

AN E-VOTING SYSTEM
BASED ON BLOCKCHAIN AND RING SIGNATURE

Yifan Wu
Student ID: 1700489
Msc Computer Science
Supervisor: David Galindo



Submitted in conformity with the requirements
for the degree of Master of Science
School of Computer Science
University of Birmingham

Declaration

The material contained within this thesis has not previously been submitted for a degree at the University of Birmingham. The research reported within this thesis has been conducted by the author unless indicated otherwise.

Signed

Abstract

**An E-voting System
based on Blockchain and Ring Signature**

Yifan Wu

Electronic voting (e-voting) is a symbol of modern democracy activities. Due to the high ballot privacy and verifiability, e-voting system has been booming in the recent years.

Particularly, Bitcoin, a digital currency system based on the cryptography, is highly open and transparent for the individual transaction. In other words, anyone can access to the transaction contents via blockchain. Besides, regarding to anonymous way it trades, the transaction of Bitcoin is untraceable.

On account of the pseudonymous of BitCoin address and the openness of the blockchain, which is consistent with part of e-voting requirement. This paper proposed an e-voting protocol based on blockchain by using the ring signature algorithm. The requirements can be satisfied with ballot-privacy, individual verifiability, eligibility, completeness, uniqueness, robustness, and coercion-resistance.

In order to prove the feasibility of protocol. This design implemented a fine web voting system software through PHP and JavaScript programming languages.

A security analysis, software performance analysis and evaluation are presented in the last section.

Acknowledgments

I would like to thank my supervisor Dr.David Galindo who gave me great help when doing the dissertation. David help me doing this challenging dissertation with his expert advice and his patient guidance. He listened to my thoughts and made suggestions and improvements to my protocol, and finally finished the dissertation. I would also like to thank my inspectors David Oswald and Per Kristian Lehre. They gave me great suggestions to write this paper.

I would like to express my gratitude to my best friend Rujia Li who gave me suggestions and encouragement to finish this dissertation and the software BlockVotes. I would also like to thank Ruoyu He, Xi Jin and Amy Li who help me to solve the grammar issues of the whole paper.

Finally, I would like to thank my parents who were always supporting me with their best wishes.

Contents

1	Introduction	6
1.1	Overview	7
1.2	E-voting	7
1.2.1	Related Work	7
1.2.2	Related Applications	9
1.3	Blockchain	10
1.3.1	Introduction	10
1.3.2	Properties	10
1.3.3	Mechanism	11
1.4	Structure of the thesis	12
2	Methodology	13
2.1	E-voting scheme Properties	14
2.2	Blockchain	15
2.2.1	Generating a Bitcoin address	15
2.2.2	OP_RETURN	16
2.3	Cryptography	18
2.3.1	RSA Algorithm	18
2.3.2	Ring Signature	18
3	Protocol	20
3.1	Definition	21
3.2	Protocol	21
3.2.1	Outline	21
3.2.2	Assumptions	22
3.2.3	Preparation Phase	22
3.2.4	First Registration Phase	22
3.2.5	Second Registration Phase	23
3.2.6	Publish Phase	24
3.2.7	Voting Phase	24

CONTENTS	5
3.2.8 Tallying Phase	24
3.2.9 Verification Phase	24
4 Implementation	25
4.1 Specification	26
4.2 Requirements	26
4.3 BlockVotes Models and Implantation	29
4.3.1 User Model	29
4.3.2 RA Model	31
4.3.3 EA Model	33
4.3.4 Voting Model	35
4.3.5 Verification Model	36
4.3.6 Tallying Model	37
5 Evaluation	40
5.1 Expected Properties	41
5.2 Performance	43
5.2.1 Ring Signature Performance	43
5.2.2 Tallying Performance	44
5.2.3 Confirmation Times	45
6 Conclusions	47
6.1 Summary	48
6.2 Conclusions	48
6.3 Future work	48
Appendices	53
A Instruction	53

Chapter 1

Introduction

1.1 Overview

Voting plays an important role in constructing a democratic society. The traditional voting requires voters to cast in appointed polling stations, which usually involves enormous expenditure on time and cost budget.

E-voting, a new substantial online voting system which is structured on cryptography technique, has been gradually implemented and emphasised by people. The system supports full-function online voting by general household devices, and the entire polling results will be counted automatically and anonymously. Compared with traditional voting, electronic voting is a more economic system addresses on transparency and impartiality.

As e-voting system mainly relies on the internet platform. The crucial challenge for e-voting is the significant security risks it might cause. In order to reduce risks, in the past 40 years, various protocols related to the ballot-privacy, individual verifiability, eligibility, completeness, fairness, uniqueness, robustness, universal verifiability and receipt-freeness have been widely proposed. Besides, the published protocols have implemented a variety of technologies, such as blind signature, ring signature, homomorphic encryption, Mix-Net, zero knowledge proof, etc [17]. In particular, the application of e-voting in digital currency has became gradually maturity nowadays.

Based on the common security requirements of participants, this paper proposed a blockchain-based protocol associated with the priorities of the ballot-privacy, verifiability, eligibility, completeness, uniqueness, robustness, and coercion-resistance.

A BlockVotes software has also been made to verify the feasibility of this protocol, by implementing a real-life online voting website, which allows participants to vote and view the results easily.

1.2 E-voting

1.2.1 Related Work

For a long time, many researchers are devoting to design a secure and efficient e-voting protocol. The first thesis related to cryptographic e-voting protocol was published by Chaum in 1981 and he used an anonymous commutation channel to encrypt the ballot [7]. With the developing of cryptography, a lot of protocols with its own properties had been proposed. In 1982, Richard A. DeMillo proposed a protocol requires all voters must participate and encrypt the ballot of each voter and at the end cast the ballots [12]. In 1985, Cohen and Fisher proposed a cryptographic protocol which can hold a secure ballot election. However, it requires the voting stage should at the same time [9]. The protocol encrypts the ballot by using homomorphism theorem and the government will release the tally result.

In 1992, Fujioka, Okamoto and Ohta proposed a practical secret e-voting scheme(FOO) used for the large scale elections, which can ensure the privacy of voters and the fairness of

voting. The scheme used the blind signature to blind the message the voter used to vote and send it to the administrator [14]. After this seminal paper has been released, a lot of e-voting software had been implemented and used for the market, such as the EVOX and SENSUS. In this scheme it also has its weakness, it requires all voters must vote and once someone abstains from voting, the result can tamper. The administrator cannot find out who tamper the result. In 1996, Juang and Lei proposed a blind signature based voting scheme but requires everyone should attend the voting event.

After 3 years, M. Ohkubo, F.Miura and M.Abe promoted the FOO scheme by using a threshold encryption protocol and the Mix-Net communication channel which can keep the privacy of the voter. For the voters, they can need not to participate the tallying part of the event and they can walk away after voting [25].

With the development of e-voting system, there are a series of criminal behaviors related to the e-voting, such as electoral fraud, threat a voter or vote buying. To deal with these problems, many new requirements or properties of e-voting scheme have been proposed likes the receipt-freeness and coercion-resistance.

The receipt-freeness means that the voter cannot prove the vote result to anyone after voting. In 1994, Benaloh discussed the terminology named receipt-freeness firstly[2]. Although Benaloh uses homomorphic encryption to make the implementation of receipt-freeness, Martin Hirt argued that it will not be valid if there are more than one tally authority. In 1995, V.Niemi and A.Renvall raised a scheme by adopting the receipt so that the voter cannot prove who he or she votes for[24]. At the same time, K.Sako and J.Kilian proposed the first Mix-Net based protocol satisfied with receipt-freeness [30]. This protocol is under the assumption that there is no private channel between the voting station and the voters. After one year, Okamoto uses a non-anonymous channel, a private channel and the bulletin board to propose a voting scheme satisfied receipt-freeness[26]. Unfortunately, this protocol has been proved does not satisfy with receipt-freeness. In 2000, M.Hirt and K.Sako use homomorphic ElGamal encryption technique to design a private channel protocol with receipt-freeness, but this protocol does not suit for the large-scale election[15]. In 2001, O.Baudron proposed a new scheme to satisfy this property by using Paillier cryptosystem and the zero-knowledge proof[1].

In recent years, a lot of researchers focus on the receipt-freeness and coercion-resistance of e-voting. In 2010, Juels introduced the new direction of e-voting which named coercion-resistance and proposed a scheme[18]. In 2012, O.Spycher and R.Koenig promote his scheme by adding the random integer f . The proposed scheme will get the encrypted integer C . The tally authority can judge if there is any fake ballot through decrypting C to the random integer f [32]. The coercion-resistance can be satisfied further.

By the developing of the decentralized digital currency, some researchers argued a way to vote and tally on the blockchain. In 2015, Czepluch discussed the application domains of the blockchain and argued that blockchain can use for the e-voting [11] . At the same time, Z.Zhao and THH.Chan proposed a way to vote using Bitcoin and zk-SNARKs with

the properties named privacy, verifiability and irrevocability [34]. In 2016, a protocol using Zerocoins has been proposed and ensures most properties of the e-voting[33]. However, using a Zerocoins is hard to make the implementation to the software. At the same year, C.Jason, Paul and K.Yuichi proposed a protocol using blind-signature and Bitcoin cards [16]. However, it cannot protect the privacy in some situation, for instance, if the administrator knows the Bitcoin address of the voter, the administrator can know who the voter is by linking the address and message on the blockchain.

1.2.2 Related Applications

The research of e-voting system is widely used nowadays. Some partial practices are listed as follows.

In 2000, e-voting has been used in US Election[8]. Although it is an experiment in some area of Florida, it was a milestone in the development of e-voting.

In 2002, United Kingdom tried out an electronic voting system. 16 public authorities were awarded to build the e-voting system. After 1 year, more than 18 authorities were awarded[22].

In 2004, US election used an electronic voting system DRE for the first time[5]. India uses this system for parliamentary elections on a national scale.

In 2007, France UMP party made a history of internet-based voting. More than 31,000 voters vote in UMP to in 2007 French Presidential election[13]. This was the first mass E-voting activity in history.

In 2009, China used electronic voting for the election of the grass-roots organization in Hangzhou. There were 3122 residents enrolled this voting activity with an electronic touch screen[21].

In 2014, the election of Ministry of National Education(France) received 1,760,000 ballots[21]. It took the lead in legal and security network voting, thus popularized the channels of network voting.

1.3 Blockchain

1.3.1 Introduction

In 2008, the founder of Bitcoin S.Nakamoto published a paper [23] to specify a cryptocurrency system based on the peer-to-peer network. The Bitcoin has changed the traditional way of the cash payment system. With the development of the Bitcoin, Blockchain technology has aroused the attention of people. The blockchain is a public ledger, all individuals can synchronize the latest ledger into local and they have no permission to tamper the content of the public ledger.

To distinct various blockchain, there are two categorizations of the blockchain[27]. One is classified by the requirement of the network nodes to the verification process.

- **Permissionless blockchain:** No central service or authority is required to compute during the verification process. Usually, this computational process happens in the device of anyone.
- **Permissioned blockchain :** There is a central network used for confirming the verification nodes.

Another one is classified by the publicity of the blockchain.

- **Public blockchain:** Anyone in the world can read, download, broadcast the transaction of the blockchain.
- **Private blockchain:** The blockchain only belongs to the individual, government or an organization which is not public.

In recent years, the Bitcoin and Ethereum are ever-increasing popular. They both are the permissionless and public blockchain.

For the Bitcoin, it has 2 sub network, the Bitcoin network and the testnet. The testnet is the testing environment of the Bitcoin network. In this network, the coin does not has any value. It is free to use and get the test coin form the faucet[19].

Ethereum is a digital currency similar to Bitcoin. It is also a complete set of decentralized application platform. While using Ethereum for digital currency trading, anyone can publish and use decentralized applications on Ethereum. Ethereum's advantage is that it provides a complete toolchain for decentralized application development, deployment. By using smart contract, it makes block-chain-based application development extremely convenient.

1.3.2 Properties

Since the birth of the blockchain, the blockchain has the properties of decentralization, decentralized trust, common maintenance, data reliability, privacy protection. It has been unprecedented attention and its development is very rapid.

- **Decentralization:** The blockchain is decentralized. There is no central computing devices to store the ledger of transactions. Every node of blockchain store the same copy.
- **Hard to forge:** Due to its decentralization, every block should be distributed to every node around the world.
- **Transaction traceable:** Each transaction in the blockchain is open and transparent. Every transaction details includes the sender address and the receiver address, which anyone can trace a transaction.

