiums or federated blockchains, and they are managed by consortium members only.

Chapter 5

Proof of Concept Ethereum Implementation of a Simple Voting Protocol (SVP)

This semester we have implemented an e-voting system viz. Simple Voting Protocol (SVP). This was a basic implementation. The technologies used as follows:

- Node.JS [Ethereum javascript API]
- Solidity [For writing smart contract]
- Truffle [Solidity compiler]
- Ganache [Private Ethereum like blockchain network generator]
- Metamask [Connets browser to local private blockchain network]
- We have implemented it in ubuntu 18.04. Which was run in a virtual box, host OS was windows 10.

5.1 Description of SVP

- At first Ganache, GUI is launched to create a private Ethereum network.
- Then deploy smart contract with the truffle framework.
- After we launch the application through the browser and imports an account.
- If the smart contract detects that the imported account is not voted, it allows the candidate to vote.
- After the user cast his vote, his vote as a transaction broadcasted into the blockchain.

In our protocol before voting:

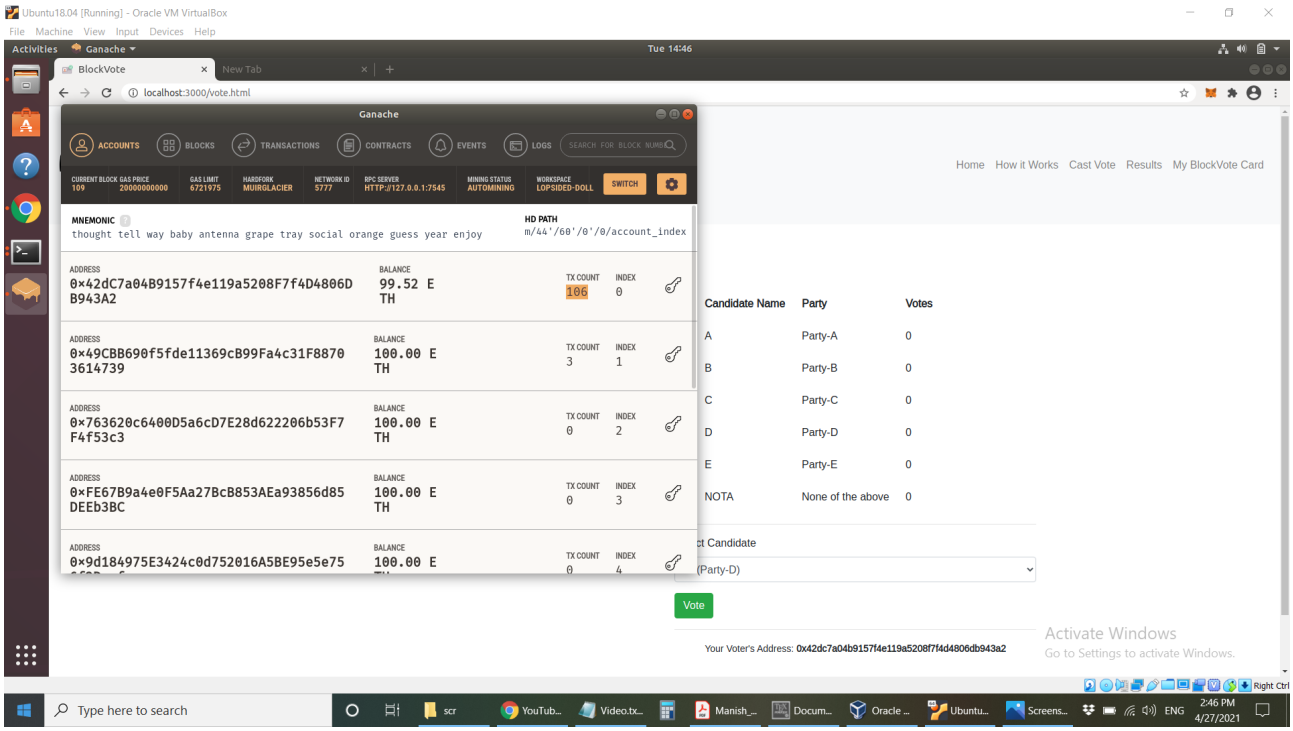

Figure 5.1: Before Voting

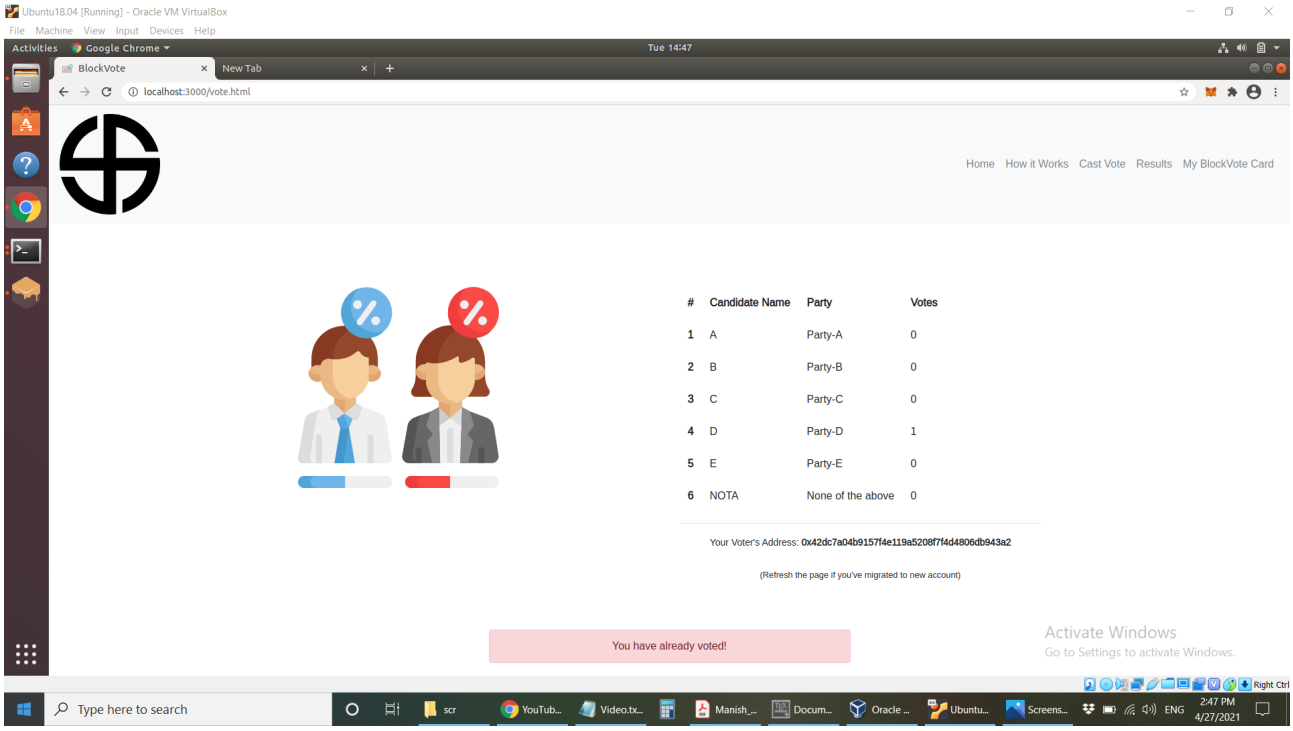

Figure 5.2: After Voting

5.2 Limitations of our Simple Voting Protocol (SVP)

In our implementation, the result can be seen in real-time. So the candidates may try to influence the voters. The votes are not in encrypted form.

Chapter 6

A New and Secure E-voting Protocol

Here we will discuss our modified e-voting protocol. In our protocol, we need a trusted person[1] called the election commissioner, responsible for creating a voters list. We have proposed extension in the following directions,

1. The extension to multiple candidates in plurality-based voting. [This is joint work with **Mr. Manish Vuppala**, former MTech scholar of IIT Bhilai.] To do so we have adapted the design of the Borda count smart contract [25]. Instead of taking rank-based voting, we have used binary responses for each of the candidates.

2. We have proposed two different methods to deal with absentee voting.

3. We have raised an issue (see section 2.3) related to registration in the open vote network protocol, here we have tried to address this issue.

4. Also, we have extended the definition of absentee voters, and designed our protocol according to the new definition (given in section 3.5)

first in the direction of a number of parties and second in terms of absentee voting.

6.1 Our first protocol

Our first protocol is the extension open vote network into multiple candidate settings.

6.1.1 Description of our first protocol

Let G be a finite cyclic group of order q and with generator g. We assume that the discrete logarithm problem is hard. Our voting protocol consisting of five phases.

Setup Phase

1. [Broadcast] Election commissioner Creates a list of eligible voters. Voters whose names are on the list will be able to register to vote. [Currently, we are assuming that no voters outside of the list trying to register into the system.]

2. [Broadcast] Specifies some time limits.

   - T_{EndReg} : Voters must complete registration within this time limit.

[1] We like to recall that, the concept of using a trusted person creating a voters list, was present also in the work open vote network. There the term election administrator was used. In our protocol, we have extended the role of election administrator and also we are referring to him as election commissioner.

- $T_{BeginVote}$: Voters will be able to publish commitments of their vote after this time.
- $T_{EndCommit}$: Voters should finish publishing commitments of their vote before this time.
- $T_{EndVote}$: Voters should cast their vote before this time.

Registration Phase

1. [Local Computation] Each voter V_i draws $x_{ij} \xleftarrow{\$} G \setminus \{e\}$ $\forall j \in \{1, 2, ..., k\}$, where e is the identity element of G.
2. [Local Computation] Each voter V_i calculates $g^{x_{ij}}$ $\forall j \in \{1, 2, ..., k\}$.
3. [Broadcast] Each voter V_i broadcast $g^{x_{ij}}$ along with a proof of knowledge of x_{ij} $\forall j \in \{1, 2, ..., k\}$.
4. [Local Computation] Each voter V_i will calculate the reconstruction key

$$g^{y_{ij}} = \frac{\prod_{l=1}^{i-1} g^{x_{ij}}}{\prod_{l=i-1}^{k} g^{x_{ij}}}$$

Commitment Phase

1. [Local Computation] Each voter V_i computes his vote in the form $v_i = (v_{i1}, v_{i2}, ..., v_{ik})$, where $v_{ij} \in \{0, 1\}$ and $\sum_j v_{ij} = 1$.
2. [Local Computation] Each voter V_i computes $Z_{ij} = g^{x_{ij}y_{ij}} g^{v_{ij}}$ $\forall j \in \{1, 2, ..., k\}$
3. [Broadcast] Each voter V_i broadcasts commitments of Z_{ij}, we denote it by $COM(Z_{ij})$.

Voting Phase

1. [Broadcast] Each voter V_i broadcasts Z_{ij}, along with a zero knowledge proof that $v_{ij} \in \{0, 1\}$ $\forall j \in \{1, 2, ..., k\}$ and $\sum_j v_{ij} = 1$.

Self-tallying Phase

1. Since all votes have been broadcasted, so any of the voters can compute the results. As follows,

$$\prod_{l=1}^{n} g^{x_{ij}y_{ij}} g^{v_{ij}} = g^{\sum_j v_{ij}}$$

as $\sum_j x_{ij}y_{ij} = 0$

6.2 Our second protocol

This protocol is an extension of our first protocol. In this protocol we have addressed the registration issue, incorporated the new definition of absentee voters, also we have tackled the absentee vote problem. This protocol is based on **honesty assumption of election commissioner**.

6.2.1 Description of our second protocol

Let G be a finite cyclic group of order q and with generator g. We assume that the discrete logarithm problem is hard. Our voting protocol consisting of six phases.

Setup Phase

1. [Broadcast] Election commissioner Creates a list of eligible voters. Voters whose names are on the list will be able to register to vote.

2. [Broadcast] Election commissioner specifies some time limits.

   - T_{EndReg} : Voters must complete registration within this time limit.
   - $T_{BeginVote}$: Voters will be able to publish commitments of their vote after this time.
   - $T_{EndCommit}$: Voters should finish publishing commitments of their vote before this time.
   - $T_{EndVote}$: Voters should cast their vote before this time.
   - $T_{EndCheck}$: Smart contract will finish check for absentee voting before this time and will broadcast a list of absentee voters.
   - $T_{CastDummyVote}$: Election commissioner will cast zero on behalf of absentee voters.

3. [Broadcast] Election commissioner broadcast his public key (G, g, q, h). For this he, draws $x \xleftarrow{\$} G \setminus \{e\}$, where e is the identity element of G, and computes $h := g^x$.