In this paper, we proposed a protocol based on Bitcoin. For each transaction, everyone can download the information from the blockchain. In the Bitcoin, every Bitcoin address has no relation to its personal identity. Therefore the blockchain is pseudonymous for anyone and has the transparent transactions, which has the same requirements for the e-voting properties.

1.3.3 Mechanism

The blockchain consists of a set of nodes based on a peer-to-peer network. For each node, it maintains the consistency of the data by performing a consensus algorithm. To specify the mechanism of the blockchain, the Bitcoin is a typical representation of the blockchain. To specify the blockchain, we should have a basic concept of the block. The block is composed of the block header and the main part of the block including a serialized transaction raw. The transaction raw contains the unique identifier(TxID) which is the hash value of the transaction. The identification value of all transactions on each block constitutes each leaf node of the Merkle tree.

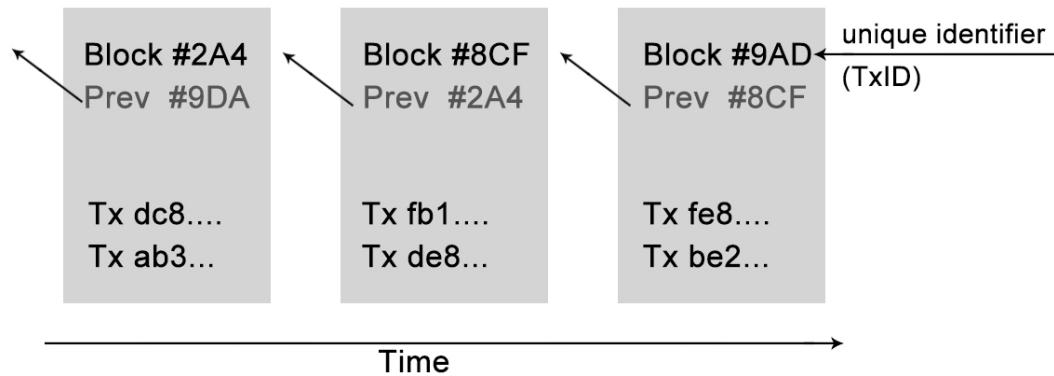


Figure 1.1: Ring signature

By storing the previous block TxID into the next block, all nodes are linked with the block header which is also called blockchain. When creating a new block, the blockchain

will use consensus algorithm to create a new transaction unique identifier. A new block is generated by the consensus algorithm, which generates a new block by calculating the hash value of the block header. After most nodes accept the new block, it will be added to the blockchain.

1.4 Structure of the thesis

This thesis contains 6 chapters.

Chapter 1 is the introduction of the problem and the literature review of its related technologies.

Chapter 2 is the methodology of the problem. This chapter detailed the properties and requirements of the e-voting scheme and other technique or algorithm used in the protocol.

Chapter 3 is the most important part of this thesis. The proposed protocol indicates the whole design scheme of the implementation.

Chapter 4 is the implementation of the protocol. By creating an application named BlockVotes, any properties can be tested. This chapter is written in the methodology of the software engineering.

Chapter 5 is the evaluation of the protocol, which evaluated its properties and argued the reason.

Chapter 6 is the conclusion of the proposed protocol and the implementation. This part discussed the future work.

Chapter 2

Methodology

2.1 E-voting scheme Properties

In recent 30 years, more and more e-voting protocols has been published. The major properties of e-voting scheme can be described from the papers wrote by Cranor[10], Cetinkaya[6] and Fujioka[14].

Basic Properties

Ballot privacy: Anyone cannot know whom the voter voted for. The ballot is hidden from outside observers.

Individual verifiability: The voter can verify his ballot is counted correctly after he voted.

Eligibility: Only the legal voters can enroll the voting event.

Accuracy/Completeness: Every votes should be counted correctly.

Fairness: Nothing can influence the result of voting. If the system leaks the voting result or the authority adds a voter during the voting, the event can be defined as unfair.

Uniqueness: Every voter can only vote once. The voter will have no permission to vote more if he votes.

Robustness: Anyone cannot influence or modify the final voting result when tallying.

Advanced Properties

Universal Verifiability: Anyone can verify the eligibility of each ballot and the impartiality of the result.

Receipt-freeness: The voter cannot receive or try to build any receipt after he voted to prove how he vote.

Coercion-Resistance: There is no coercer can cooperate with the voter. The voter cannot prove who he voted.

2.2 Blockchain

2.2.1 Generating a Bitcoin address

To broadcast a message on the blockchain, the participants need a Bitcoin address. By using SHA256, RIPEMD160 Hashing and Base58 Encoding, the Bitcoin address can be generated as Fig 2.1 [20].

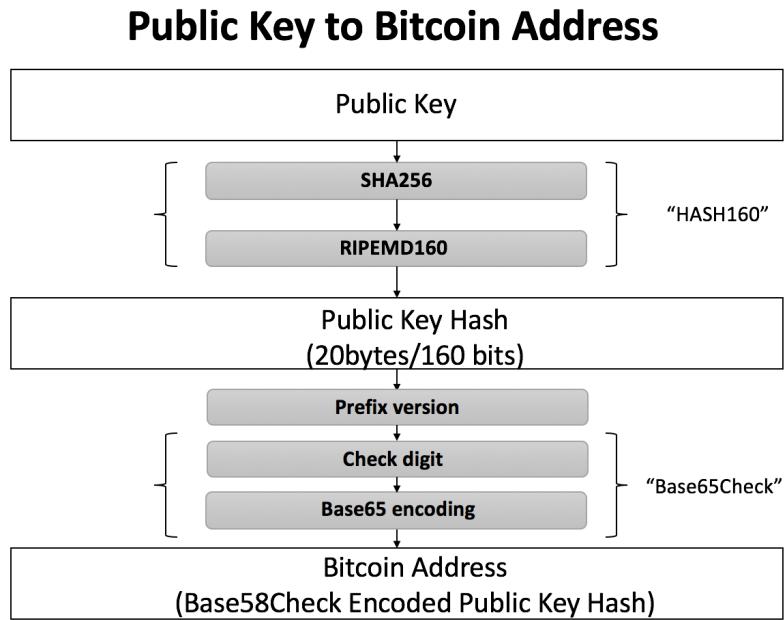


Figure 2.1: Public Key to Bitcoin Address[20]

1. Generate the private key by Elliptic Curve Digital Signature Algorithm, generally named *secp256k1*. The size of the Bitcoin private key is 256 bits.
2. Generate the Bitcoin public key from the Bitcoin private key (x,y) with the DER format.
3. Hash the Bitcoin public key as $PK_{hash160}$ into hash160 by producing the SHA256 and RIPEMD160 algorithm.
4. Add the prefix of the version at the head of $PK_{hash160}$ according Table 2.1. Define the intermediate hash value of public key $fingerprint = prefix + PK_{hash160}$, which is also named the fingerprint.
5. Define $Sha256(Sha256(fingerprint))$ as the check digit d . Adding the d at the end of *fingerprint*.
6. Generate the final Bitcoin address by encoding $fingerprint + d$ with the Base65 encoding algorithm. Define $address = Base65(fingerprint+d)$ as the final Bitcoin address.

Blockchain	prefix
BITCOIN	'00'
TESTNET	'08'

Table 2.1: The prefix of the version

2.2.2 OP_RETURN

To discuss the OP_RETURN of the blockchain. we should consider the transaction first. For each transaction on Bitcoin, it contains the input script and the output script as the Fig 2.2. The Figure is shown the transaction between Alice and Bob who are the transaction participants.

Here is an example of a transaction between Alice and Bob with the reference of "haha" on the blockchain of testnet.

Define the Bitcoin address of Alice as mxLqfJvTTEojWVZVTanEcXs1kXaBkdoqfX, and the Bitcoin address of Bob as n4Kc1AwFos3aZRvD3Tc9imzeMeA8E9DEUr.

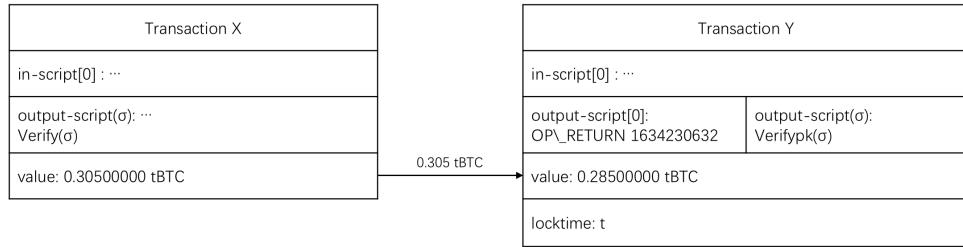


Figure 2.2: Transaction between Alice and Bob

Alice creates Transaction Y with the input of one transaction. In order to confirm Transaction Y in the blockchain, the input-script[0] should refer to the Transaction X. The output script of Transaction Y contains 2 parts. One is the signature for Transaction Y which is signed by the private key of Alice SK . Another is the OP_RETURN code which is the reference of "haha". OP_RETURN is a stack-based script without loops. As it is defined in the protocol of Bitcoin, it can store up to 80 bytes in the transaction. The locktime t means the Transaction Y should not be placed before time t [34].

To confirm this transaction before t , the transaction creator Alice should pay the mining fees to the miner. The miner can use the PoW(Proof of Work) algorithm to find a block including $Verifypk(\sigma)$. Finally, the Bob will receive the money with the reference of "haha".

Further, to specify this example, this example has been made in the network of testnet. The details can be listed in the Table.

From the Table 2.2, we can see that output-script[0] indicates the OP_RETURN which is 1634230632. To decode the OP_RETURN, we can use *hex2bin()* algorithm to convert it to the characters.

Transaction ID	025e4fd916832c4028b1bcc446c8a41c10798fbea49793ce245ccf700e621d4f
input[0]	mprsHi9HKpn9bH14ZZV7YC7mxdrJ6wws7r(0.12249999)
input-script[0]	3045022100b7a6e8aa5cd553e4c21ad68e851c140c3e87d3eefc4d0a 3045022100ae906a357c927d170f19710aca1de3c5ebcbe2c60fe9 626b24ed876d7f23fad40220354d0b0c254679817deac98f4fcfa 33be48eaf74c77a2e0b4db2046747cb2b3d0102ce592b293c66 88ca587dea59780acca8da8215d4d3261db338e9ea39fc46ae19
input-value[0]	0.30500000
output[0]	OP_RETURN 68616861
output-script[0]	OP_RETURN 1634230632
output-value[0]	0.00000000
output[1]	n4Kc1AwFos3aZRvD3Tc9imzeMeA8E9DEUr
output-script[1]	OP_DUP OP_HASH160 fa25611aeed75a33a9fc8cc83d2039059c37d837 OP_EQUALVERIFY OP_CHECKSIG
output-value[1]	0.28500000
tx_hex	0100000001cfbd41e842b9f6752b8a76e4803ded991bdbf0c1ec193e5add bfc8467c1b6d1c010000006b483045022100ae906a357c927d170f19710a ca1de3c5ebcbe2c60fe9626b24ed876d7f23fad40220354d0b0c25467981 7deac98f4fcfa33be48eaf74c77a2e0b4db2046747cb2b3d012102ce592b 293c6688ca587dea59780acca8da8215d4d3261db338e9ea39fc46ae19ff fffff020000000000000000066a046861686120e0b201000000001976a9 14fa25611aeed75a33a9fc8cc83d2039059c37d83788ac00000000

Table 2.2: An example of Bitcoin testnet transaction

In this example, by decoding the OP_RETURN 1634230632, we can get the message "haha". The OP_RETURN can store the messages. In this implementation, we use it to store the ring signatures and candidate id.

2.3 Cryptography

2.3.1 RSA Algorithm

RSA algorithm is a kind of asymmetric the cryptographic algorithm which used to encrypt and decrypt the messages[29]. Its security is based on the difficulty of large integer decomposition. There are many implementations in reality. The specific algorithm can be described as follows.