Registration Phase

1. (a) [Local Computation] Each voter V_i draws $x_{ij} \xleftarrow{\$} G \setminus \{e\}$ $\forall j \in \{1, 2, ..., k\}$, where e is the identity element of G.

   (b) [Local Computation] Each voter V_i calculates $g^{x_{ij}}$ $\forall j \in \{1, 2, ..., k\}$.

2. [Broadcast] Each voter V_i creates a transaction puts the following into the data section of that transaction,

   (a) $g^{x_{ij}}$ along with a proof of knowledge of x_{ij} $\forall j \in \{1, 2, ..., k\}$.

   (b) $Enc_{pk_{EC}}[x_{ij}]$ $\forall j \in \{1, 2, ..., k\}$. [each x_{ij} is being encrypted with public key of the election commissioner, $\forall j \in \{1, 2, ..., k\}$]

   and signs it by his secret key [his blockchains wallets secret key], along with he pledges some money on behalf of the smart contract. Before accepting the input of the transaction, the smart contract will verify whether it is signed by a listed voter otherwise smart contract will not allow the owner of the transaction to register to vote.

3. [Local Computation] Each voter V_i will calculate the reconstruction key

$$g^{y_{ij}} = \frac{\prod_{l=1}^{i-1} g^{x_{ij}}}{\prod_{l=i-1}^{k} g^{x_{ij}}}$$

Commitment Phase

1. (a) [Local Computation] Each voter V_i computes his vote in the form $v_i = (v_{i1}, v_{i2}, ..., v_{ik})$, where $v_{ij} \in \{0, 1\}$ and $\sum_j v_{ij} = 1$.
   (b) [Local Computation] Each voter V_i computes $Z_{ij} = g^{x_{ij}y_{ij}} g^{v_{ij}} \quad \forall j \in \{1, 2, ..., k\}$
2. [Broadcast] Each voter V_i broadcasts commitments of Z_{ij}, we denote it by $COM(Z_{ij})$.

Voting Phase

1. [Broadcast] Each voter V_i broadcasts Z_{ij}, along with a zero knowledge proof that $v_{ij} \in \{0, 1\}$ $\forall j \in \{1, 2, ..., k\}$ and $\sum_j v_{ij} = 1$.

Absentee Detection and Refund Phase[2]

1. [Broadcast] After $T_{EndVote}$ smart contract checks for absentee voter. Publishes a list of absentee voter before $T_{EndCheck}$. Since all votes have been broadcasted into the blockchain, there is the timestamp when the block is created, smart contract utilizes this timing for checking purpose, thus it discards any votes which was cast outside of the allowed interval.
2. Non-absentee voters get back their pledged money and pledged money of absentee voters can be used to incentivize honest voters or it can be donated to charity.
3. [Broadcast] Election commissioner decrypt the "voting secret keys" of absentee voters according to the list and then broadcast zero votes on behalf of absentee voters. Also, the election commissioner has to give zero-knowledge proof of his vote.

Self-tallying Phase

1. Smart contract can compute the tally, (by counting votes given voters, and votes given election commissioner in case of absentee) as follows,

$$\prod_{l=1}^{n} g^{x_{ij}y_{ij}} g^{v_{ij}} = g^{\sum_j v_{ij}}$$

   as $\sum_j x_{ij}y_{ij} = 0$
   Since all votes have been broadcasted, so any of the voters [even any observer] can verify the published results.

6.2.2 Advantages and disadvantages of the protocol

Advantages:

1. It saves time in case of absentee voting, as protocol need not be repeated.

Disadvantages:

1. If the election commissioner is not trusted then the privacy of voters will be lost.

Note: Malicious election commissioner will not be able to change the votes because the list of absentee voters is published by the smart contract. So even if the malicious election commissioner publishes forged votes he can not misguide any person who is counting votes.

[2]We have put this phase after voting phase because unless voting has been done it is not possible to predict absentee behaviour.

6.2.3 A modification on our second protocol

An obvious improvement over our second protocol could be the use of Shamir's secret sharing scheme [29]. Here we are giving only the textual description.

- Instead of using a single trusted election commissioner, we could have an election committee of n members.
- Each member of this election committee will own a secret key and public key pair.
- Each voter will split his secret key into n shares, if d ($d < n$) shares combined the secret will be reviled.
- The voter sends encrypted [with corresponding secret key] shares to each member of the committee.