1. Choose two different large prime numbers.
2. Define $n = pq$, $\varphi(n) = (p - 1)(q - 1)$.
3. Choose $e \in [0, \varphi(n) - 1]$.
4. Calculate the modular multiplicative inverse of $\varphi(n)$ as d which ensures $ed = 1 \bmod \varphi(n)$.
5. Define e, n as public key and p, q, d as private key
6. Encryption: Give the message x , compute $y = x^d \bmod n$ to encrypt the message by using the public key (e, n) .
7. Decryption: Give the ciphertext y , compute $x = y^d \bmod n$ to encrypt the message by using the private key (p, q, d) .

2.3.2 Ring Signature

In 2001, Rivest, Shamir and Tauman proposed a question that how to leak a secret[28]. To answer this question, they told a story about a member of cabinet informs against Prime Minister. Bob is a member of the cabinet who wants to leak a message about the illegal actives about the Prime Minister to the journalist. To ensure his safety, Bob must inform him an anonymous channel then the journalist can easily verify his identity of the cabinet. To solve this problem, Bob cannot use a group signature scheme to send the message because he cannot confirm if the group administrator is controlled by the Prime Minister.

They proposed a new scheme called ring signature, and each member of the cabinet is the ring member and everyone is equal and anonymous.

The ring signature scheme can be described as follows. Supposing the scheme has a number of n members. For each user u_i , he has his own public key y_i and his private key x_i and they sit down in a ring as the Figure 2.3.

The scheme can be divided into 3 parts: Generating a key pair, generating a ring signature and verifying the signature.

Generating a key pair: A key pair generator algorithm for the signer through computing the symmetric key k_i . The algorithm can compute each public key y_i and private key x_i from the k_i .

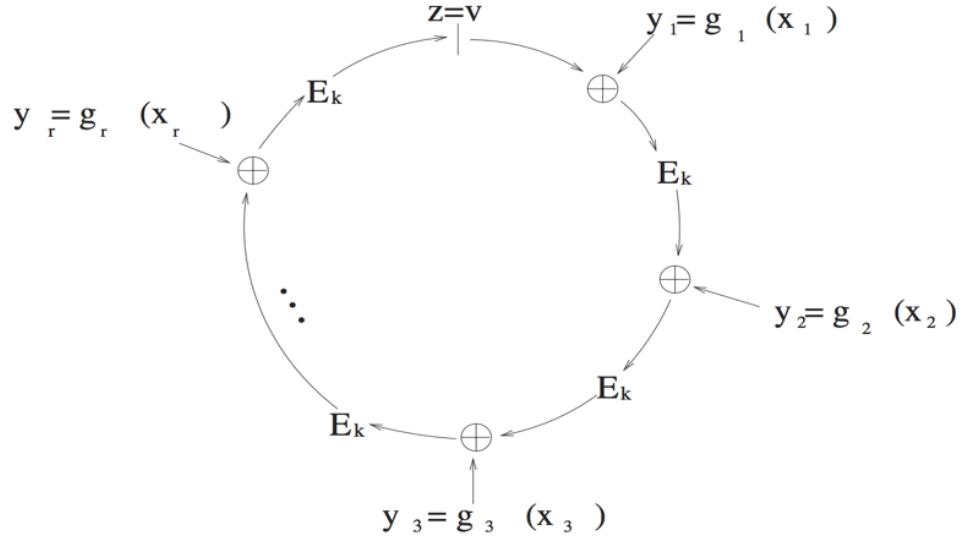


Figure 2.3: Ring signature

Generating a ring signature: By inputting the message m , numbers of n and its public keys list $L = y_1, y_2, y_3, \dots, y_n$ and the private key x_i of the signer, the algorithm can output a signature which is called the ring signature as σ .

Verifying the signature: By inputting the message m and the ring signature σ . If σ is the signature of m , output true and else output false.

For the ring signature, the security properties and advantages can be separated as anonymity and unforgeability [28].

1. Unconditional anonymity. Even if the attacker steals all private keys of the voters, the probability of confirming the identity of voter will be less than $1/n$, which n is numbers of all members in the ring.
2. Unforgeability. Even if the outside attacker tampers a ring signature in accordance with the message m without any private key of the voters, the coincidence probability can be overlooked.
3. Compared to the group signature scheme, there is no administrator in the group for the ring signature scheme. Every member is equal and the scheme does not need any trusted third party.

Chapter 3

Protocol

3.1 Definition

The proposed protocol consists of three entities: Voters (V_i), RA (Registration Authority), EA (Election Authority) and Bitcoin Address Pool.

Voters (V_i): The voters should be a set of list. For each voter to vote can be defined as V_i .

Candidate(C_i): The candidates should be a set of list. For each candidates to vote can be defined as C_i .

Registration Authority(RA): The voters should sign up as a register in the current e-voting system at first. The voter should save their public keys(PK_i) and Bitcoin address(A_i) into this system and the system transfer it to the database. For the RA, it provides the candidate(C_i) to the voters.

Election Authority(EA): The election authority is responsible for tallying the votes. The EA has its own Bitcoin address(A_E). When the voting has been finished, the EA should start counting the votes and transfer the result to the voting system.

Bitcoin Address Pool: The Bitcoin address pool is a list of all Bitcoin addresses generated from the EA system randomly by using ECC algorithm. The private key SK_{Ai} of each address will store into the EA system.

Public supervision: To build this protocol, some of the content should be public and be supervised under anyone as the open-audit part. Anyone can check its completeness and validity. All public keys of the voter PK , the EA's Bitcoin address A_E and the sets of $(\sigma, sha256(\sigma))$ should be public through the inner API of the system without any permission.

3.2 Protocol

3.2.1 Outline

The whole protocol development consists of seven sequential phases, each stage is executed by different actors.

Preparation Phase: To begin with, the Election Authority (EA) should set up a new voting project first, and then store BitCoin address as its private key into EA system.

First Registration Phase: The voting registration stations run by Registration Authority (RA) are located nearby residential areas. Voters and election candidates are eligible to vote after authentication of passport or other essential IDs. A random register code will be sent to participants as a link by RA through email.

Second Registration Phase: Once participants click the random register code link sent by RA, they can generate their public key and private key by using key pair tools or local RSA tool. The new generated public key should be stored in the system, while private key can be held privately.

Publish Phase: On the voting cut-off date, every single public key from voters will be collected under supervision. As long as the start voting button has been clicked, RA will

not accept new registration requests anymore.

Voting Phase: When voters use their private key to sign favourable candidate, they will get a unique ring signature which will be broadcasted to the blockchain. The protocol does not require every voter to take part in during voting phase.

Tallying Phase: Tallying stage is strictly under public supervision, people can access to the tally page to view or cast ballots.

Verification Phase: In order to monitor the validity of the voting result, EA transaction history are open for public on blockchain.

3.2.2 Assumptions

The protocol is under the assumptions to promote the properties of privacy and verifiability.

1. Registration Authority and Election Authority will not correspond.
2. The hashing algorithm $sha256()$ is secure.
3. Every actor will follow the phases to enroll the voting event.

Now we will detail the phases we talk about earlier.

3.2.3 Preparation Phase

The details of this phase can be described in order as follows.

1. The EA saves his own private key of the Bitcoin(SK_b) into the system.
2. The system will generate EA's Bitcoin address (A_{EA}) from his private key of Bitcoin(SK_b).
3. The EA creates a new voting item with the voting id(L_i), title, limitation of the voting numbers(n) and the description of this voting item.
4. The EA system will generate the numbers of the n bitcoin addresses($A_1, A_2 \dots A_n$) as the Bitcoin Address Pool automatically.

3.2.4 First Registration Phase

The details of this phase can be described in order as follows.

1. The candidate(C_i) takes his passport and authenticate to the RA in person.
2. The RA verifies the identity of the candidate and ask his name, his personal description and save it into RA system.
3. The RA will generate and give him his candidate id(C_i).
4. The voter(V_i) takes his passport and authenticate to the RA in person.
5. The RA verifies the identity of the voter and asks the email address of the voter then sends him an email with an random registration code link as LK_i to avoid multiple registrations.
6. The LK_i is generated randomly and has no relationship with the name of voter and his email address.

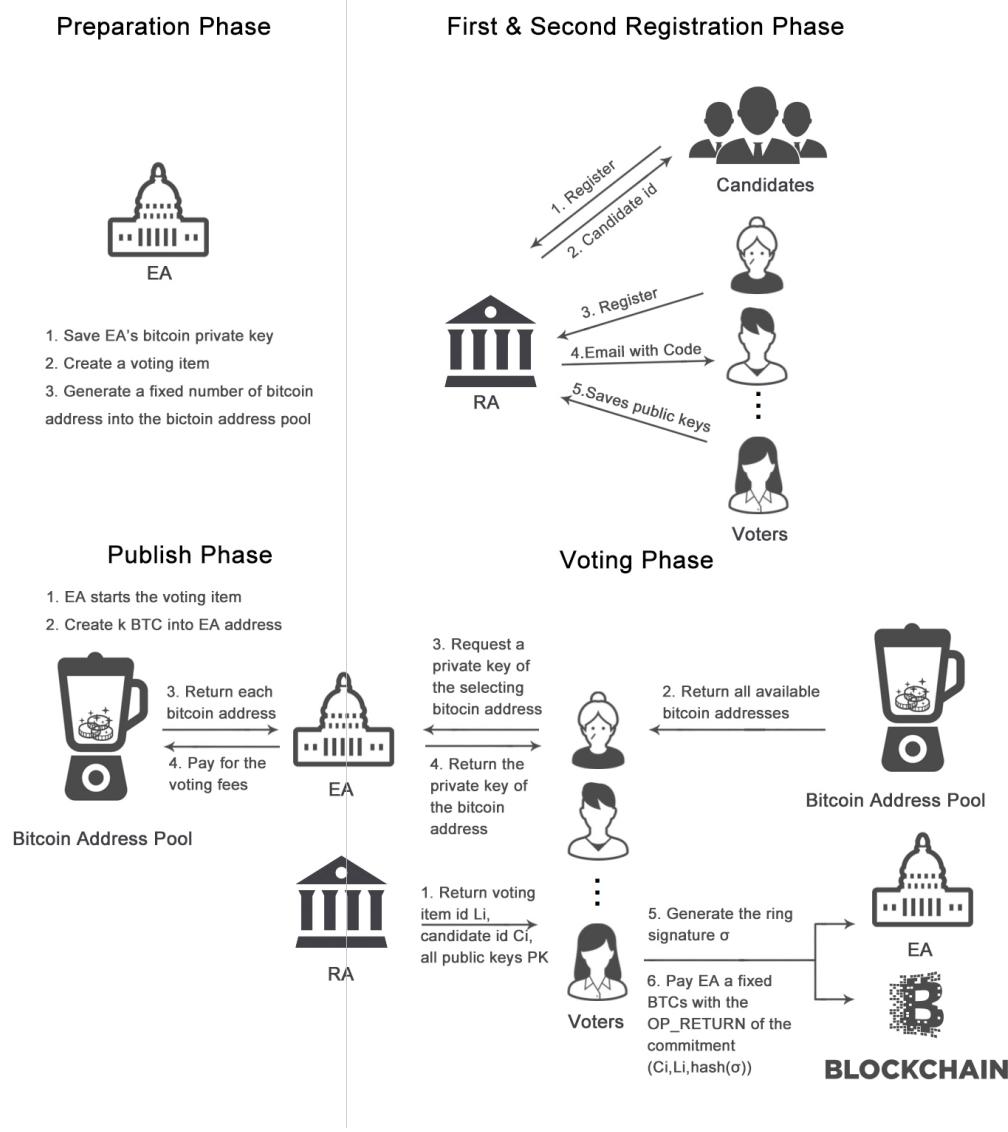


Figure 3.1: Proposed Protocol(Mainly Phases)

3.2.5 Second Registration Phase

The details of this phase can be described in order as follows.

1. The voter opens the registration links LK_i .
2. The voter V_i generates his key pair (SK_i, PK_i) .
3. The voter V_i saves his public key PK_i into the system.
4. At the end of the registration, the set of voters should be fixed as a number of n .

3.2.6 Publish Phase

The details of this phase can be described in order as follows.

1. On the voting cut-off date, the EA decides to start the voting which means the ring of public keys has been confirmed and the RA should not accept any registration requests.
2. EA creates k BTC in his own Bitcoin account.
3. EA pays a fixed amount of bitcoin k/n as the voting fees to each A_i , such as 0.0001 BTC. (Once the voter voted, the voting fees would send back to EA)

3.2.7 Voting Phase

The details of this phase can be described in order as follows.

1. The voter chooses the candidate C_i he vote for and the current voting id L_i .
2. The RA returns the public keys set $(PK_1, PK_2, PK_3 \dots PK_n)$ to the voter.
3. The voter uses his private key SK_i and all public keys PK to sign the signature of the candidate C_i as $\sigma(C_i, SK_i, (PK_1, PK_2, PK_3 \dots PK_n))$. The system saves the set of $(\sigma, sha256(\sigma))$ at the same time.
4. The voter selects a Bitcoin address A_i to publish from the Bitcoin Address Pool and EA returns the private key SK_{A_i} of the address to the voter.
5. The voter V_i pays all balance of A_i the to EA address A_{EA} with an OP_RETURN of the commitment($sha256(\sigma(C_i, SK_i, (PK_1, PK_2, PK_3 \dots PK_n))), C_i, L_i$).

3.2.8 Tallying Phase

The details of this phase can be described in order as follows.

1. The system returns all sets of $(\sigma, sha256(\sigma))$ and all public keys PK automatically.
2. The system fetches all transactions in EA Bitcoin address A_{EA} automatically.
3. The system fetches the OP_RETURN form each transaction and verify the signature σ validity.
4. The system counts each valid transaction and add 1 to the candidate C_i .
5. If the voter V_i is absent, mark it as the abstain from voting.
6. If the Bitcoin transaction history has more than twice transactions from the same A_i , count the first and ignore others.

3.2.9 Verification Phase

The details of this phase can be described in order as follows.

1. The system returns all public keys $(PK_1, PK_2, PK_3 \dots PK_n)$ automatically.
2. For each voter V_i , he can use a set of all public keys $(PK_1, PK_2, PK_3 \dots PK_n)$, the ring signature σ , the candidate C_i to verify his vote.
3. The voter V_i can use the transaction id to fetch the commitment from the blockchain to verify if the signature is published in the right way.

Chapter 4

Implementation

4.1 Specification

The e-voting system has been designed using the above protocol named BlockVotes. The system will have the properties of ballot privacy, individual verifiability, eligibility and etc.

When developing this project, I choose iterative and incremental development as the software development process model. To make an implementation, the system should consider the basic models and the properties of the e-voting.

In general, BlockVotes allows voters and candidates to register in the system. To save the money of mining, we recommend the EA to use the testnet network to save the mining fee. The voter can vote in the system and anyone can verify the result of voting. The administration can be broken into 2 authorities, the election authority and registration authority. The BlockVotes system should have the models as follows.

1. User Model is the logic of logging in.
2. RA Model is a model for Registration Authority including its APIs.
3. EA Model is a model for Election Authority including its APIs.
4. Voting Model is a model for the voter to register and vote.
5. Verification Model is a model for anyone to verify the ring signature.
5. Tallying Model is the vital part of the voting which allows anyone can count the result of the voting from blockchain in real time.

4.2 Requirements

To build such a system, the functional and nonfunctional requirements should be established as follows.

Functional Requirements

REQ 1.1 The page must be broken down into the login page, EA page, RA page, public page.

REQ 1.2 The server must show different pages for the different user role.

REQ 1.3 The server must have privilege control for the different user role.

REQ 1.4 The user roles should divide into 4 actors: voter, Registration Authority, Election Authority and public.

Public Pages

REQ 2.1 The server must have a public API to make some content public such as public keys of the voter PK , the Bitcoin address A_{EA} of Election Authority and the sets of $(\sigma, sha256(\sigma))$ without any permission.

REQ 2.2 The verification page must be public and under supervision according to the protocol.

REQ 2.3 The tallying page must be public and under supervision according to the protocol.

REQ 2.4 The voting page must be public.

REQ 2.5 The voting page must be able to use the ring signature algorithm to sign a message.

REQ 2.5 The voting page must be able to make a transaction with the blockchain account address of EA to broadcast a message on the blockchain.

REQ 2.5 The voting page must be able to make a transaction with the blockchain account address of EA to broadcast a message on the blockchain.

REQ 2.6 The verification page could fetch the OP_RETURN code from transaction id the voter input.

REQ 2.7 The verification page could decode the commitment of OP_RETURN code to the candidate and signature.

REQ 2.8 The verification page must be able to fetch the signatures σ from the $sha256(\sigma)$ fetched from the blockchain OP_RETURN code via public API.

REQ 2.9 The verification page must be able to verify the validity of the ring signature σ .

REQ 2.10 The verification page must be able to verify the integrity of all public keys.

REQ 2.11 The verification page should be able to verify the correctness between signatures and the hash value of the signatures $(\sigma, sha256(\sigma))$.

REQ 2.12 The verification page could choose a voting item and then redirect to its corresponding verification page.

REQ 2.13 The tallying page must be able to fetch all transactions of EA blockchain account.

REQ 2.14 The tallying page must be able to fetch the signatures σ from the $sha256(\sigma)$ fetched from the blockchain OP_RETURN code via public API.

REQ 2.14 The tallying page could decode the commitment of OP_RETURN code to the candidate and signature.

REQ 2.15 The tallying page should be able to verify the correctness between signatures and the hash value of the signatures $(\sigma, sha256(\sigma))$.

REQ 2.16 The tallying page could choose a voting item and then redirect to its corresponding tally page.

REQ 2.17 The tallying page must be able to verify the validity of the ring signature σ .

REQ 2.17 The tallying page must be able to tally and show the voting result in real time.
Registration Page

REQ 3.1 The registration page must verify the validity of the link code, if it is invalid, show the forbidden page.

REQ 3.2 The registration page could provide a key pairs generator by using RSA and running all in the front-end.

REQ 3.3 The registration page should store the public key of each voter into Registration Authority system.

REQ 3.4 The registration page could edit the public key if the voting has not been started.

REQ 3.5 The registration page could show the voting date if the voter filled the public key.

REQ 3.6 The registration page could show the description of this voting item.

Registration Authority Page (RA Page)

REQ 4.1 The RA system must set up an STMP account for sending an email to the voters.

REQ 4.2 The RA system must have a database to store all link codes.

REQ 4.3 The RA system must be able to add a candidate with his name and the description.

REQ 4.4 The RA system must be able to edit or delete a candidate.

REQ 4.5 The RA system must be able to generate a link code by using a random algorithm.

REQ 4.6 The RA system must provide an API for the voter to fetch the id, names, and descriptions of all candidates.

Election Authority Page (EA Page)

REQ 5.1 The EA system should be able to create a voting item with its title, maximum voting numbers n and description.

REQ 5.1 The EA system should be able to edit or delete a voting item.

REQ 5.2 The EA system could be able to view all candidates.

REQ 5.3 The EA system must be able to store the blockchain account private key of EA.

REQ 5.4 The EA system must be able to convert the blockchain account private key of EA into the bitcoin address of EA.

REQ 5.5 The EA system could be able to set different blockchain networks, such as the Bitcoin(BTC) and Bitcoin testnet (TESTNET).

REQ 5.6 The EA system must provide an API for the voter to fetch the private keys from the Bitcoin Address Pool.

REQ 5.7 The EA system must be able to create a number of n Bitcoin address and its private keys into the Bitcoin Address Pool randomly.

REQ 5.8 The EA system must be able to make the transactions on the blockchain network between the addresses of Bitcoin Address Pool and its own Bitcoin address A_{EA} .

REQ 5.9 The EA system must be able to fetch all transactions of EA blockchain account.

Non-Functional Requirements

REQ 6.1 The backend should be developed in PHP using MVC framework.

REQ 6.2 The front-end should be developed in Javascript.

REQ 6.3 The response time of voting page must be less than 300ms.

REQ 6.4 The response time of public API must be less than 300ms.

REQ 6.5 The response time of voting page must be less than 300ms.

REQ 6.6 The system could use third party API to make the transaction on the blockchain.

REQ 6.7 The tallying system should be stable enough to show the correct result.

REQ 6.8 The voting system should be stable enough to broadcast the commitment to the blockchain.

REQ 6.9 The system should use a relational database to store the information of the voting item, Bitcoin addresses candidates and voter.

REQ 6.10 The system should be tested with the different browser and OS system.

REQ 6.10 The public page should have a friendly user experience.

4.3 BlockVotes Models and Implantation

The BlockVotes has been developed by the PHP programming language. To make the logic clearly, the MVC framework named Slim has been used to route the page. The Laravel/Illuminate used to Data Access Object(DAO) and the TWIG used to connect the front-end page with the back-end. To manage the dependencies, the Composer has been used as the dependency management tool. The system is developed with the database MYSQL and the testnet of the Bitcoin. The Git is used to version control of the whole project.

To build this system, many third party libraries have been chosen. The repository from Github named ring-signature to generate the signature. BitcoinJS is used to generate the Tx_hex. BitcoinECDSA has been used to generate the blockchain account address.

To broadcast a Tx_hex on the blockcahin, it is unnecessary to set up a blockchain client locally. There are plenty of broadcast APIs. For this implantation, SoChain and smartbit have been chosen to fetch the transactions from the blockchain or broadcast a Tx_hex.

4.3.1 User Model

According to the protocol, the user can be broken down into voter, candidate, RA, EA and public supervision.

When EA or RA want to login to the system, he can use his username and password to enter to the system. The system will check the password if it is correct. And if it is correct, they will enter into different pages according its user role. If the password does not match, the system will let them know.

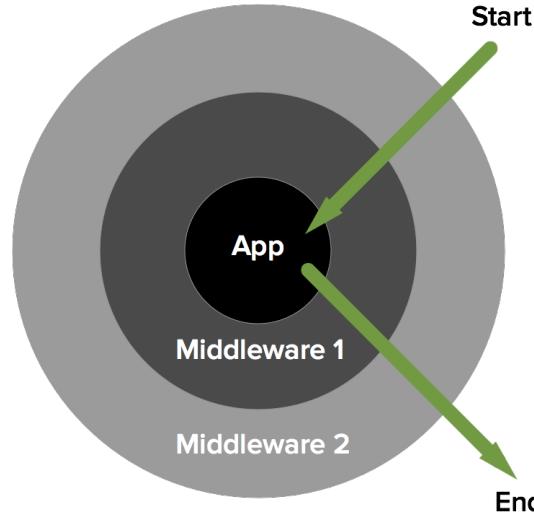


Figure 4.1: The middleware of Slim framework[31]

By using the middleware of the framework, the privilege control can be easily implemented. The middleware allows developer to inject a specific action after loading the frame-

work but before running app controller as the Figure 4.1.

Here is a code sample of privilege control for EA. To judge the user role, the system should fetch the user role from the container. If the user role is not the 2(the role id of EA user), the system will not allow the user to view.

```

1 public function __invoke($request, $response, $next){
2     if (!$this->container->auth->check()){
3         $this->container->flash->addMessage('error', 'Please sign in before
4             doing that');
5         return $response->withRedirect($this->container->router->pathFor(
6             'auth.signin'));
7     } else if ($this->container->auth->user()->role != 2) {
8         $this->container->flash->addMessage('error', 'This page is only
9             show to the Election Authority');
10    return $response->withRedirect($this->container->router->pathFor(
11        'home'));
12 }
13 $response = $next($request, $response);
14 return $response;
15 }
```

In the BlockVotes system, the user role can be described as follows.

id	username	nickname	password	role
1	ra	Registration Authority	...	1
2	ea	Election Authority	...	2

Table 4.1: The user role table

Here is the login user interface design for EA and RA. For the voters or the public, they do not need to login and type the URL to enter the page.