Advantages:

1. A single malicious election commissioner will not able to break the privacy of the voters.

Disadvantages:

1. This increases the communication complexity of the protocol.
2. If d or more election committee members join hands together, they can break voter privacy.

6.3 Our third protocol

Before going into our third protocol, let us discuss the issue of voters' privacy loss, which was raised in our second protocol (see section 6.2.1 and section 6.2.2). The issue was as follows. The election commissioner has the voting secret key of each voter. Thus a corrupted election commissioner can decrypt and get the voting secret key of each voter. This can result in the privacy loss of voters as their choice is revealed to the election commissioner. Furthermore, the election commissioner can calculate the trend of the vote. Finally, he also can bribe other voters to make his preferred candidate victorious. To tackle this situation we have proposed a $TimeLock(*,*)$ function **in black box manner**. It functions as follows, if election commissioner have $TimeLock(Enc_{pk_{EC}}[x_{ij}], T_{EndVote})$, then after time $T_{EndVote}$ he will get $Enc_{pk_{EC}}[x_{ij}]$ out of the time lock. We would like to make a note here, that **so far we do not known any practical implementation of such time lock function**. Therefore this protocol is completely theoretical.

6.3.1 Description of our third protocol

Let G be a finite cyclic group of order q and with generator g. We assume that the discrete logarithm problem is hard. Our voting protocol consisting of six phases.

Setup Phase

1. [Broadcast] Election commissioner Creates a list of eligible voters. Voters whose names are on the list will be able to register to vote.

2. [Broadcast] Election commissioner specifies some time limits.

   - T_{EndReg} : Voters must complete registration within this time limit.
   - $T_{BeginVote}$: Voters will be able to publish commitments of their vote after this time.
   - $T_{EndCommit}$: Voters should finish publishing commitments of their vote before this time.
   - $T_{EndVote}$: Voters should cast their vote before this time.
   - $T_{EndCheck}$: Smart contract will finish check for absentee voting before this time and will broadcast a list of absentee voters.
   - $T_{CastDummyVote}$: Election commissioner will cast zero on behalf of absentee voters before this time.

3. [Broadcast] Election commissioner broadcast his public key (G, g, q, h). For this he, draws $x \xleftarrow{\$} G \setminus \{e\}$, where e is the identity element of G, and computes $h := g^x$.

Registration Phase

1. (a) [Local Computation] Each voter V_i draws $x_{ij} \xleftarrow{\$} G \setminus \{e\}$ $\forall j \in \{1, 2, ..., k\}$, where e is the identity element of G.

   (b) [Local Computation] Each voter V_i calculates $g^{x_{ij}}$ $\forall j \in \{1, 2, ..., k\}$.

2. [Broadcast] Each voter V_i creates a transaction puts the following into the data section of that transaction,

   (a) $g^{x_{ij}}$ along with a proof of knowledge of x_{ij} $\forall j \in \{1, 2, ..., k\}$.

   (b) $TimeLock(Enc_{pk_{EC}}[x_{ij}], T_{EndVote})$ $\forall j \in \{1, 2, ..., k\}$. [each x_{ij} is being encrypted with public key of the election commissioner, $\forall j \in \{1, 2, ..., k\}$]

   and signs it by his secret key [his blockchains wallets secret key], along with he pledges some money on behalf of the smart contract. Before accepting the input of the transaction, the smart contract will verify whether it is signed by a listed voter otherwise smart contract will not allow the owner of the transaction to register to vote.

3. [Local Computation] Each voter V_i will calculate the reconstruction key

$$g^{y_{ij}} = \frac{\prod_{l=1}^{i-1} g^{x_{ij}}}{\prod_{l=i-1}^{k} g^{x_{ij}}}$$

Commitment Phase

1. (a) [Local Computation] Each voter V_i computes his vote in the form $v_i = (v_{i1}, v_{i2}, ..., v_{ik})$, where $v_{ij} \in \{0, 1\}$ and $\sum_j v_{ij} = 1$.

   (b) [Local Computation] Each voter V_i computes $Z_{ij} = g^{x_{ij} y_{ij}} g^{v_{ij}}$ $\forall j \in \{1, 2, ..., k\}$

2. [Broadcast] Each voter V_i broadcasts commitments of Z_{ij}, we denote it by $COM(Z_{ij})$.