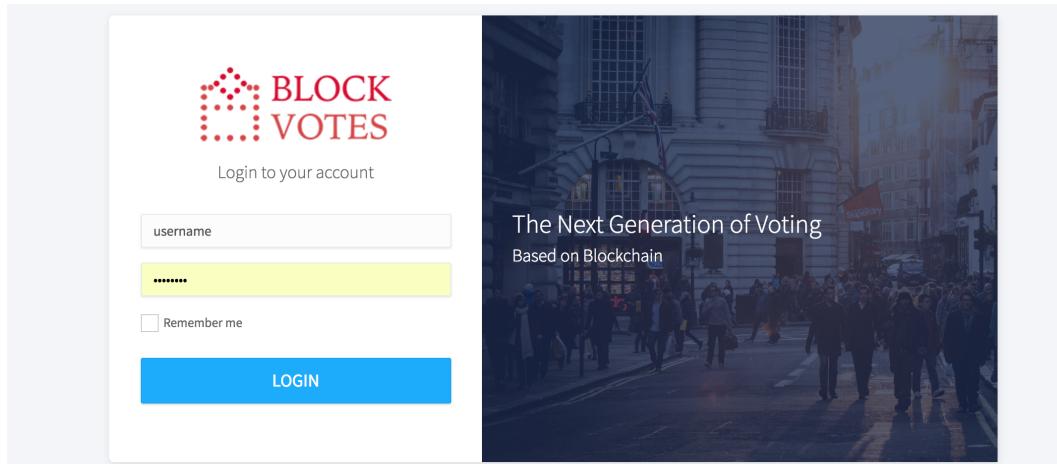


Figure 4.2: Login interface

4.3.2 RA Model

For the Registration Authority, they signs in the system and serves the voters and candidates. Thus, the system must satisfy these basic features.

- Add a voter with his email address and send him an email.
- Add a candidate with his name, description and avatar image.
- Edit or delete a candidate.

When sending an email, the framework will load the `env` configure properties file first including the STMP server properties. And then the system use the third party dependency named Swift Mailer to make the connection with a STMP server and post the email to the target email address. Some partial codes can be written as bellow.

```

1 $transport = (new \Swift_SmtpTransport( getenv('STMP.SERVER'), getenv(
2     'STMP.PORT'), 'ssl'))
3         ->setUsername(getenv('STMP.USERNAME'))
4         ->setPassword(getenv('STMP.PASSWORD'));
5 $mailer = new \Swift_Mailer($transport);
6 $message =(new \Swift_Message('Start your vote now'))
7         ->setFrom(array(getenv('STMP.USERNAME') => 'BlockVotes Team'))
8         ->setTo(array($email => 'Voter'))
9         ->setBody($content)
10        ->setContentType("text/html");

```

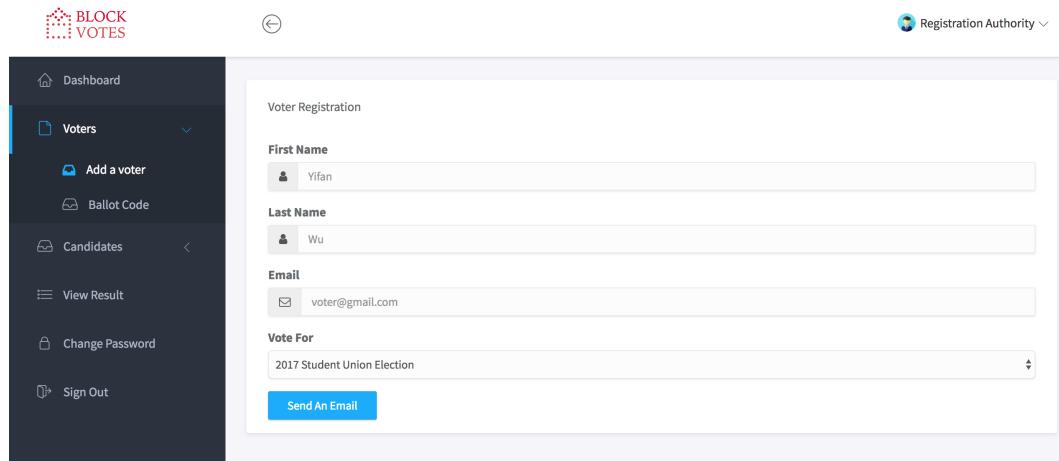


Figure 4.3: Add a voter interface

To generate a code link, the code must be random and have no relationship with the voter identity. The technique behind the code link is the use of random algorithm.

```

1 public function makeCard() {
2     $randomdict = "0123456789ABCDEFGHIJKLMNPQRSTUVWXYZ";

```

```

3 $results = "";
4 for ($i = 0; $i < 16; $i++)
5     $results .= $randomdict[mt_rand(0, strlen($chars)-1)];
6 return $results;
7 }

```

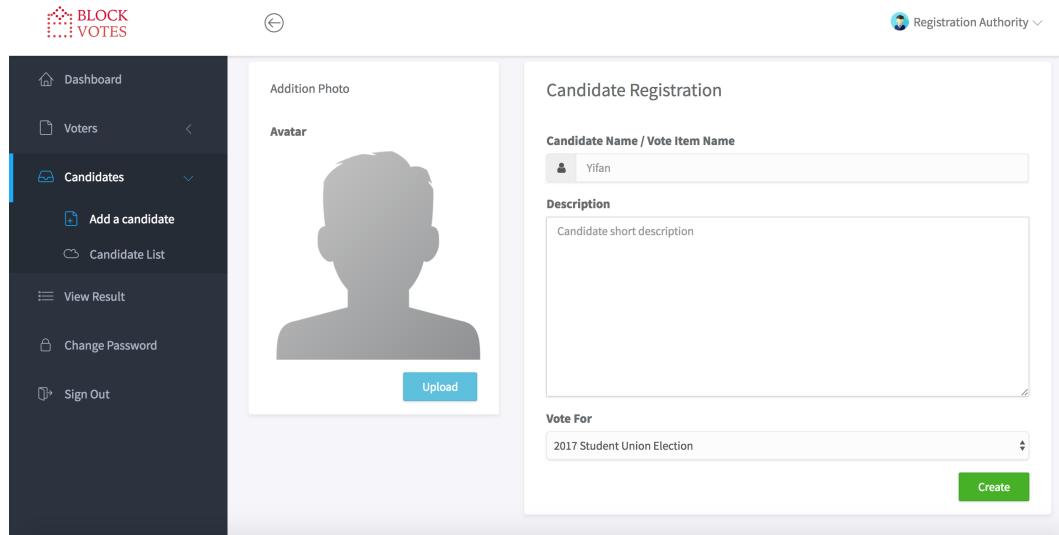


Figure 4.4: Add a candidate interface

Candidate Name	Candidate ID	Bio	Action
Tanno Tom	#1	I can lead you to the future.	Edit Delete
Yifan Wu	#2	Please vote me!	Edit Delete
Taylor Hayward	#3	I love you guys	Edit Delete
Rhys Scott	#4	Vote me! Vote me!	Edit Delete
Andrew Phillips	#5	No reasons	Edit Delete
Fernando Arcuri	#6	For the future of student life	Edit Delete

Figure 4.5: RA candidates management interface

4.3.3 EA Model

For the Election Authority, the system must satisfy these basic features.

- The dashboard to show the balance of the EA blockchain account, the blockchain network and the voting information.
- Check the balance of the blockchain account generated from the Bitcoin Address Pool.
- Pay the voting fees to all or single blockchain account address.
- Create a voting event with the title, description and the image.
- Edit or delete a voting event.
- Start or stop a voting event.
- View the candidate of a voting event.
- The setting page to set up the private key of the EA blockchain.

When EA creates a voting item, the system will use the dependency BitcoinECDSA to create the bitcoin addresses with the inputted number of n .

ID	Title	Voters Numbers	Created Time	Operation
1	2017 Student Union Election	10	2017-08-05 16:51:06	<button>Stop Voting</button> <button>Profile</button> <button>Candidate</button>
2	Business Meeting Proposal	5	2017-08-08 19:23:55	<button>Start Voting</button> <button>Profile</button> <button>Candidate</button>
3	A test voting	2	2017-08-15 14:31:59	<button>Stop Voting</button> <button>Profile</button> <button>Candidate</button>
4	Test	10	2017-08-16 05:00:58	<button>Stop Voting</button> <button>Profile</button> <button>Candidate</button>
5	A test voting for demo	2	2017-08-21 03:25:08	<button>Stop Voting</button> <button>Profile</button> <button>Candidate</button>

Figure 4.6: Voting List Management interface

EA can check the balance of the blockchain account address pool (Bitcoin Address Pool) he generated before.

To make a transaction between two Bitcoin addresses, the system uses a third party Javascript library named BitcoinJS to create transaction serialized as the hexadecimal value. By using the third party API to broadcast the hexadecimal value on the blockchain, that is to make the real transaction.

ID	Bitcoin Address	Balance
5	mikAVj3qeVAgQSHBTYHWywNGbe1cmFMvn	1.3000
6	mxLqfJvTTEojWVZVTanEcXs1kXaBkdoqfx	0.0000
7	mn34yVxPJcnMvZ7MwGgxweNd1xt6pviYcr	0.0000
8	mwA2Jgb77QzHng1vifq8Bdp2znME6WL	0.0000
9	moEti8SJE4jFt1BthZDf55zpphQrjbxun	0.0000
10	moNhkpefZ4tDFUyszPpgChm9VEKuBvhBr	0.0000

Figure 4.7: Check balance of the Bitcoin Address Pool interface

Here is a code sample about how to make a transaction from *sourceAddress* to *targetAddress* on the bitcoin testnet.

```

1  function makeTransaction(itemid,targetAddress,pbhash){
2      if(!lock){
3          $.getJSON("https://chain.so/api/v2/get_tx_unspent/BTCTEST/" +
4              sourceAddress, function(result){
5                  lock =true;
6                  var last = result.data.txs.length - 1;
7                  var unspent_txid = result.data.txs[last].txid;
8                  var unspent_vout = result.data.txs[last].output_no;
9                  txb = new Bitcoin.TransactionBuilder(network);
10                 txb.addInput(unspent_txid, unspent_vout);
11                 value = Number(result.data.txs[last].value * 100000000);
12                 pay = 0.0001 * 100000000; // mining fees
13                 change = parseInt(value - pay);
14                 var commit = new Buffered(pbhash);
15                 var dataScript = Bitcoin.script.nullData.output.encode(commit);
16                 txb.addOutput(dataScript, 0);
17                 txb.addOutput(targetAddress,change);
18                 txb.sign(0, keyPair);
19                 var txRaw = txb.build();
20                 var txHex = txRaw.toHex();
21                 postdata = { tx_hex : txHex };
22                 postTransaction(itemid,postdata);
23             });
24             return true;
25         }
26     }

```

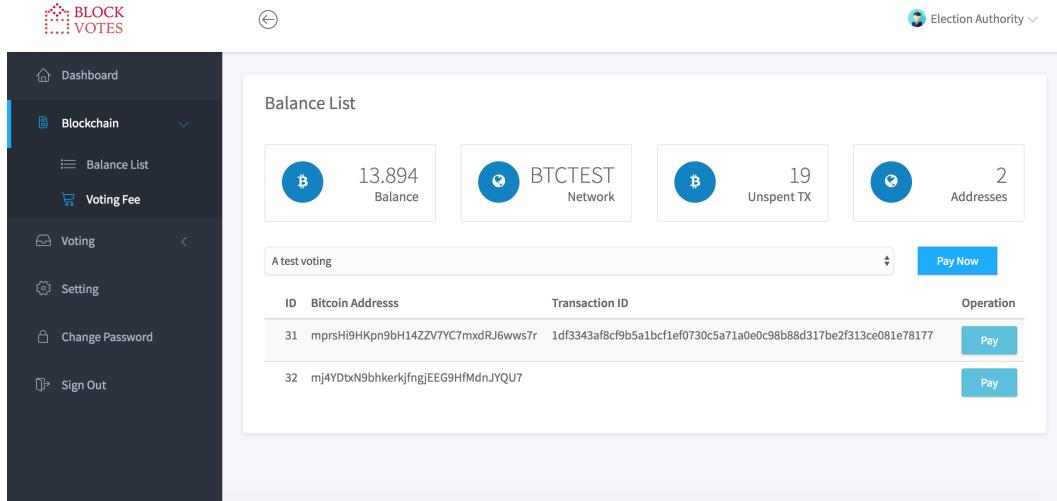


Figure 4.8: EA voting fee payment interface

4.3.4 Voting Model

When the voter has signed up in Registration Authority, he has to open the code link from his email. Once he open the link, the system will do the logic judgement to redirect him to the different page. The logic diagram can be described as the Figure 4.2.

The voter should paste his public key and save it to the system. If in the voting day, the voter should open the link and the system will let him to vote. By inputting the private key and selecting the candidates he vote for, the voting page will broadcast his vote on the blockchain automatically.

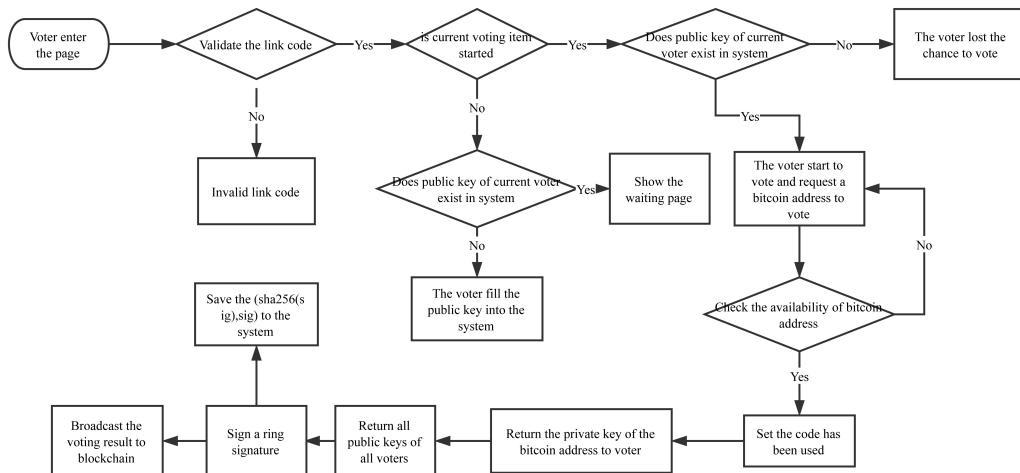


Figure 4.9: Voting Logic Workflow

To create a ring signature, the system uses the third party Javascript library named ring-signature. This library includes the methods about creating a ring signature and verifying a ring signature with the message. Some crucial codes can be written as follows.

```

1 keys.push(new JSEncryptRSAKey(value.public_key)); // push a publickey
2 var z = Math.floor(Math.random() * (keys.length+1));
3 keys.splice(z, 0, pubkey);
4 init(keys);
5 var sig = sign(candidate, z); // got the ring signature
6 console.log(keys_match(sig, keys)); // verify the ring signature

```

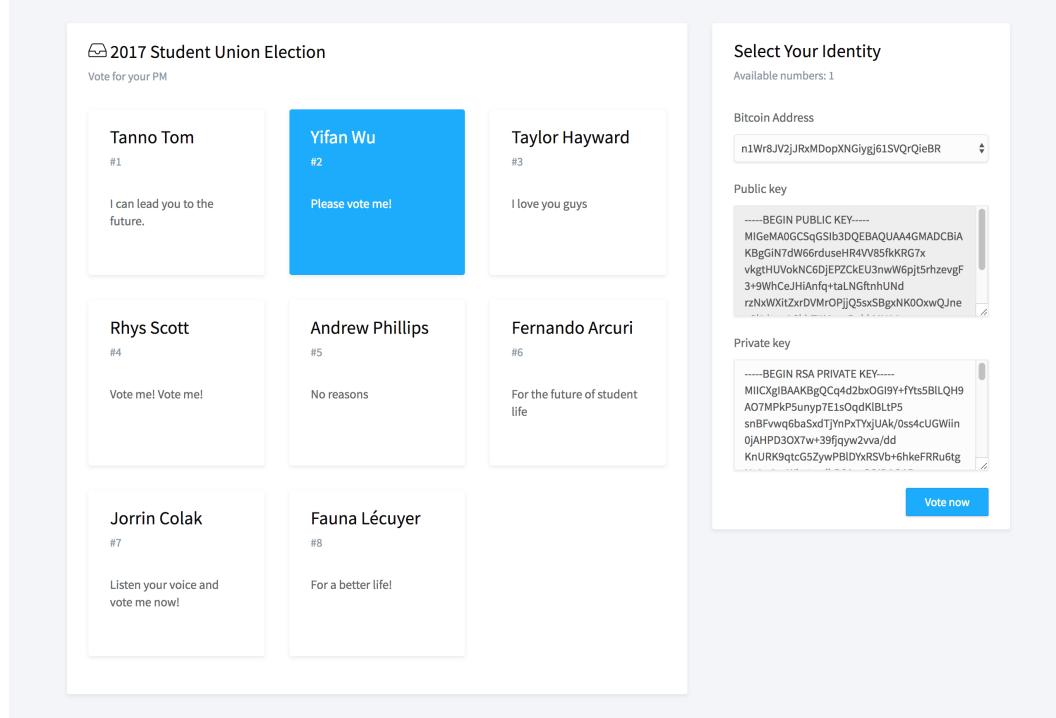


Figure 4.10: RA candidates management interface

4.3.5 Verification Model

When the voter has voted, the voter will get the ring signature or the blockchain id as the receipt. To verify the vote is counted correctly, the voter should go to the verifying page. The logic diagram can be shown as Figure 4.11 .

The system provide 2 ways to verify the votes result.

1. Transaction ID: It can automatically fetch the candidate id, ring signature hash from the blockchain. And then it can fetch the ring signature from the public API according to its hash.

2. Ring signature: The voter or anyone can type the ring signature and choose the candidate id manually. This process will have no connection with any server.

And then the verifying page should use the Javascript `verfiy()` function to verify if the candidate match with the ring signature as the Fig 4.12. Further, he can check if the public keys match with the API fetched from the server as Fig4.13.

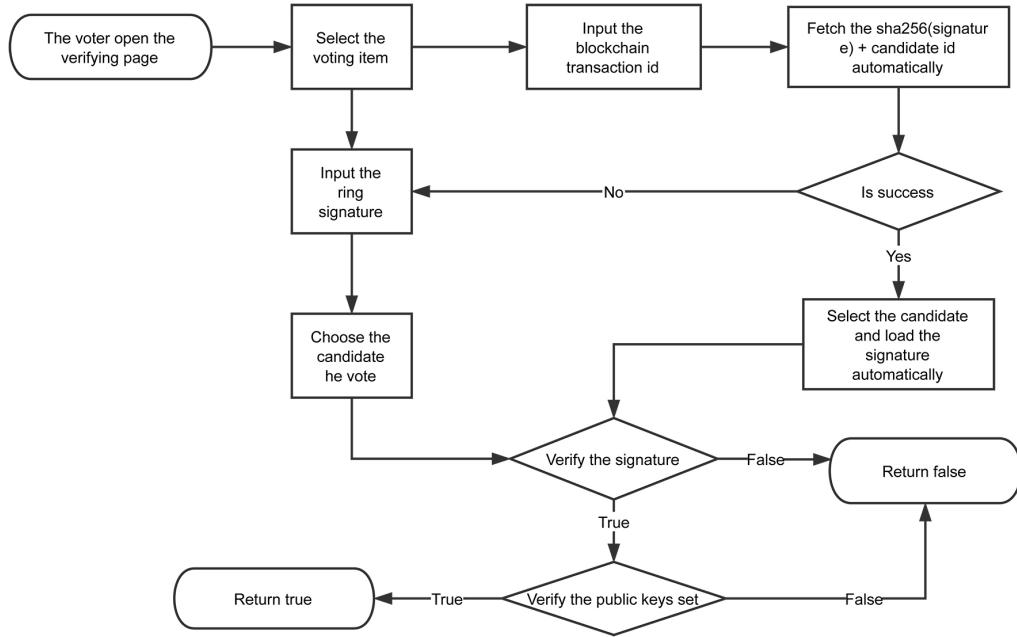


Figure 4.11: The work flow of the verification Page

4.3.6 Tallying Model

The tallying phase is the most crucial part of the voting. This phase must happens in the front-end to ensure anyone can tally in the real time. The detail of this phase can be described as follows.

1. Fetch all public keys and store it locally.
2. Fetch all transaction of the blockchain account of EA.
3. Decode the commitment of the OP_RETURN code into candidate id and hash value of ring signature.
4. Verify the validity of each ring signature.
5. Count the valid ballot.

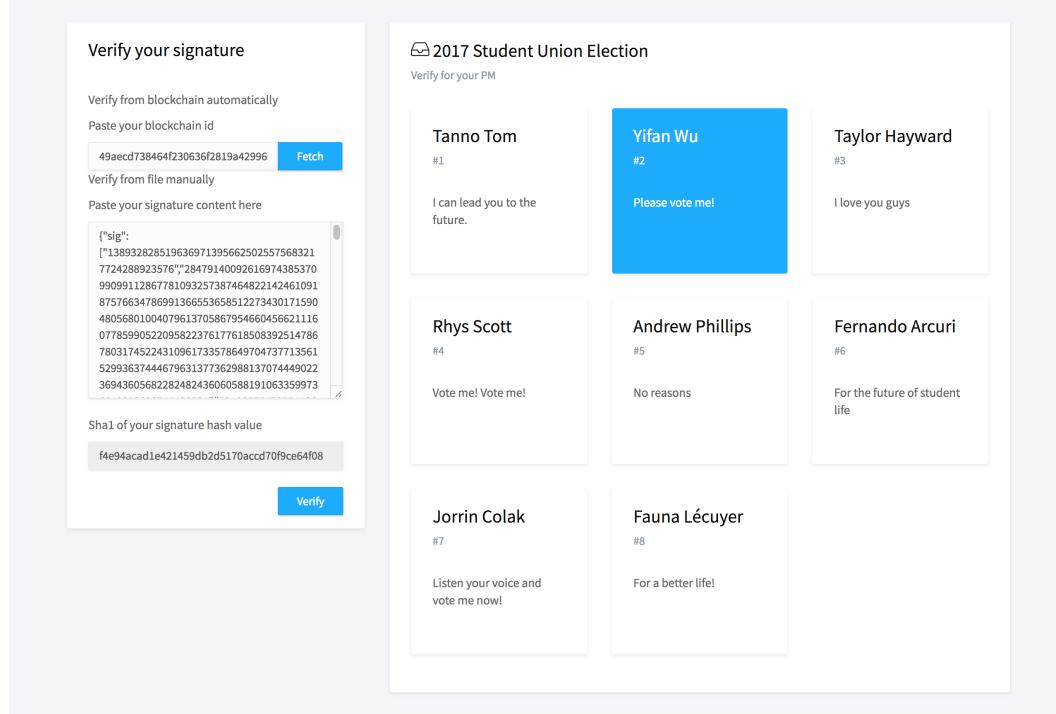


Figure 4.12: RA candidates management interface

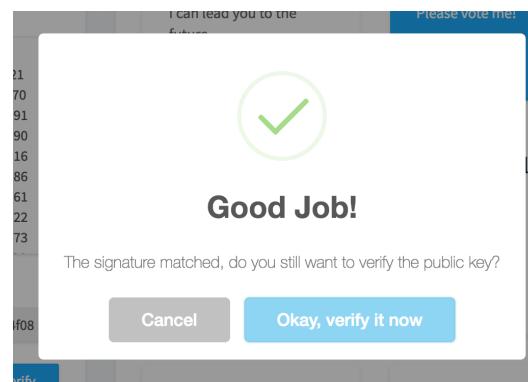


Figure 4.13: RA candidates management interface

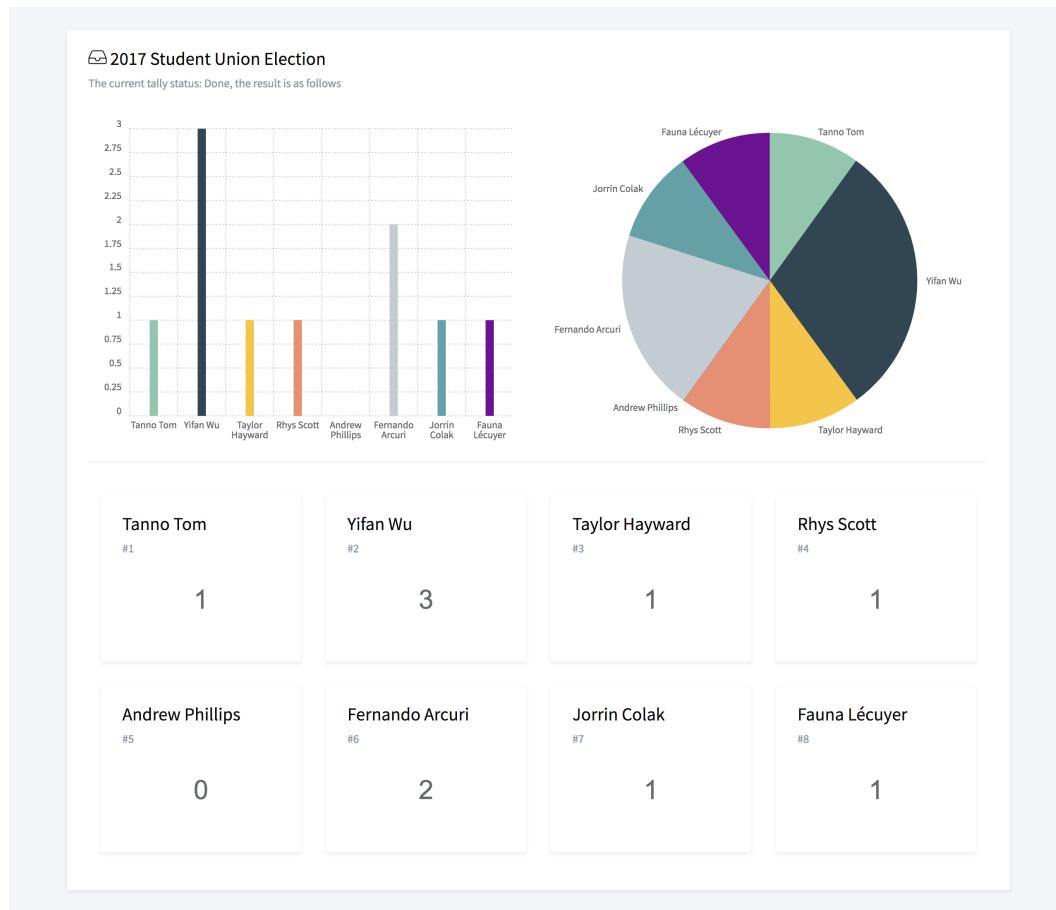


Figure 4.14: RA candidates management interface

Chapter 5

Evaluation

5.1 Expected Properties

According to the properties described above, the properties of the voting scheme are sufficient. For this protocol and implementation, the properties can be listed as below.