Voting Phase

1. [Broadcast] Each voter V_i broadcasts Z_{ij}, along with a zero knowledge proof that $v_{ij} \in \{0,1\}\ \forall j \in \{1,2,...,k\}$ and $\sum_j v_{ij} = 1$.

Absentee Detection and Refund Phase[3]

1. [Broadcast] After $T_{EndVote}$ smart contract checks for absentee voter. Publishes a list of absentee voter before $T_{EndCheck}$. Since all votes have been broadcasted into the blockchain, there is the timestamp when the block is created, smart contract utilizes this timing for checking purpose, thus it discards any votes which was cast outside of the allowed interval.
2. [Broadcast] Non-absentee voters get back their pledged money and pledged money of absentee voters can be used to incentivize honest voters or it can be donated to charity.
3. [Local computation] Election commissioner retrieves encrypted "voting secret keys", out of the $TimeLock$.
4. [Broadcast] Election commissioner decrypt the "voting secret keys" of absentee voters according to the list and then broadcast zero votes on behalf of absentee voters. Also, the election commissioner has to give zero-knowledge proof of his vote.

Self-tallying Phase

1. Smart contract can compute the tally, (by counting votes given voters, and votes given election commissioner in case of absentee) as follows,

$$\prod_{l=1}^{n} g^{x_{ij}y_{ij}} g^{v_{ij}} = g^{\sum_j v_{ij}}$$

as $\sum_j x_{ij}y_{ij} = 0$.
Since all votes have been broadcasted, so any of the voters [even any observer] can verify the published results.

6.3.2 Advantages and disadvantages of the protocol

In this protocol, we have successfully reduced the trust assumption on the election commissioner. The advantages of this protocol over our second protocol are as follows,

Advantages:

1. It saves time in case of absentee voting, as protocol need not be repeated.
2. Here we have used $TimeLock(*,*)$ function which will prevents election commissioner to obtain encrypted secret by unlocking, before $T_{EndVote}$. Thus here before $T_{EndVote}$ there is no voter privacy loos.
3. As a consequence of the fact election commissioner can not decrypt before $T_{EndVote}$, there is no loss in the voting trend.

Disadvantages:

1. After voting is over a malicious election commissioner can know the votes.

[3]We have put this phase after voting phase because unless voting has been done it is not possible to predict absentee behaviour.

Chapter 7

Conclusion And Future Work

We have studied the various problems on e-voting, also we have studied two most recent papers by McCorry, Shahandashti, and Hao [20] and by Panja, Bag, Hao, and Roy [25] on this topic. These two papers are on decentralized blockchain-based e-voting. We also studied the basics of blockchain. We also have implemented a basic blockchain-based e-voting protocol. Also, we have proposed a solution for absentee voting and proposed a multiple candidates setting for **open vote network**.

In the future, we would like to extend our work from both theoretical and implementation-related perspectives. We have listed our future tasks as follows:

(A) The first step of the voting phase of the protocol described in (see section 6.1.1), requires a voter to publish a zero-knowledge proof that $v_{ij} \in \{0, 1\}$ and $\sum_j v_{ij} = 1$, we would like to device such proof. Here we have used it in a black-box manner. Also we would like to add another zero-knowledge proof in the first step of self-tallying phase, to grantee $v_{ij} \in \{0, 1\}$ and $\sum_j v_{ij} = 0$.

(B) We would like to implement the protocol we have described in the (see section 6.3.1). A challenging task will be implementing the time-lock condition that we have proposed because we are trying to time-lock some data using blockchain. We would like to explore new features of Ethereum for this purpose.

(C) We would also like to modify our third protocol. There the problem was the **voter privacy loss after end of voting**. Here we are giving the currently the two ideas we have, so far

Idea 1

We would add another a third argument, which is essentially an indicator variable in our $TimeLock()$ function. Thus three arguments are, $TimeLock(LockValue, TimeBound, Indicator)$. And condition will be no locked value can be recovered before $TimeBound$ and after the time bound only certain values can be recovered specified by the Indicator.

Advantages: A direct advantage of this approach, is that this approach can solve the absentee voter problem entirely.

But implementation of this function will be hard. Currently we do not know how to implement such function.