Ballot privacy: *Anyone cannot know whom the voter voted for.*

The blockchain account address is random and no outside observer or even the system can not know the relationship between the voters and the Bitcoin address. For the ring signature, it has its property of anonymity. No one can conjecture the real identity of the voter from the ring signature.

Individual verifiability: *The voter can verify his ballot is counted correctly after he voted.* The voter can use the $(\sigma, sha256(\sigma))$ to verify the $sha256(\sigma)$ is publish to the blockchain correctly, where σ is the signature. The voter can use his ring signature σ to verify he votes for the right candidate.

Eligibility: *Only the legal voter can enroll the voting event.*

In this system, the voter should register in RA and get the code link if he verified to be legal to vote. When starting to vote, only the voter who has the code link can enroll the voting event.

Completeness: *Every votes should be counted correctly.*

All votes can be counted correctly. By using the ring signature, the system will provide all public keys. The EA blockchain account will receive an amount of bitcoins when the voting is finished. The tallying system can easily count the OP_RETURN of each transaction. Anyone who does not vote but saves his public key PK_i to the system will be seen as the abstention.

Uniqueness: *Every voter can only vote once. The voter will have no permission to vote more if he votes.*

The protocol has a method to ignore extra votes from the same voter. In the tallying phase, if the Bitcoin transaction history has more than twice transactions from the same A_i , count the first and ignore others.

Robustness: *Anyone can not influence or modify the final voting result when tallying.*

The result has been broadcasted to the blockchain if the voter votes. The blockchain is hard to forge and modify.

Coercion-Resistance: *There is no coercer can cooperate with the voter. The voter can not prove who he voted.*

The protocol can ensure this property only happens when the number of voters n is big enough. If someone threatens a voter to vote him, the voter can tell him a transaction id and the ring signature which votes for the menace from the public API. For the menace, he can not confirm the signature voter give him is belong to the voter.

The tallying system fetches all transactions of the blockchain account address of EA. When tallying the ballots, the system verifies the validity of the ring signature and only count the first and legal vote.

Not all the properties are satisfied with this protocol and implementation. Here are the properties does not satisfy with the implementation.

Fairness: *Nothing can influence the result of voting.*

The tallying system is in the real time. The system can not guarantee this property.

Receipt-freeness: *The voter can not receive or try to build any receipt after he voted to prove how he vote.*

When the voter starts to vote, he will get the ring signature σ , $sha256(\sigma)$ and his transaction id of the blockchain. This is the receipt for the voter to verify his ballot.

5.2 Performance

In order to make sure the test results are convincing, we had done sufficient tests on Testnet (Bitcoin test environment) to evaluate the algorithm and software.

5.2.1 Ring Signature Performance

For the proposed protocol, if the number of the voter members n are small enough, ring signature will be effectively sign and verify. On the contrary, the public keys size, ring signature size and the efficient of signing and verifying will increase gradually.

Unit test ran on Chrome 62 web browser of a laptop with 2.9 GHz Intel Core i5. The results have been illustrated on Table 5.1. We found that the errors of unit test depends on the performance of various CPU.

Public keys number	Signing time(ms)	Verifying time(ms)	Signature size(byte)
1	69.88598633	2.386962891	355
10	109.7609863	10.9831543	3127
50	135.7580566	28.3449707	15441
100	232.9682617	54.00390625	30842
200	414.2949219	107.8620605	61622
300	607.6271973	145.9350586	92413
400	813.138916	190.0830078	123202
500	976.5891113	248.5419922	153982
600	1160.855957	292.5158691	184754
700	1325.011963	344.0258789	215558
800	1500.853027	375.4931641	246351
900	1721.103271	437.5969238	277141
1000	1866.314209	474.9248047	307911
3000	6189.906006	1392.203125	923712
5000	12598.85913	2501.8479	1539489

Table 5.1: The performance of the ring signature

According to the Figure 5.1 and Figure 5.2, there is a linear relationship between public keys number(voter number) and other parameters such as the time of singing or verifying, the signature size. The more voters enroll in, the less the efficient is.

We recommend that the voter number should less than 3000 so that every voter will not wait for more than 1 minute to get the ring signature. And the size of signature file will less than 93KB in that case.

The proposed protocol does not suit for large election activity if the voter want to get a much better user experience.

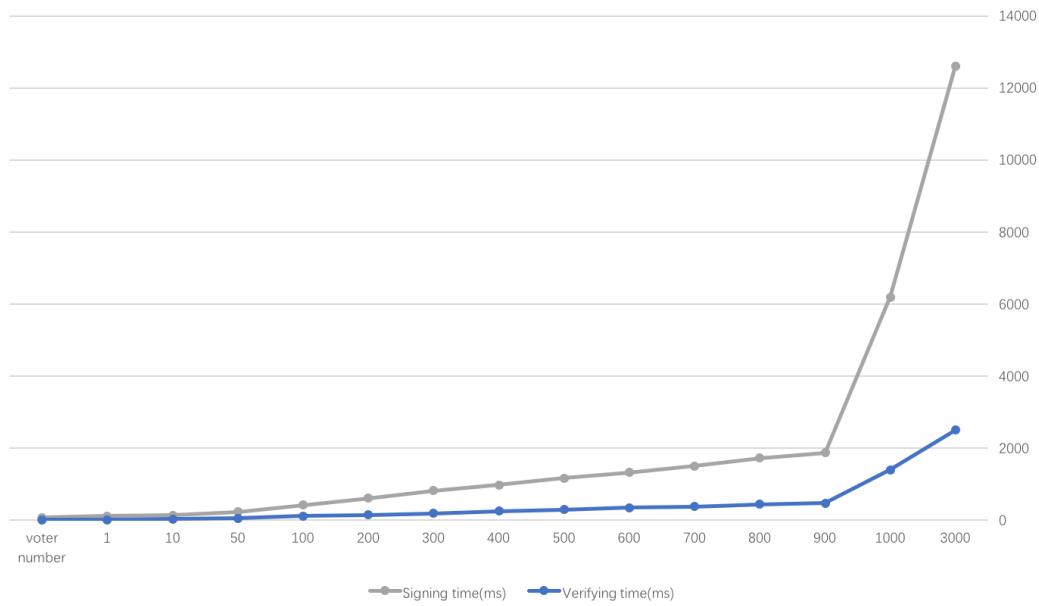


Figure 5.1: The relationship between voter numbers and the time of signing and verifying

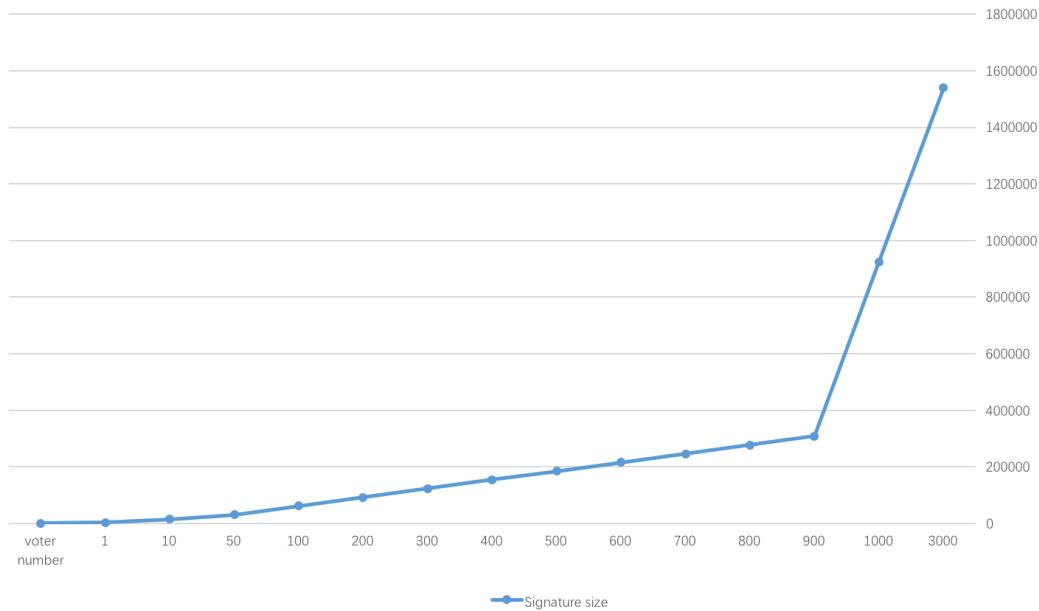


Figure 5.2: The relationship between voter numbers and the signature key size

5.2.2 Tallying Performance

The tallying phase is the most crucial part of the voting. The performance of this phase can be evaluated.

When doing the evaluation, there are 3 factors that affect the whole efficiency of tallying phase according to the Figure 5.3.

- **Blockchain API:** It is responsible for fetching all transactions of EA according to the proposed protocol. It depends on the voters network and the numbers of the transactons. The tested response time is 868ms in the testing environment(Figure 5.3).
- **Public API of getting candidates:** To get all candidates, the browser should send a request to the server. The time can be estimated to 163 ms in the testing environment(Figure 5.3).
- **Public API of getting all public keys:** To get all public keys and verify the ring-signature, the browser should send a request to the server. The time can be estimated to 195ms in the testing environment(Figure 5.3).
- **Public API of matching the ring-signature through the hash value:** When the client got all transactions, it will send a request to the server. The average time can be estimated to 55ms in the testing environment(Figure 5.3).
- **Verifying time:** The time of verifying the ring signature can be evaluated as the Section 5.2.1 discussed.

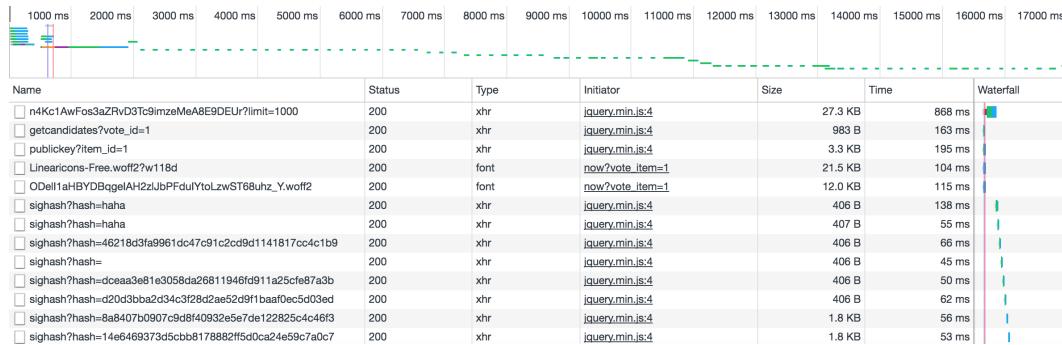


Figure 5.3: The Chrome developer console of the tallying page

Overall, if the transaction of EA is big enough, the whole process will increase significantly. To avoid this, the EA should replace his blockchain address frequently once the voting item has been stopped.