Idea 2

A new direction in the blockchain literature is of mutable blockchain [26]. Here our rational is to use mutable blockchains, as a implementation platform of our protocol. Then delete the $TimeLock(Enc_{pk_{EC}}[x_{ij}], T_{EndVote})$ as soon has the voter gives vote.

Appendix A

Schnorr zero-knowledge proof

Here we will briefly discuss the two zero-knowledge proofs that were used in the voting protocol open vote network. [20]. This protocol is used for giving proof of knowledge of discrete log problem. The protocol was given by C.P. Schnorr [28]. Now consider two parties prover and verifier. Let us consider a group G of prime order q and is generator is being g. This is the common input to both prover and verifier. Let us consider that prover chooses $x \xleftarrow{\$} G \setminus \{e\}$, where e is the identity element of group G and then sends $h = g^x$ to the verifier. Using Schnorr's protocol a prover can give the proof knowledge of x to the verifier. The protocol is as follows.

1. [Local computation] Prover chooses a $r_1 \xleftarrow{\$} G \setminus \{e\}$. Computes $a = g^{r_1}$.
2. [Off chain] Prover sends a to the verifier.
3. [Local computation] Verifier chooses a $r_2 \xleftarrow{\$} G \setminus \{e\}$.
4. [Off chain] Verifier sends a r_2 to the prover.
5. [Local computation] Prover computes $r = (r_1 + r_2 w)(mod q)$.
6. [Off chain] Prover sends r to the verifier.
7. [Local computation] Verifier checks whether $g^r = ah^{r_2}$

- **Completeness property:** Completeness property of the protocol holds with probability 1. As any honest prover will able to convince the verifier.
- **Special soundness:** If the verifier gets two valid responses corresponding, two different challenges, r_2 and r_2' the he can gain the knowledge of of the secret x. As from $r = (r_1 + r_2 w)(mod q)$ and $r' = (r_1 + r_2' w)(mod q)$, verifier can calculate $w = (r - r')(r_2 - r_2')^{-1}(mod q)$. Thus special soundness holds.
- **Construction of simulator:** By choosing r and r_2 at random, a simulated conversation $(g^r h^{-r_2}, r, r_2)$ can be made between honest verifier and prover.

Thus protocol is honest verifier zero knowledge.

Appendix B

System Configuration

```
>lscpu

Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   39 bits physical, 48 bits virtual
CPU(s):                          8
On-line CPU(s) list:             0-7
Thread(s) per core:              2
Core(s) per socket:              4
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           142
Model name:                      Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz
Stepping:                        10
CPU MHz:                         743.736
CPU max MHz:                     3400.0000
CPU min MHz:                     400.0000
BogoMIPS:                        3600.00
Virtualization:                  VT-x
L1d cache:                       128 KiB
L1i cache:                       128 KiB
L2 cache:                        1 MiB
L3 cache:                        6 MiB
NUMA node0 CPU(s):               0-7

>cat /proc/meminfo
MemTotal:        8064524 kB
```

Bibliography

[1] Andreas M Antonopoulos and Gavin Wood. *Mastering ethereum: building smart contracts and dapps*. O'reilly Media, 2018.

[2] J J Bambara and P R Allen. *Blockchain: A Practical Guide to Developing Business, Law, and Technology Solutions*. Mc Graw Hill Education (India) Private Limited, 2018.

[3] Imran Bashir. *Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained*. Packt Publishing Ltd, 2018.

[4] Mariya Benji and M Sindhu. "A study on the Corda and Ripple blockchain platforms". In: *Advances in Big Data and Cloud Computing*. Springer, 2019, pp. 179–187.

[5] *Blockchain workflow. Available at: http://gsmtraders.in/smac/blockchain/*.

[6] Jennifer L Breese, Sung-Jin Park, and Ganesh Vaidyanathan. "BLOCKCHAIN TECHNOLOGY ADOPTION IN SUPPLY CHANGE MANAGEMENT: TWO THEORETICAL PERSPECTIVES." In: *Issues in Information Systems* 20.2 (2019).

[7] Vitalik Buterin et al. "A next-generation smart contract and decentralized application platform". In: *white paper* 3.37 (2014).

[8] Ronald Cramer, Ivan Damgard, and Berry Schoenmakers. "Proofs of partial knowledge and simplified design of witness hiding protocols". In: *Annual International Cryptology Conference*. Springer. 1994, pp. 174–187.

[9] Álvaro García-Pérez, Alexey Gotsman, Yuri Meshman, and Ilya Sergey. "Paxos consensus, deconstructed and abstracted". In: *European Symposium on Programming*. Springer, Cham. 2018, pp. 912–939.

[10] Jens Groth. "Efficient maximal privacy in boardroom voting and anonymous broadcast". In: *International Conference on Financial Cryptography*. Springer. 2004, pp. 90–104.

[11] Aastha Gupta and Munish Gupta. "Electronic Mode of Payment–A Study of Indian Banking System". In: *International Journal of Enterprise Computing and Business Systems* 2.2 (2013), pp. 3–5.

[12] Feng Hao, Peter YA Ryan, and Piotr Zieliński. "Anonymous voting by two-round public discussion". In: *IET Information Security* 4.2 (2010), pp. 62–67.

[13] Heidi Howard. *ARC: analysis of Raft consensus*. Tech. rep. University of Cambridge, Computer Laboratory, 2014.

[14] Dalia Khader, Ben Smyth, Peter Ryan, and Feng Hao. "A fair and robust voting system by broadcast". In: *Lecture Notes in Informatics (LNI), Proceedings-Series of the Gesellschaft fur Informatik (GI)* (2012), pp. 285–299.

[15] Aggelos Kiayias and Moti Yung. "Self-tallying elections and perfect ballot secrecy". In: *International Workshop on Public Key Cryptography*. Springer. 2002, pp. 141–158.

[16] *Kovan Testnet. The fast and reliable Ethereum test chain. Available at: https://kovan-testnet.github.io/website/*.

[17] Andrea Kriskó. "Crypto currencies–currencies governed by belief Bitcoin, Piggycoin, Monero, Peercoin, Ethereum and the rest". In: *Conference book Konferenciakötet*, p. 283.

[18] Ranjit Kumaresan and Iddo Bentov. "How to use bitcoin to incentivize correct computations". In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 2014, pp. 30–41.

[19] PATRICK MCCORRY, MARYAM MEHRNEZHAD, EHSAN TOREINI, SIAMAK F SHAHANDASHTI, and FENG HAO. "On Secure E-Voting over Blockchain". In: (2021).

[20] Patrick McCorry, Siamak F Shahandashti, and Feng Hao. "A smart contract for boardroom voting with maximum voter privacy". In: *International Conference on Financial Cryptography and Data Security*. Springer. 2017, pp. 357–375.

[21] *Merkle Tree. Available at: https://changelly.com/blog/merkle-tree-explain/*.

[22] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. Tech. rep. Manubot, 2019.

[23] Jens David Ohlin. "Did Russian cyber interference in the 2016 election violate international law". In: *Tex. L. Rev.* 95 (2016), p. 1579.

[24] Kelly Olson, Mic Bowman, James Mitchell, Shawn Amundson, Dan Middleton, and Cian Montgomery. "Sawtooth: an introduction". In: *The Linux Foundation* (2018).

[25] Somnath Panja, Samiran Bag, Feng Hao, and Bimal Roy. "A smart contract system for decentralized Borda count voting". In: *IEEE Transactions on Engineering Management* 67.4 (2020), pp. 1323–1339.

[26] Deepa Ramachandra. *Are Blockchains mutable?* Ed. by medium.com. [Online; posted 30-Oct-2019]. Oct. 2019. URL: https://medium.com/swlh/are-blockchains-mutable-be65dbcd9f3.

[27] *Rinkeby Testnet. Available at: https://rinkeby.etherscan.io/*.

[28] Claus-Peter Schnorr. "Efficient signature generation by smart cards". In: *Journal of cryptology* 4.3 (1991), pp. 161–174.

[29] Adi Shamir. "How to share a secret". In: *Communications of the ACM* 22.11 (1979), pp. 612–613.

[30] wikipedia.org, ed. *Representative democracy*. URL: https://en.wikipedia.org/wiki/Representative_democracy.

[31] Scott Wolchok, Eric Wustrow, J Alex Halderman, Hari K Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati, and Rop Gonggrijp. "Security analysis of India's electronic voting machines". In: *Proceedings of the 17th ACM conference on Computer and communications security*. 2010, pp. 1–14.