5.2.3 Confirmation Times

The confirmation time is the symbol of broadcasting the voting result all over world in this implementation. According to the protocol of the Bitcoin, Once a transaction with the

OP_RETURN including the candidate id and the hash value of ring signature broadcast to the blockchain network, the miners should find it and confirm it.

The miners should use a consensus algorithm to confirm the transaction. We estimated that the time interval is 10 minutes and the result satisfied a poisson process[3].

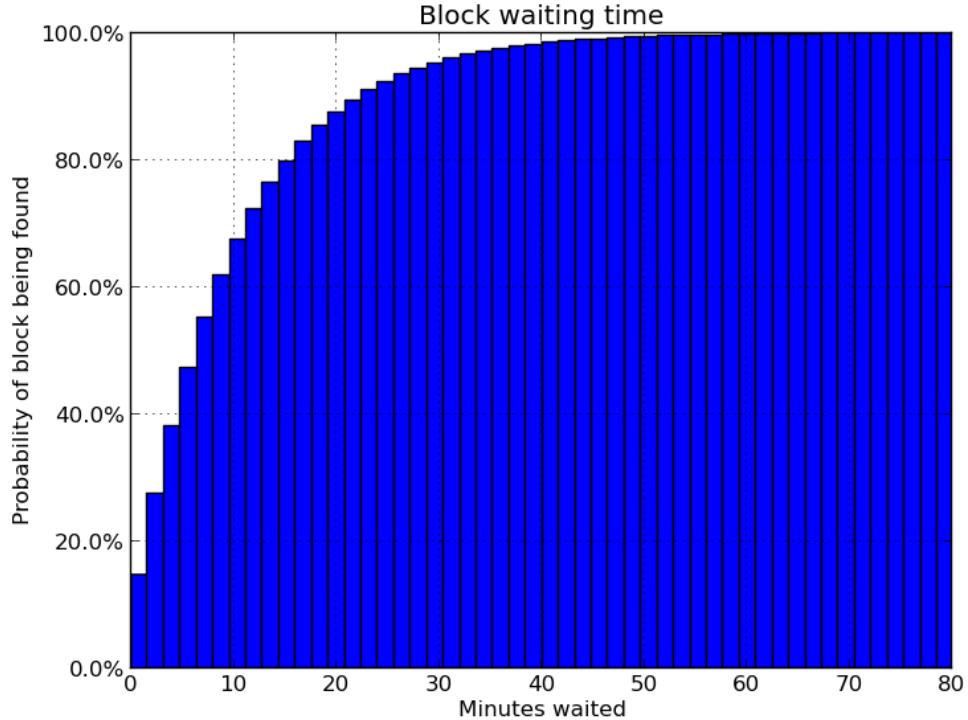


Figure 5.4: Block waiting time[3]

As the Figure 5.4 shown, in early 10 minutes, there are about 60 percent block can be confirmed. This time is also influenced by the mining fees. For the EA, it is a trade off to balance the mining fees and the confirmation time. In general, for Bitcoin, the relationship between mining fees and the confirmation time can easily fetch from the predicting Bitcoin fees API[4].

In the present experiments, even if the transaction is unconfirmed, the result can also be counted. If the transaction get the confirmation, the voting result can be more credible.

Chapter 6

Conclusions

6.1 Summary

This paper has mainly explored the basic concept of e-voting and blockchain by specifying Bitcoin address algorithm and OP_RETURN concept.

A blockchain based protocol with seven phases has been proposed later on. The definition and process of individual phase have been explained in details as well.

The entire protocol development process has also been described from the transitional software development perspective, such as how the blockchain transaction happens and some mechanism analysis of the voting system.

Finally, the paper evaluates the performance and potential security risk for the protocol, further limitations have been discussed at the end.

6.2 Conclusions

Even though the generated protocol satisfied with the properties of ballot-privacy, individual verifiability, eligibility, completeness, uniqueness, robustness, and coercion-resistance. However, it does not fulfill the needs of fairness and receipt-freeness.

In the performance evaluation, the protocol works efficiently for ring signature, especially when the number of the voter is less than 3000. Therefore, the efficiency of ring signature algorithm is limited by the number of participants.

The primary advantage of this protocol is to guarantee the authenticity of electronic voting. As every ballot will be broadcasted to the blockchain once voting starts. Moreover, as blockchain is a decentralized public ledger, ballots result are represented in a real time and cannot be modified by an individual, which satisfies the design of open-auditing.

BlockVotes confirmed the feasibility of the proposed protocol in disguise. The purpose of selecting testnet as the blockchain network , primarily rests with its free of charge and ease when comparing with Bitcoin and Ethereum. Beyond that, the high similarity degree to BitCoin network structure is another principal reason to appointed testnet to broadcast voting result.

6.3 Future work

Blockchain based e-voting protocol still have a large room for improvement, such as improving its transparency, fulfilling unconsummated functions within current status, and reducing public API.

As for BlockVotes, functions like switching more networks between BitCoins, testnet and LiteCoin can be appended. In addition, accomplishment multiple voting within one vote can be an ideal topic for further study.

Bibliography

- [1] BAUDRON, O., FOUQUE, P.-A., POINTCHEVAL, D., STERN, J., AND POUPARD, G. Practical multi-candidate election system. In *Proceedings of the twentieth annual ACM symposium on Principles of distributed computing* (2001), ACM, pp. 274–283.
- [2] BENALOH, J., AND TUINSTRA, D. Receipt-free secret-ballot elections. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing* (1994), ACM, pp. 544–553.
- [3] BITCOIN-WIKI. Confirmation - bitcoin wiki. <https://en.bitcoin.it/wiki/Confirmation>.
- [4] BITCOINFEES.21.CO. Predicting bitcoin fees for transactions. <https://bitcoinfoes.21.co/>.
- [5] CARD, D., AND MORETTI, E. Does voting technology affect election outcomes? touch-screen voting and the 2004 presidential election. *The Review of Economics and Statistics* 89, 4 (2007), 660–673.
- [6] CETINKAYA, O., AND CETINKAYA, D. Towards secure e-elections in turkey: requirements and principles. In *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on* (2007), IEEE, pp. 903–907.
- [7] CHAUM, D. L. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM* 24, 2 (1981), 84–90.
- [8] CHRISTIAN SCHAUPP, L., AND CARTER, L. E-voting: from apathy to adoption. *Journal of Enterprise Information Management* 18, 5 (2005), 586–601.
- [9] COHEN, J. D., AND FISCHER, M. J. *A robust and verifiable cryptographically secure election scheme*. Yale University. Department of Computer Science, 1985.
- [10] CRANOR, L. F., AND CYTRON, R. K. Sensus: A security-conscious electronic polling system for the internet. In *System Sciences, 1997, Proceedings of the Thirtieth Hawaii International Conference on* (1997), vol. 3, IEEE, pp. 561–570.

- [11] CZEPLUCH, J. S., LOLLIKE, N. Z., AND MALONE, S. O. The use of block chain technology in different application domains. *The IT University of Copenhagen, Copenhagen* (2015).
- [12] DEMILLO, R. A., LYNCH, N. A., AND MERRITT, M. J. Cryptographic protocols. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing* (1982), ACM, pp. 383–400.
- [13] FRAUNHOLZ, B., AND UNNITHAN, C. E-governance: enabling the french web 2.0 revolution? In *Foundations of e-government* (2007), [International Conference on E-Governance] Academic Publishing, pp. 344–359.
- [14] FUJIOKA, A., OKAMOTO, T., AND OHTA, K. A practical secret voting scheme for large scale elections. In *International Workshop on the Theory and Application of Cryptographic Techniques* (1992), Springer, pp. 244–251.
- [15] HIRT, M., AND SAKO, K. Efficient receipt-free voting based on homomorphic encryption. In *Advances in CryptologyEUROCRYPT 2000* (2000), Springer, pp. 539–556.
- [16] JASON, P. C., AND YUICHI, K. E-voting system based on the bitcoin protocol and blind signatures. *TOM 10*, 1 (2017), 14–22.
- [17] JONKER, H., MAUW, S., AND PANG, J. Privacy and verifiability in voting systems: Methods, developments and trends. *Computer Science Review 10* (2013), 1–30.
- [18] JUELS, A., CATALANO, D., AND JAKOBSSON, M. Coercion-resistant electronic elections. *Towards Trustworthy Elections 6000* (2010), 37–63.
- [19] KIWI. Bitcoin testnet sandbox. <https://testnet.manu.backend.hamburg/faucet>.
- [20] LEE, K., JAMES, J. I., EJETA, T. G., AND KIM, H. Electronic voting service using block-chain. *The Journal of Digital Forensics, Security and Law: JDSSL 11*, 2 (2016), 123.
- [21] LUO, F. Design and analysis of coercion-resistant electronic voting scheme. Master’s thesis, Fujian Normal University, 2015.
- [22] MACINTOSH, A. Characterizing e-participation in policy-making. In *System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on* (2004), IEEE, pp. 10–pp.
- [23] NAKAMOTO, S. Bitcoin: A peer-to-peer electronic cash system, 2008.
- [24] NIEMI, V., AND RENVALL, A. How to prevent buying of votes in computer elections. In *International Conference on the Theory and Application of Cryptology* (1994), Springer, pp. 164–170.

- [25] OHKUBO, M., MIURA, F., ABE, M., FUJIOKA, A., AND OKAMOTO, T. An improvement on a practical secret voting scheme. *Information Security* (1999), 771–771.
- [26] OKAMOTO, T. An electronic voting scheme. In *Advanced IT Tools*. Springer, 1996, pp. 21–30.
- [27] PETERS, G. W., AND PANAYI, E. Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In *Banking Beyond Banks and Money*. Springer, 2016, pp. 239–278.
- [28] RIVEST, R., SHAMIR, A., AND TAUMAN, Y. How to leak a secret. *Advances in CryptologyASIACRYPT 2001* (2001), 552–565.
- [29] RIVEST, R. L., SHAMIR, A., AND ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* 21, 2 (1978), 120–126.
- [30] SAKO, K., AND KILIAN, J. Receipt-free mix-type voting scheme. In *Advances in CryptologyEUROCRYPT95* (1995), Springer, pp. 393–403.
- [31] SLIM. Middleware - slim. <https://www.slimframework.com/docs/concepts/middleware.html>.
- [32] SPYCHER, O., KOENIG, R., HAENNI, R., AND SCHLÄPFER, M. A new approach towards coercion-resistant remote e-voting in linear time. In *International Conference on Financial Cryptography and Data Security* (2011), Springer, pp. 182–189.
- [33] TAKABATAKE, Y., KOTANI, D., AND OKABE, Y. An anonymous distributed electronic voting system using zerocoins.
- [34] ZHAO, Z., AND CHAN, T.-H. H. How to vote privately using bitcoin. In *International Conference on Information and Communications Security* (2015), Springer, pp. 82–96.

Appendices

Appendix A

Instruction

The project has been uploaded to the git-teaching git repository. To download the project, please run the following command.

```
git clone https://git-teaching.cs.bham.ac.uk/mod-msc-proj-2016/yxw689.git
```

To build and run the project, the running environment is required as follows.

- Mac OS X or Linux or Windows
- Apache 2.4.27 or Nginx 1.12.1
- MySQL 5.7.19 (at least 5.4)
- PHP 7.1.8 (at least 7.0)
- php7.0-gmp
- composer

To run the system, some critical but private information are not uploaded by gitignore file. Please add the .env file into the root of this project directory.

```
1 DB_DRIVER=mysql
2 DB_HOST=127.0.0.1
3 DB_DATABASE=blockvotes
4 DB_USERNAME=root
5 DB_PASSWORD=
6 DB_PORT=3306
7
8 STMP_SERVER=smtp.gmail.com
9 STMP_PORT=465
10 STMP_USERNAME=xxx@gmail.com
11 STMP_PASSWORD=password
```

The sample SQL file is generated and uploaded as the file blockvotes.sql , please import it to MySQL. Finally, please set the default directory to /public in Nginx or Apache configuration file.